ТРУДЫ

ИНСТИТУТА СИСТЕМНОГО ПРОГРАММИРОВАНИЯ РАН

PROCEEDINGS OF THE INSTITUTE FOR SYSTEM PROGRAMMING OF THE RAS

ISSN Print 2079-8156
Том 34 Выпуск 3

ISSN Online 2220-6426
Volume 34 Issue 3

Институт системного
программирования
им. В.П. Иванникова РАН

Москва, 2022

ИСП РАН

ISSN 2079-8156 (Print), ISSN 2220-6426 (Online)

Труды Института системного программирования РАН
Proceedings of the Institute for System Programming of the RAS

Труды ИСП РАН – это издание с двойной анонимной системой рецензирования, публикующее научные статьи, относящиеся ко всем областях системного программирования, технологий программирования и вычислительной техники. Целью издания является формирование научно-информационной среды в этих областях путем публикации высококачественных статей в открытом доступе.
Издание предназначено для исследователей, студентов и аспирантов, а также практиков. Оно охватывает широкий спектр тем, включая, в частности, следующие:

- операционные системы;
- компиляторные технологии;
- базы данных и информационные системы;
- параллельные и распределенные системы;
- автоматизированная разработка программ;
- верификация, валидация и тестирование;
- статический и динамический анализ;
- защита и обеспечение безопасности ПО;
- компьютерные алгоритмы;
- искусственный интеллект.

Журнал издается по одному тому в год, шесть выпусков в каждом томе.
Поддерживается открытый доступ к содержанию издания, обеспечивая доступность результатов исследований для общественности и поддерживая глобальный обмен знаниями.
Труды ИСП РАН реферируются и/или индексируются в:

Proceedings of ISP RAS are a double-blind peer-reviewed journal publishing scientific articles in the areas of system programming, software engineering, and computer science. The journal's goal is to develop a respected network of knowledge in the mentioned above areas by publishing high quality articles on open access.
The journal is intended for researchers, students, and practitioners. It covers a wide variety of topics including (but not limited to):

- Operating Systems.
- Compiler Technology.
- Databases and Information Systems.
- Parallel and Distributed Systems.
- Software Engineering.
- Software Modeling and Design Tools.
- Verification, Validation, and Testing.
- Static and Dynamic Analysis.
- Software Safety and Security.
- Computer Algorithms.
- Artificial Intelligence.

The journal is published one volume per year, six issues in each volume.
Open access to the journal content allows to provide public access to the research results and to support global exchange of knowledge. **Proceedings of ISP RAS** is abstracted and/or indexed in:

УДК004.45

Адрес: 109004, г. Москва, ул. А. Солженицына, дом 25.
Телефон: +7(495) 912-44-25
E-mail: info-isp@ispras.ru
Сайт: http://www.ispras.ru/proceedings/

Address: 25, Alexander Solzhenitsyn st., Moscow, 109004, Russia.
Tel: +7(495) 912-44-25
E-mail: info-isp@ispras.ru
Web: http://www.ispras.ru/en/proceedings

Содержание

Table of Contents

DOI: 10.15514/ISPRAS-2022-34(3)-1

Исследование свойств алгоритма слайсинга предиката пути

А.В. Вишняков, ORCID: 0000-0003-1819-220X <vishnya@ispras.ru>
Институт системного программирования им. В.П. Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25

Аннотация. Безопасный цикл разработки ПО (SDL) применяется для повышения надежности и защищенности программного обеспечения. В жизненный цикл программы добавляются этапы для проверки свойств ее безопасности. Среди прочего повсеместно применяется фаззинг-тестирование, которое позволяет обнаруживать аварийные завершения и зависания анализируемого кода. Гибридный подход, совмещающий в себе фаззинг и динамическую символьную интерпретацию, показал еще большую эффективность, чем классический фаззинг. Более того, символьная интерпретация позволяет добавлять дополнительные проверки, называемые предикатами безопасности, которые ищут ошибки работы с памятью и неопределенное поведение. Данная статья исследует свойства и характеристики алгоритма слайсинга предиката пути, который позволяет устранять избыточные ограничения из предиката пути без потери точности. В статье доказывается, что алгоритм конечен и не теряет решений. Более того, производится оценка асимптотической сложности алгоритма.

Keywords: динамическая символьная интерпретация; DSE; предикат пути; слайсинг; устранение избыточных ограничений; анализ бинарного кода

Для цитирования: Вишняков А.В. Исследование свойств алгоритма слайсинга предиката пути. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 7-12. DOI: 10.15514/ISPRAS–2022–34(3)–1

Благодарности: Работа поддержана грантом РФФИ № 20-07-00921 А

Analyzing properties of path predicate slicing algorithm

A.V. Vishnyakov, ORCID: 0000-0003-1819-220X <vishnya@ispras.ru>
Ivannikov Institute for System Programming of the RAS,
25, Alexander Solzhenitsyn Str., Moscow, 109004, Russia

Abstract. Security development lifecycle (SDL) is applied to improve software reliability and security. It extends program lifecycle with additional testing of security properties. Among other things, fuzz testing is widely used, which allows one to detect crashes and hangs of the analyzed code. The hybrid approach that combines fuzzing and dynamic symbolic execution showed even greater efficiency than classical fuzzing. Moreover, symbolic execution empowers one to add additional runtime checks called security predicates that detect memory errors and undefined behavior. This article explores the properties of the path predicate slicing algorithm that eliminates redundant constraints from a path predicate without accuracy loss. The article proves that the algorithm is finite and does not lose solutions. Moreover, the algorithm asymptotic complexity is estimated.

Keywords: dynamic symbolic execution; DSE; path predicate; slicing; irrelevant constraints elimination; binary analysis

For citation: Vishnyakov A.V. Analyzing properties of path predicate slicing algorithm. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 7-12 (in Russian). DOI: 10.15514/ISPRAS–2022–34(3)–1

Acknowledgments: This work was supported by RFBR grant 20-07-00921 A

1. Введение

Безопасный цикл разработки ПО (SDL) [1-3] применяется для повышения надежности и защищенности программного обеспечения. В жизненный цикл программы добавляются этапы для проверки свойств ее безопасности. Среди прочего повсеместно применяется фаззинг-тестирование [4, 5], которое позволяет обнаруживать аварийные завершения и зависания анализируемого кода. Гибридный подход [6-9], совмещающий в себе фаззинг и динамическую символьную интерпретацию [10, 11], показал еще большую эффективность, чем классический фаззинг. Более того, символьная интерпретация позволяет добавлять дополнительные проверки, называемые предикатами безопасности [12], которые ищут ошибки работы с памятью и неопределенное поведение.

Динамическая символьная интерпретация осуществляется следующим образом. Сначала каждому входному байту программы (например, байту входного файла) ставится в соответствие свободная символьная переменная. Далее производится интерпретация конкретного пути выполнения программы, заданного некоторыми фиксированными входными данными. Инструкции, которые оперируют над символьными значениями (зависящими от входных данных) порождают символьные выражения (формулы) в соответствии с их операционной семантикой. Изменения памяти и регистров обновляют символьное состояние, которое содержит отображение из регистров и байтов памяти в символьные выражения. Каждое условное ветвление добавляется в предикат пути. Конъюнкция ограничений из предиката пути имеет модель — значения символьные переменных (входных байтов программы), которые проведут программу по тому же пути выполнения.

Обычно выделяют две задачи, которые позволяет решать динамическая символьная интерпретация: увеличение покрытия кода (открытие новых путей выполнения) и поиск ошибок.

Открытие новых путей осуществляется путем инвертирования условия переходов. Для этого составляется конъюнкция ограничений из предиката пути до инвертируемого перехода с его отрицанием. Таким образом, модель для полученного предиката проведет программу по другому пути выполнения.

Поиск ошибок [12] производится за счет решения предикатов безопасности, описывающих условие возникновение ошибки (например, равенства делителя нулю). Если аналогично составить конъюнкцию предиката пути и предиката безопасности, то в результате решения могут получиться входные данные, активирующие ошибку. Таким образом, предикат пути отвечает за достижение программой точки проявления ошибки, а предикат безопасности — за ее реализацию.

2. Алгоритм слайсинга предиката пути

Алгоритм 1 слайсинга предиката пути [13] (вдохновлен KLEE [14]) позволяет устранять избыточные ограничения из предиката пути без потери точности. Алгоритм принимает на вход целевое ограничение *cond* (условие для инвертирования перехода или предикат безопасности) и конъюнкцию ограничений из предиката пути П до точки проверки целевого ограничения. Предварительно вычисляются используемые символьные переменные для каждого ограничения в предикате пути и целевого условия. Их вычисление производится путем обхода абстрактных синтаксических деревьев для символьных выражений. Изначально множество переменных слайсинга *vars* содержит символьные переменные, от которых зависит *cond*. Далее происходят проходы по всем ограничениям *c* из предиката пути П. Если переменные *vars* пересекаются с используемыми переменными в *c*, то множество *vars* пополняется символьными переменными из *c*. Проходы происходят до тех пор, пока множество *vars* пополняется новыми элементами. В результате будут получены все переменные *vars*, которые транзитивно зависят от переменных в целевом ограничении *cond*. В конце производится последний проход по ограничениям из предиката пути П. Ограничения

из П, используемые переменные которых пересекаются с *vars*, составляются в конъюнкцию вместе с ограничением *cond*. Полученный предикат $П_S$ будет отвечать решаемой задаче: инвертированию перехода или проверке предиката безопасности. Таким образом, берется лишь часть ограничений из предиката пути. При этом далее будет показано, что алгоритм решает ту же задачу.

Алгоритм 1. Алгоритм слайсинга предиката пути
Algorithm 1. Path predicate slicing algorithm

```
Входные данные: cond – ограничение для инвертирования целевого  перехода
(или предикат безопасности), П – предикат пути (ограничения пути до
целевого перехода).
    vars ← used_variables (cond)                ▷переменные слайсинга
    change ← vars
    while change ≠ ∅
      change ← vars
      for all c ∈ П                  ▷итерирование по ограничениям пути
        if vars ∩ used_variables(c) ≠ ∅ then
          vars ← vars ∪ used_variables (c)
      change ← vars \ change
    П_S ← cond   ▷ограничение для инвертирования/предикат безопасности
    for all c ∈ П do                 ▷итерирование по ограничениями пути
     if vars ∩ used_variables(c) ≠ ∅ then
       П_S ← П_S ∧ c
    return П_S
```

Перед тем как приступить к изучению свойств и характеристик алгоритма 1 слайсинга предиката пути, приведем необходимые определения.

Определение 1. *Модель для предиката* $P(v_1, \ldots, v_n)$ *– это такая подстановка* $v_i \leftarrow c_i,\ i = 1..n$, *где* c_i *– константы, при которой предикат истинен:* $P(\mathrm{c}_1, \ldots, \mathrm{c}_n) \equiv 1$.

Определение 2. *Символьная переменная – свободная переменная, которая ставится в соответствие с каждым входным байтом программы.*

Определение 3. *Предикат пути* П, *построенный для пути выполнения, заданного конкретными значениями символьных переменных* α_i *(входными байтами программы) – это конъюнкция ограничений над символьными переменными и константами, каждая модель которой проведет программу по тому же пути выполнения.*

Следует отметить, что из определения предиката пути следует, что значения символьных переменных α_i являются для него моделью, т.к. это изначальные входные байты программы, которые проводят ее по заданному пути.

3. Свойства и характеристики алгоритма слайсинга предиката пути

Теорема 1. *Алгоритм 1 слайсинга предиката пути конечен, т.е. завершается за конечное число итераций.*

Доказательство. Алгоритму требуются предварительно вычисленные множества используемых символьных переменных *used_variables (c)* для всех ограничений в предикате пути и ограничения *cond* ($c \in П \vee c = cond$). Производится обход деревьев символьных выражений (для ограничений c) в ширину и составляется множество номеров, используемых в них переменных. Алгоритм поиска в ширину является конечным для деревьев с конечным числом вершин, что справедливо для деревьев символьных выражений. Множество

ограничений в предикате пути Π также конечно. Следовательно, алгоритм вычисления используемых символьных переменных конечен.

Проход по всем ограничениям $c \in \Pi$ осуществляется за конечное число шагов, т.к. множество ограничений в предикате пути Π конечно. Из этого следует, что вложенный цикл на каждой итерации **while**, который обновляет *vars*, и цикл, конструирующий итоговый предикат Π_S, конечны. Для доказательства конечности всего алгоритма 1 остается только доказать, что итераций цикла **while** с условием завершения *change* = ∅ конечное число.

Каждая итерация цикла **while** заключается в переборе всех ограничений *c* из предиката пути Π с возможным обновлением множества *vars*. При этом, если на итерации не происходит изменение множества *vars*, выполняется условие завершения цикла.

Для $i \in \mathbb{N}$ обозначим $V_i = vars_i$ – множество переменных слайсинга *vars* в начале *i*-й итерации. Тогда $V_1 = used\_variables(cond)$, а $V_i \subseteq V_{i+1}$, так как множество переменных *vars* на каждой итерации включает в себя множество переменных предыдущей итерации: $V_{i+1} = V_i \cup U_i$, где $V_i \cap U_i = \emptyset$, и U_i – множество добавленных в *vars* переменных.

Заметим, что на каждой итерации возможны 2 варианта работы алгоритма:

1) происходит перебор всех ограничений из предиката пути Π без изменения множества V_i, и цикл завершается;
2) множество V_i пополняется новыми элементами.

Покажем, что цикл **while**, отбирающий символьные переменные для слайсинга, имеет конечное число итераций. Докажем «от противного»: предположим, что цикл бесконечен. Следовательно, никогда не выполняется условие завершения цикла. Это возможно (в соответствии с вариантами его работы) в том и только в том случае, когда на каждой итерации *i* множество V_i пополняется хотя бы одним новым элементом x_i (при этом x_i может быть не единственным новым элементом на *i*-й итерации: $V_{i+1} = V_i \cup U_i$, где $V_i \cap U_i = \emptyset$, а $x_i \in U_i$). Более того, множества V_i строго вложены ($V_i \subset V_{i+1}$), в противном случае стало бы истинным условие завершения цикла $change = \emptyset$ (другими словами, $V_i = V_{i+1}$). Исходя из строгой вложенности множеств V_i, все элементы x_i различны. Тогда существует бесконечная последовательность $\{x\}_i$, такая что $x_i \in V_{i+1}$, состоящая из различных элементов. Обозначим множество элементов последовательности как $X = \{x \mid x \in \{x\}_i\}$. Поскольку каждое из множеств V_i является подмножеством множества V_P всех символьных переменных программы, то множество элементов последовательности $X \subseteq V_P$. Следовательно, мощность множества $|X| \leq |V_P|$. Но множество всех символьных переменных V_P конечно: $|V_P| = N$, где $N \in \mathbb{N}$. А значит, $|X| \leq N$, и последовательность $\{x\}_i$ не может содержать бесконечное число различных элементов. Получили противоречие с изначальным определением $\{x\}_i$, из чего следует, что цикл отбора символьных переменных для слайсинга конечен, а значит, и алгоритм 1 слайсинга предиката пути конечен.

Теорема 2. *Пусть предикат пути П строился по пути выполнения, заданному конкретными значениями символьных переменных* $v_i \leftarrow \alpha_i$, $i = 1..n$, $n \in \mathbb{N}$, *где* v_i *– символьная переменная, а* α_i *– ее значение (входной байт программы). Если конъюнкция ограничений предиката пути* Π *и ограничения* cond *(*$\Pi \wedge cond$*) выполнима, то предикат* Π_S *(полученный в результате применения алгоритма 1 слайсинга предиката пути к* Π *и* cond*) тоже выполним, и для любой его модели* $v_j \leftarrow \beta_j, j \in J \subseteq \{1, \ldots, n\}$ *подстановка* $v_j \leftarrow \beta_j$, $v_k \leftarrow \alpha_k$, $k \in K = \{1, \ldots, n\} \setminus J$ *(*α_k *– входные байты программы) является моделью для предиката* $\Pi \wedge cond$.

Доказательство. Исходя из описания алгоритма, предикат Π_S состоит из конъюнкции части ограничений из предиката пути Π и ограничения *cond*: $\Pi_S = P \wedge cond$, $\Pi = P \wedge Q$. Предикат $\Pi \wedge cond$ выполним, а значит, и предикат $P \wedge Q \wedge cond$ выполним. Следовательно, предикаты $P \wedge cond = \Pi_S$ и Q тоже выполнимы.

Обратим внимание на последнюю итерацию цикла **while**, когда выполняется условие выхода из цикла, а именно множество переменных *vars* остается неизменным в результате

итерирования по всем ограничениям *c* из предиката пути Π. Для этого на каждой итерации вложенного цикла **for** не должно происходить пополнение множества *vars*. Это возможно в том и только том случае, когда либо *c* не использует ни одну переменную из *vars*, либо $used\_variables(c) \subseteq vars$ (в противном случае множество *vars* пополнилось бы переменными из $used\_variables(c) \setminus vars$). Другими словами, $\forall c \in \Pi\ (used\_variables(c) \subseteq vars) \vee (used\_variables(c) \cap vars = \emptyset)$.

Из последнего цикла **for** алгоритма следует, что конъюнкты *P* зависят хотя бы от одной переменной из множества переменных *vars*, а конъюнкты *Q*, напротив, не зависят ни от одной переменной из *vars*. Из доказанного выше утверждения следует, что каждый конъюнкт *P* зависит от подмножества переменных из *vars*. Таким образом, каждый конъюнкт *P* содержит только переменные из *vars* и не содержит других переменных. Значит, множества переменных в предикатах *P* и *Q* не пересекаются.

Из первой строчки алгоритма следует, что множество переменных в *cond* является подмножеством *vars*, а значит, множества переменных в *cond* и *Q* тоже не пересекаются. Следовательно, множество переменных предиката $P \wedge cond = \Pi_S$ не пересекается с множеством переменных предиката *Q*.

Пусть $v_j \leftarrow \beta_j,\ j \in J \subseteq \{1,\dots,n\}$ – модель для предиката Π_S. Тогда $v_k \leftarrow \alpha_k,\ k \in K = \{1,\dots,n\} \setminus J$ – модель для предиката *Q*, т.к. *Q* выполним, множества переменных в Π_S и *Q* не пересекаются, $\Pi = P \wedge Q$, а $v_i \leftarrow \alpha_i$ – модель для предиката пути Π из определения. Осталось найти модель для предиката $\Pi \wedge cond = P \wedge Q \wedge cond = (P \wedge cond) \wedge Q = \Pi_S \wedge Q$. Моделью является только что найденная подстановка непересекающихся множеств переменных: $v_j \leftarrow \beta_j, v_k \leftarrow \alpha_k, j \in J \subseteq \{1,\dots,n\}, k \in K = \{1,\dots,n\} \setminus J$.

Следствие 2.1. *Для получения входных байтов программы, приводящих к инвертированию целевого перехода (или проявлению ошибки), достаточно заменить часть входных байтов v_j на значения β_j из модели для предиката Π_S, полученного в результате применения алгоритма 1 слайсинга предиката пути.*

Теорема 3. *Асимптотическая сложность алгоритма 1 – $O(|\Pi| * |V_P|^2 * log(|V_P|))$, где $|\Pi|$ – число ограничений в предикате пути, а $|V_P|$ – число всех символьных переменных (входных байтов) программы.*

Доказательство. Номера используемых символьных переменных хранятся в двоичных (красно-черных) деревьях поиска. Операция объединения двух деревьев имеет сложность $O(|V_P| * log(|V_P|))$. Операция определения факта пересечения (пересекаются/не пересекаются) двух отсортированных контейнеров имеет линейную сложность $O(|V_P|)$. Определение факта изменения множества *vars* имеет константную сложность: для этого достаточно сравнить количество переменных *vars* в начале и в конце итерации цикла. Конъюнкция двух предикатов также имеет константную сложность, т.к. предикаты представляются абстрактными синтаксическими деревьями, а их конъюнкция равносильна созданию одной вершины.

Предварительно до начала работы алгоритма вычисляются множества используемых символьных переменных для ограничений *c* из предиката пути Π и ограничения *cond*. Для этого производится обход по абстрактным синтаксическим деревьям. Таким образом, вычисление используемых переменных имеет сложность $O(|\Pi| * |V|)$, где $|V|$ – максимальное число вершин в дереве. Следует отметить, что используемые переменные в ограничениях предиката пути Π могут быть вычислены единожды, а алгоритм применен многократно.

Оценим вычислительную сложность непосредственно самого алгоритма 1 слайсинга предиката пути. В худшем случае цикл **while** будет иметь $|V_P|$ итераций. Это возможно, когда на каждой итерации множество *vars* пополняется ровно одной переменной. Таким образом, асимптотическая сложность цикла **while**: $O(|V_P| * |\Pi| * (|V_P| + |V_P| * log(|V_P|))) = O(|\Pi| * |V_P|^2 * log(|V_P|))$. Вычислительная сложность каждой итерации внутреннего цикла

for состоит из суммы сложности пересечения и объединения множеств. Последний цикл **for** осуществляет $|\Pi|$ пересечений множеств: $O(|\Pi| * |V|)$. Итого асимптотическая сложность алгоритма – $O(|\Pi| * |V_P|^2 * log(|V_P|) + |\Pi| * |V_P|) = O(|\Pi| * |V_P|^2 * log(|V_P|))$.

4. Заключение

В данной статье были изучены свойства и характеристики алгоритма слайсинга предиката пути [13]. Было показано, что алгоритм конечен и позволяет уменьшать число избыточных ограничений из предиката пути без потери решений. Более того, была проведена оценка асимптотической сложности алгоритма.

Список литературы / References

[1] M. Howard and S. Lipner. The security development lifecycle. SDL: A Process for Developing Demonstrably More Secure Software. Microsoft Press, Redmond, 2006, 348 p.

[2] ISO/IEC 15408-3:2008: Information Technology – Security Techniques –Evaluation Criteria for It Security – Part 3: Security Assurance Components. ISO Geneva, Switzerland, 2008.

[3] ГОСТ Р 56939-2016: Защита информации. Разработка безопасного программного обеспечения. Общие требования. Национальный стандарт Российской Федерации, 2016 / GOST R 56939-2016: Information Protection. Secure Software Development. General Requirements, National Standard of Russian Federation, 2016 (in Russian).

[4] K. Serebryany. Continuous Fuzzing with libFuzzer and AddressSanitizer. In Proc. of the the 2016 IEEE Cybersecurity Development (SecDev), 2016, p. 157.

[5] A. Fioraldi, D. Maier et al. AFL++: combining incremental steps of fuzzing research. In Proc. of the 14th USENIX Workshop on Offensive Technologies (WOOT 20), 2020, 12 p.

[6] I. Yun, S. Lee et al. QSYM: a practical concolic execution engine tailored for hybrid fuzzing. In Proc. of the 27th USENIX Security Symposium, 2018, pp. 745-761.

[7] S. Poeplau and A. Francillon. Symbolic execution with SymCC: don't interpret, compile! In Proc. of the 9th USENIX Security Symposium (USENIX Security 20), 2020, pp. 181-198.

[8] S. Poeplau and A. Francillon. SymQEMU: compilation-based symbolic execution for binaries. In Proc. of the 2021 Network and Distributed System Security Symposium, 2021, 18 p.

[9] L. Borzacchiello, E. Coppa, and C. Demetrescu. FUZZOLIC: mixing fuzzing and concolic execution. Computers & Security, vol. 108, 2021, article no. 102368.

[10] E. J. Schwartz, T. Avgerinos, and D. Brumley. All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask). In Proc. of the 2010 IEEE Symposium on Security and Privacy, 2010, pp. 317-331.

[11] R. Baldoni, E. Coppa et al. A survey of symbolic execution techniques. ACM Computing Surveys, vol. 51, issue 3, 2018, article no. 50, pp. 1-39.

[12] A. Vishnyakov, V. Logunova et al. Symbolic security predicates: hunt program weaknesses. In Proc. of the 2021 Ivannikov ISPRAS Open Conference (ISPRAS), 2021, pp. 76-85.

[13] A. Vishnyakov, A. Fedotov et al. Sydr: cutting edge dynamic symbolic execution. In Proc. of the 2020 Ivannikov ISPRAS Open Conference (ISPRAS), 2020, pp. 46-54.

[14] C. Cadar, D. Dunbar, and D. R. Engler. KLEE: unassisted and automatic generation of high-coverage tests for complex systems programs. In Proc. of the 8th USENIX Conference on Operating Systems Design and Implementation, pp. 209-224, 2008.

Информация об авторах / Information about authors

Алексей Вадимович ВИШНЯКОВ – младший научный сотрудник отдела компиляторных технологий в Институте системного программирования им. В.П. Иванникова РАН, закончил бакалавриат и магистратуру ВМК МГУ. Сфера научных интересов: компьютерная безопасность, жизненный цикл безопасной разработки (SDL), символьная интерпретация, фаззинг, автоматическое обнаружение ошибок, анализ бинарного кода и компиляторы.

Alexey Vadimovich VISHNYAKOV works for the Compiler Technology Department at Ivannikov Institute for System Programming of the RAS, obtained BSc degree and M.D. in the Faculty of Computational Mathematics and Cybernetics at Lomonosov Moscow State University. Research interests: computer security, security development lifecycle (SDL), symbolic execution, fuzzing, automatic bug detection, binary analysis, and compilers.

DOI: 10.15514/ISPRAS-2022-34(3)-2

Unified type system for the modern general-purpose programing language

A.V. Kanatov, ORCID: 0000-0001-8844-7365 <alexey.v.kanatov@gmail.com>
E.A. Zouev, ORCID: 0000-0001-7136-8935 <eugene.zueff@gmail.com>
Innopolis University,
1, Universitetskaya Str., Innopolis, 420500, Russia

Abstract. The paper presents an overview of the type system, which supports the convergence of procedural, object-oriented, functional, and concurrent programming paradigms relying on static type checking with smart type inference support and the ability to ensure dynamic type safety as well. The key element of the type system is that it is fully based on just 2 basis constants and all other constructions are derived.

Keywords: object; constant object; type; unit; class; module; interface; conformance; compatibility; type conversions; setters; reference and value objects; immutability

For citation: Kanatov A.V., Zouev E.A. Unified type system for the modern general-purpose programing language. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 13-30. DOI: 10.15514/ISPRAS–2022–34(3)–2

Унифицированная система типов для современного языка программирования общего назначения

А.В. Канатов, ORCID: 0000-0001-8844-7365 <alexey.v.kanatov@gmail.com>
Е.А. Зуев, ORCID: 0000-0001-7136-8935 <eugene.zueff@gmail.com>
Университет Иннополис,
420500, Россия, г. Иннополис, ул.Университетская, д.1

Аннотация. Данная статья представляет обзор системы типов, которая отражает тенденцию конвергенции процедурной, объектно-ориентированной, функциональной и параллельной парадигм программирования, базирующейся статической проверке типов с использование их автоматического вывода и возможностью гарантии целости типов при выполнении. Ключевым аспектом системы типов является ее базирование на 2х атомарных константах и выводах всех остальных типов из этого базиса.

Keywords: объект; константный объект; тип; юнит; класс; модуль; интерфейс; конформность типов; согласованность типов; преобразования типов; процедуры установки значений; объекты ссылки и объекты значения; понятие неизменяемости

Для цитирования: Канатов А.В, Зуев Е.А. Унифицированная система типов для современного языка программирования общего назначения. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 13-30. DOI: 10.15514/ISPRAS–2022–34(3)–xx. DOI: 10.15514/ISPRAS–2022–34(3)–2

1. Introduction

The type system sets the basis for the reliable programming language and allows programmers to effectively express software design solutions using the power of the particular programming language raising the productivity of the software development process.

The modern tendency of convergence of different programming paradigms (merging procedural programming, structured programming, object-oriented programming, functional programming, and concurrent programming) forces the type system to support this.

In this paper, a highly condensed overview of the type system is presented and a programming language called SLang is used for the illustration of concepts. Necessary syntax constructs will be presented using simple notation based on conventions, where [term] means optional, {term} may be repeated zero or more times, term1 | term2 is the selection of term1 or term2, **bold font** is used to highlight keyword or special symbols.

Next is to define the notion of type as an important characteristic of every object during execution time (runtime). The type fixes the number of operations and their properties (signatures) as well as the size of memory required to store the object (number, valid values, and types of object attributes). So, a type is an abstraction used to describe the structure and behavior of objects.

Authors rely on concepts that are well-known by a broad audience of programmers and terms like class or variable will be used without formal definitions. Some definitions will be given right now to simplify the understanding of examples.

The unit is a named set of members, where a member can be a routine or an attribute. Routines stand for actions while attributes stand for data. If a routine returns some value as a result of its execution, we call it a function otherwise a procedure. If an attribute can change its value during the program execution, we call it a variable attribute (or simply variable) otherwise we call it a constant attribute (or simply constant or immutable attribute). Unit is very similar to class and the difference is that the unit incorporates characteristics of classes and modules (The term module is used like it was introduced in Ada (package) [2], Modula-2 (module) [4] – a generally available collection of data and routines with initialization) in one concept and the foundation for types.

So, the most important type is the unit-based type, and let's review units first.

2. Units

Any unit is a named collection of attributes or members. Such a definition sets away routines because they can be treated as constant attributes of routine type initialized with the routine signature and body. Units have other characteristics related to inheritance and usage; they will be explored below.

Every unit defines a type, and the name of the unit will be used as a type name. Such type is a unit-based type. The formal definition of the unit is

```
UnitDeclaration:
[final] [ref|val|concurrent|abstract|extend]
unit Identifier [AliasName] [FormalGenerics]
   [InheritDirective] [UseDirective]
{
   MemberSelection |
   InheritedMemberOverriding |
   InitProcedureInheritance |
   ConstObjectsDeclaration |
   MemberDeclaration
}
[InvariantBlock]
end
```

Unit is a central component and has a lot of elements. For the purpose of the paper, only `ConstObjectsDeclaration` and `MemberDeclaration` will be reviewed.

Specifiers indicate some characteristics of the unit and objects which can be built based on this unit-based type.

As a unit may inherit members from other units' `final` specifier prevents further inheritance from this unit.

`ref` | `val` specifies the default form of objects which will be created using this unit as a type. The example below explains the difference. All objects of type Integer are to be values but not references to the integer number.

```
val unit Integer ... end
i: Integer is 5
```

Here, `i` is a value object. **`is`** works as a combination of entity declaration with initialization.

```
ir: ref Integer is 5
```

Here, `ir` is a reference object.

The default kind of object is a reference one. It's important to note that **ref** | **val** specifiers apply both to units and for particular objects and attributes. The unit-based type itself is not related to the form of objects of this type.

The **`concurrent`** specifier indicates that objects of this unit will be processed (executed) by a processing element that is different from the one which is used for all objects which are not marked as concurrent. The processing element is a general term for a physical processor, thread, process, remote server, or whatever computing machine. The mapping between the concurrent unit and actual physical executors is to be done outside of the programming language and it is not described here.

```
concurrent unit Philosopher
   // There are 5 of them
   // eating spaghetti...
end
```

If we like to ensure that there will be no objects created for the unit, it is to be marked as **abstract**. Of course, if there are some abstract routines within the body of the unit it is not possible to create an object of this unit type. So, it is not mandatory to mark such units as abstract as the compiler knows this, but if one likes to prevent objects creation for some units with having all routines as non-abstract then marking the unit abstract will allow to make it. Example:

```
abstract unit AnArray[G]
```

The **`extend`** specifier allows to extend already compiled unit with new members. For example: Source #1 has

```
unit A
   foo do ... end
end
```

Source #2 has

```
extend unit A
   goo do ... end
end
```

Source #3 has

```
a is new A
a.foo
a.goo
```

Here, the second call to routine goo is valid if and only if the `A` unit extension was provided. In other words, sources #1, #2, and #3 will be compiled separately, but a compilation of Source #2 relies on the interface from Source #1, and a compilation of #Source 3 relies on interfaces of #1 and #2 sources.

As in many other OO-languages, **`final`** will not work together with **`abstract`** as it is out of sense to create a unit when it is not possible to create objects of this unit and unit descendants are prohibited as well.

After the unit name aliasing name can be put (`AliasName`). It can be used to give an alternative name for the unit-based type. Some programmers do not like Integer they prefer **`int`** or `INTEGER`

```
val unit Integer alias Int
```

As we follow the style guideline that unit names should start with the capital letter.

Aliasing is a part of the type system. Although it does not create a new type it affects type equivalence. It also allows to create unique names, to use short names instead of long ones. So, alias declaration can be put at the global level of the source like in the following example:

```
alias StandardInputOutput as IO
IO.print ("Hello world!\n")
```

However, the name `StandardInp Sta utOutput` still stays as a valid name of the unit. So, unit-based types `ndardInputOutput` and `IO` refer to the same type.

`FormalGenerics` is an optional parametrization of the unit with some unit-based type, or value, or routine. For such kind of parametrization, the term *genericity* is used. The notation uses square brackets.

```
abstract unit AnArray[G]
```

where `G` is the name of the type which is to be provided to get particular instantiation of the unit-based type.

```
abstract unit OneDimentionalArray
        [G extend Any init()]
```

`G` can be constrained meaning that any type which is used for instantiation is to be conformant to the type specified as a constraint. In case of the example above it should be a descendant of `Any`. If it's necessary to create objects of the formal generic type, we need to know which initialization procedure (constructor) to be used – in this example requirement for the instantiating type is to have an initialization procedure without arguments.

```
unit Array [G extend Any init(),N: Integer]
   extend OneDimentionalArray[G]
```

Here we have two generic parameters and the second one is the constant of the type which is specified.

`InheritDirective` specifies from which units this unit inherits members. Here it is essential just to mention that inheritance is multiple and does not use the subobject concept. Every unit member is inherited on its own. The keyword **`extend`** (which is well-known by many programmers) is used to highlight the set of parent (base) units. The example above in the section on generics shows that unit Array inherits all members from the unit `OneDimentionalArray`.

`UseDirective`. The idea of a module as a container of functionality seems to be similar to that of [1]. However, there are some other differences between classes and modules. The key point is that based on the class one may create an unlimited number of objects while for the module there will be just one object created and properly initialized. Modules are created and initialized implicitly while object creation is a special statement or expression. So, it implies that a unit may be used as a module if and only if it has no initialization procedure or at least one initialization procedure with no arguments. The example below highlights that

```
alias StandardInputOutput as IO
IO.print("Hello world!\n")
```

Here, `IO` is the name of the module which is created and initialized at some moment of the program execution (actually, two options are possible – to create all module objects at the program start or right before the first access to the module members).

```
io is new IO.init(IO.TextMode)
```

Here, `io` is an object which is initialized with the creation of a new object of type `IO`

```
io1 is new IO.init(IO.GraphicalMode)
```

An unlimited number of objects like `io1` can be created, initialized, and used when uint is used as a type.

```
io.print("Hello world!\n")
```

In this example, `IO` is a global module which is available across all components of the program, but if we like to have a module dedicated to the unit hierarchy (current unit and all its descendants (derived units)) then we can specify it using `UseDirective` syntax like this

```
unit A use B ... end
```

So, inside of `A` all calls of the form `B.foo()` are calls to the functionality fo the module B.

If access to the global unit `B` is required, then it is possible to give a local name for the `B` which is used as a module for `A` unit hierarchy like this

```
unit A use B as BB ... end
```

So, inside of `A` all calls of the form `B.foo()` are calls to the functionality of the global module `B`, and calls like `BB.foo()` are calls to the local module.

Next is the `MemberDeclaration` section of the unit declaration.

2.1 Unit members

There are 3 kinds of unit members – unit routines (procedures or function), unit attributes (data fields), and unit initialization procedures. By default, all unit members are visible for unit descendants and clients and this visibility implies an ability to call routines and read the attributes while clients are not able to change the value of attributes and override routines. Of course, there should be a mechanism to change the visibility of the particular unit member or a group of members. One may limit visibility in the following ways

```
unit A
   rtn1 do end
   // Routine ‘rtn1’ is visible for all
   // descendants and clients

   {} rtn2: T do end
   // Routine ‘rtn2’ is visible for all
   // descendants only

       {this} rtn3 do end
   // Routine ‘rtn3’ is visible only for
   // the current unit A

       {B, C} rtn4 do end
   // Routine ‘rtn4’ is visible for all
   // descendants and clients B and C

       {}: // Group of members with the
     // same visibility

         attr1: T1
     var attr2: T2

       end
end
```

One may notice that the second attribute is marked with the **var** specifier while the first one has nothing. By default, all attributes are in fact constants with initialization. So adding **var**, it will be possible to change the value of this attribute and its content at any time during program execution. The concept of ‘constantness’ (immutability) will be explored later but now let's review initialization procedures.

2.2 Unit initialization procedures

When an object is being created there should be a way to put it into a consistent stage that fully matches its invariant. That is why an initialization procedure is needed (a constructor or a creation procedure in other programming languages) as the only task it has is to initialize all attributes of the unit. The straightforward choice for the name was “init” and as the name of the initialization procedure is known it can be skipped when a new object is being created, as well the empty parenthesis if init has not arguments. So, here is a reduced example of the initialization procedure of unit Boolean

```
val unit Boolean
    init do
        data := 0xb
```

```
    end
    {} var data:
            Bit[Platform.BooleanBitsCount]
end // Boolean
```

Variable attribute `data` that is not visible to the clients of `Boolean` is initialized with zero, interpreted as false. So, here is implicit magic (no defaults) – all units including basic ones explicitly define initial values for all their attributes.

```
b is new Boolean
```

This means that object `b` is created with the value false. This is a short cut for the declaration like this

```
b: Boolean is new Boolean.init()
```

A unit may have several init procedures and the programmer can select the one which is required for the particular case.

```
unit A
    init (a1: T1; a2: T2) do end
    {} init (a: A) do end
    foo do
        a is new A(this)
    end
end
```

In this example, `a` is a local `attribute` of routine `foo`, created by **`new`** and initialized with the second **`init`** procedure which is available only for this unit.

```
a1 is new A.init(new T1, new T2)
a2 is new A(new T1, new T2)
```

As **`init`** name is known it can be skipped while creating new objects. Outside of unit A only one initialization procedure is visible and has to be used while creating new objects.

2.3 Unit invariant

Unit invariant is a set of predicates that state when objects of this unit type and its descendants be consistent. It is a requirement to objects consistency – that is why the keyword **`require`** is being used to highlight that.

```
abstract unit Numeric
   one: as this abstract
   zero: as this abstract
   // Declarations of * and +
   // are skipped
require
   this = this * one
   zero = this * zero
   this = this + zero
end // Numeric
```

Every numeric object of a type which is a descendant of `Numeric` should implement concepts of `one` (1) and `zero` (0) and should be consistent with the invariant stated in `Numeric`. So, if some operation is applied to an object of some type then after completing the operation the unit invariant is to be checked to ensure that object is still in the consistent state and ready again to perform new operations.

2.4 Unit setters and getters

As all visible unit attributes are directly accessible for clients and descendants – their names are effective getters. For setters, it is rather convenient to use syntax like `a.b := expr` instead of `a.b.set_b(expr)`, but semantically they have the same meaning – we need to call some procedure which will set the value of some unit attribute to a proper state. So, the straightforward approach is to use `:=` as the name of the setter and associate it with the attribute declaration.

```
unit A
```

```
    var attr1: T1 :=
         alias setAttr1 (other: T2) do...end
end
```

In unit A, the variable attr1 has a setter with an argument of type T2 and this setter has an additional name setAttr1.

After that, objects can be defined and setter used. Both last statements do the same – they set attribute a to the same value.

```
a is new A
a.attr1 := new T2
a.setAttr1 (new T2)
```

3. Immutability

As a: Type is a declaration of a constant attribute, a similar scheme is applied for routine arguments. It implies that it is not possible to assign new values to formal arguments. Other implications of the constantness status of an attribute that it is not possible to change the state of an object. It implies that any call to routines which change such state are statically detected by the compiler and a proper error message is generated. So, if an attribute is marked as **var** attribute – assignment to this attribute and any correct routine call will be a valid action. If no mark in place or attribute is marked as **rigid**, then the attribute can only be initialized once, and then it will keep its value. In the case of **rigid**, the whole object tree accessible from this object is immutable. So, **rigid** implies deep constantness of an attribute while no mark means shallow constantness.

As data attributes can be of two kinds – reference and value, the semantic of the assignment statement is a bit different. There are four possible cases

```
ref1 := ref2
// Copy ref2 into ref1.
// After the assignment, they both point
// to the same object.

val1 := val2
// Field by field copy of the object named
// val2 into the corresponding fields of the
// object named val1.

ref := val
// Clone the object named val and reference
// to this clone is put into ref.

val := ref
// Field by field copy all fields of the
// object pointed by ref into the
// corresponding fields in the object named
// val.
```

Once again: the type itself is agnostic to the kind of objects which will be created. So, **ref** and **val** objects of the same type can be easily assigned to each other (boxing unboxing is done by the compiler automatically). The example below illustrates this.

```
unit A
   var attr: Type := (other: Type) do
                          attr := other
                      end
   foo (arg: Type) do
     // The assignment below generates
     // a compile-time error as 'arg'
     // is a constant object.
     arg.attr := Type
   end
   goo (var arg: Type) do
     arg.attr := Type
     // This assignment is OK, as 'arg' was
     // explicitly marked as mutable.
```

```
    end
    // The immutable attribute should not
    // have a setter. The code below leads to
    // a compile-time error.
    attr2: T1 := (other: T1) do ... end
end // A
```

One more illustration of how **var** works in the context of **ref** and **val** objects.

```
i is 6
```

Type of `i` is deduced by the compiler based on the type of constant object 6 into **val** Integer.

```
ir: ref Integer is 6
```

Here, `ir` has got an explicit type and 6 will be cloned into **ref** `Integer`. No operations that change the contents of the object can be performed over `i` and `ir` – they are immutable. Compile-time errors will be raised for both following statements.

```
ir++
i++
```

The following code compiles and run with no issues. `++` is the routine of unit `Integer`.

```
var j is 5
var jr: ref Integer is 5
j++
jr++
```

So, **ref** and **val** kinds of objects are completely unrelated to the immutability status of objects and both mechanisms give the full control over objects' semantic. Now we have described how to define immutable attributes but how can we properly define constants like numbers, characters, string, and value constants of any type. This leads to the constant objects section.

4. Constant objects

4.1 Backbone – two fundamental constants

Learning computer science usually starts with two simple idioms – 0 and 1 (zero and one). Generalizing we may state that we have two signs circle and bar and start defining everything in the digital world combining these signs into sequences and giving a different interpretation of such chains. Binary digit (bit) was selected as a term to represent this. So, in fact, we have defined some unit Bit which has two constant objects of type `Bit`: `Bit.0b0` and `Bit.0b1`. An example with the part of the source code of unit `Bit` illustrates how these constants are defined.

```
val unit Bit
  const 0b0, 0b1
    // As unit Bit has no init procedure,
    // 0b0 and 0b1 are just two different
    // objects, and 0b0 and 0b1 are their
    // names and values at the same time.
  end

  // Function & is fully defined in the
  // source code of the unit. Both names
  // & and 'and' can be used.
  pure & alias and (other: as this): as this
       => if this = 0b0 do 0b0
          elsif other = 0b0 do 0b0 else 0b1
end // Bit
```

All other types are based on unit `Bit`. And has explicit source code with the implementation of all routines (methods or member-functions). No more need to remember what functions can be applied to 'int' – there is a source code and as we inspect any other type basic types can be inspected too.

4.2 Basic units – basic types

Using the same approach all basic types are being introduced. As one more example, we will use some fragments of units `Integer` and `Integer[BitsNumber:Integer]`. It illustrates one more concept of unit names overloading which works well within our type system.

```
val unit Integer
   extend Integer[Platform.IntegerBitsCount]
   ...
end
```

That is a general `Integer` which uses the platform description constant, the number of bits in integer for setup

```
val unit Integer[BitsNumber: Integer]

  // Thus, we can instantiate this type like
  // Integer[4] or Integer [16] when we need
  // particular types of a particular size
  // in bits
  minInteger is - (2 ^ (BitsNumber-1))
  maxInteger is 2 ^ (BitsNumber-1)-1
  // This is an ordered set defined as a
  // range of all integer constant values
  // (objects)

  const
      minInteger..maxInteger
  end
  ...
  init do
     data := new Bit[BitsNumber]
  end
  {} data: Bit[BitsNumber]

end
```

For types like `String` and `Bit[N]` regular expressions are being used to define all possible constants of these types.

4.3 Constant objects – the general case

Every unit may define all known constant objects or specify the rule with help of regular expression how all constants will be generated. Block **`const ... end`** aims to do that.

For example, `Integer.1` is a valid constant object of type Integer.

To skip unit name prefix, apply **`use ... const`** – import all constants into the place where one needs them.

As an example of constants import, we may consider unit Any which resides at the top of all units (like class `Object` in Java)

```
unit Any
  use const Integer, Real, Boolean,
            Character, String,
            Bit[2**Integer.MaxInteger]
```

All constant objects from basic units are imported into `Any` and respectively to all other units allowing usage of these constants without respective unit name prefix.

Below is an example of weekdays which shows that constant objects replace enumeration types.

```
unit WeekDay
  const
    Monday, Tuesday, Wednesday, Thursday,
    Friday, Saturday, Sunday
  end
end
```

```
use const WeekDay
```

This imports all constant from unit `WeekDay` into this script code. First, call procedure `foo` with the parameter `Monday`.

```
foo(Monday)
```

Here is the declaration of `foo`. It contains an example of pattern matching inside.

```
foo (day: WeekDay) do
  if day is
    Monday .. Friday:
      StandardIO.print("Go to the office")
    Saturday, Sunday:
      StandardIO.print("Do what you like!")
   end
end
```

The last synthetic example shows the exact meaning of constant objects. Some unit `A` is declared. It defines three constant objects and uses all three initialization procedures for their creation. After the unit code, the small script shows how type `A` can be used.

```
unit A
  const
    a1.init,
    a2.init (new T),
    a3.init (new T1, new T2)
  end
  init do end
  init(arg: T) do end
  init(arg1: T1; arg2: T2) do end
end
x is A.a1
```

Here, `x` is defined as a valid constant object and initialized with the value of the constant object from `A`.

```
var y is A.a2
```

However, the attempt to declare a variable and initialize it with the const object will lead to a compile-time error.

5. Types

As mentioned before, there are 8 kinds of proposed types – unit-based type, anchored type, multi-type, detachable type, tuple type, range type, routine type, and unit type. Every type has an explicit description – type declaration.

5.1 Unit-based types

Unit-based type is the most commonly used kind of type. Every new unit declaration defines a new type. Such unit declaration explicitly defines all attributes and all routines of this unit – fixing the set of operations over objects of this type and size of objects of this type in memory. Units are a more general form of classes and modules. Units may inherit like classes and may be used like modules (provide a single object, supplier of functionality). All examples above used unit-based types.

5.2 Anchored types

Anchored type is the type, which is the same as another entity has. Such kind of types was introduced in Eiffel [3]. It works as an automatic overriding while inheriting and allows not to repeat the exact type name. Example

```
b: as a
```

So, `b` is defined as having type the same as `a` has.

```
x: as this
```

Here, `X` has a type similar to the current unit. In descendants type of `X` automatically changes.

5.3 Multi-types (ADT product)

Multi-type states that objects of this type can be one of the types specified in the type declaration. So, the set of operations which can be applied to such objects is an intersection of operation from all types included in the multi-type declaration. So, it allows producing code, which works with objects of already compiled units with no need for inheritance. In the example below, `C` may be assigned with objects of types `A` or `B`.

```
C: A | B
C := new A
C := new B
C.foo(expression)
```

Both types `A` and `B` must have a routine `foo` with the proper signature for the *`expression`* to be compatible with both signatures. The exact definition of type compatibility will be given later.

5.4 Detachable types

Detachable type in the form of `?UnitBasedType` allows to declare attributes with no initial value and such attributes can be initialized later with objects of `UnitBasedType` or its descendants and dynamic type check has to be applied to deal with such objects (call member-routines or read member-attributes). Example

```
d: ?A
if d is A do
   d.foo
   ?d
end
```

`d` is declared as having no value. So, `d` cannot be used unless its type is checked at runtime. Inside of the do block (then part) of the if statement `d` has the type of `A` till the first assignment to it or detachment `?d`.

5.5 Tuple types (ADT sum)

Tuple type defines a group of entities of potentially different types specified in the type declaration. The number of entities is part of the type declaration. It is possible to name these tuple fields with identifiers for access by name. The example below introduces `e` as a group of values. Its type is a tuple with three types in the specified order and `e` is initialized with tuple value.

```
e: (Integer, Real, String) is
              (5, 6.6, “Hello world!”)
```

Next is the square equation solution, which uses tuple to get the result. Type of object `(x1, x2)` is `(Real, Real)`. Function `SolveSquareEquation` returns a tuple in which has named fields in it. Both ways to call it are presented below.

```
SolveSquareEquation (a, b, c: Real):
      (r1: Real; r2: Real) do ... end
(x1, x2) is SolveSquareEquation(1.0,2.0,3.0)
roots is SolveSquareEquation (3.0, 2.0, 4.0)
x1 is roots.r1 // First root
x2 is roots.r2 // Second root
```

Important comment: array is a kind of tuple when all elements have the same static type. That is another reason why access to array elements uses the syntax similar to access to tuple elements by index.

5.6 Range types

The range type explicitly defines a set of possible values objects of this type may have. There are two kinds of this type presented below.

```
f: 1..6
```

`f` can store Integer values between 1 and 6.

```
g: 1|3|5|7
```

`g` can have odd integer values between 1 and 7. `f` and `g` have different types, so any attempts to assign will lead to compile-time errors. Both assignments below are wrong.

```
f := g
g := f
```

5.7 Routine types

The routine type defines objects which are routines and it means that activation (call or application) of the routine associated with the object can be done later. Routines are treated as first-class citizens. The example below defines procedure `foo`, which can be called with the routine object which has the routine type – a function with 2 arguments of types `Type1` and `Type2` returning objects of type `Type3`. The body of `foo` contains a call to routine passed as an argument.

```
foo(h: rtn (Type1, Type2): Type3) do
  x is h (new Type1, new Type2)
end
```

`foo` can be called providing the inline routine object.

```
foo(rtn (Type1; Type2): Type3 do
              return new Type3 end)
```

5.8 Unit types

The unit type defines objects which define types as first-class citizens. One can declare an attribute of type unit and provide a full description of this unit at some time and then use the name of this attribute as a type for declaration of other entities.

```
Type0 is new unit
                foo do end
                init do end
                var attr: X
              end
```

Attribute `Type0` has a type equal to the unit type deduced by the compiler. This unit type is characterized by members: routine `foo`, initialization procedure, and a mutable attribute `attr`.

```
Type1: unit is unit
                  foo do ... end
                end
```

Attribute `Type1` is defined as having type unit initialized by inline unit declaration. Also, it is possible to specify the unit interface of interest and then dynamically assign conforming types to this variable. The order of unit members is not essential; that is the difference from tuples.

```
Type2: unit
          f1: T1
          f2: T2
          r1(T1,T2)
          r2(T1): T2
          init()
       end is new unit
                r1(T1, T2) do end
                r2(T1): T2 do end
                init() do end
              end
```

Here, the type of `Type2` is limited with some interface specified as unit type. So any type which conforms to the interface can be assigned to `Type2`. The initialization part should not repeat the attributes specified in the type description, but new ones may be added and all routines should get their bodies.

```
Type3: ?unit foo(), init() end
```

`Type3` attribute is not initialized but we know its interface. Now we can use new types for ordinary attributes declarations.

```
a0 is new Type0.init()
a0.foo
a1 is new Type1
a1.foo
a2 is new Type2.init()
a2.r1(new T1, new T2)
a3: Type3 is new Type0.init ()
a3.foo
```

What else can be done with attributes of the unit type? By default, assignment works for them and they can be used for declarations. Of course, conformance rules are to be adjusted for such types. But it is possible to build such a unit type during the program execution like as follows:

```
Type4 is new unit end
Type4.add(rtn foo () do end, var x: Integer)
Type4.add (y: Real; init do end)
```

As we have no exact static information about the nature of `Type4`, we have to dynamically test it. If it has proper init procedure or require routine with necessary signature and then call.

```
if Type4 is unit init () end do
    a4 is new Type4.init ()
    if a4 is unit foo () end do
        a4.foo ()
    end
end
```

Among basic units, there is a special one that defines the unit type. The code of procedure add shows how it is possible to deal with unlimited number of arguments.

```
unit unit
     add (members: ()) do
      while member in members do
      Runtime.addMemberToUnit (this, member)
      end
     end
end
```

One more aspect of such types is using them within the generics approach. Instead of parametrization by a constant of an enumerated type, one can provide an expression. See an example below

```
var v1 is new Array[String, 5]
```

`v1` will be an array of strings with five elements properly initialized by `Array` init procedure.

```
var v2 is new Array[String, 6]
```

`v2` will be an array of strings with six elements properly initialized by the `Array` init procedure.

```
v1 := v2
v2 := v1
```

Both assignments are valid as `v1` and `v2` have the same type `Array[String; N: Integer]`.

```
var v3 is
  new Array[String,StandardIO.readInteger()]
```

The actual type of generic instantiation attribute `v3` will be identified during execution.

```
v1 := v3
v3 := v2
```

However, both assignments are valid as `v1` and `v2` have the same type `Array[String; N: Integer]`. So, type compatibility is very essential.

6. Type compatibility

It is essential to define well when assignments are valid and when overriding is valid while inheriting. The latter is described by the signature conformance while the assignment is driven by the following rule. The type of the expression on the right side of the assignment should either conform to the type of the writable on the left side or have a proper conversion routine in place. So,

type A is compatible with type B if A conforms to B or objects of type A can be converted into the objects of type B. Pictures below will use the legend that every oval denotes a unit and every arrow means 'inherits from' aligned with the direction of the arrow. Rombus-ended edge means inheritance with no conformance (not able to make polymorphic assignments)

6.1 Type conformance

The simplest case of conformance is that each type conforms to itself.

```
a: A is new A
```

Unit conformance is based on the idea to check if there is a path in the inheritance graph between the current unit type and another one. And this path should consist only of conformant inheritance edges.

```
unit A end
unit B extends A end
```

That is a conformant inheritance.

```
unit C extend ~A end
```

That is a non-conformant inheritance.

```
a: A is new B
```

Valid as `B` conforms to `A`.

```
a: A is new C
```

Not valid as `C` does not conform to `A`.

When a type is a generic instantiation then in addition to unit type conformance it is necessary to take into account type by type conformance of all elements of the instantiation. Notice that square brackets are used to highlight generics. Access to tuples and arrays is done using parentheses as these are function calls with parameters.

```
unit A[U, V] end
unit B[X, Y] extend A [X, Y] end
unit T1 end
unit T2 end
unit S1 extend T1 end
unit A[A, B, C] end
a: A[T1, T2] is new A [T1, T2]
```

Valid as types are identical.

```
a: A[T1, T2] is new A [S1, T2]
```

Valid as `S1` conforms to `T1`.

```
a: A[T1, T2] is new A [T1, S1]
```

Not valid as `S1` does not conform to `T2`.

```
a: A[T1, T2] is new B [T1, T2]
```

Valid as `B` conforms to `A` and has identical instantiation types.

```
a: A[T1, T2] is new B [S1, T2]
```

Valid as `B` conforms to `A` and has conformant instantiation types.

```
a: A[T1, T2] is new B [T1, S1]
```

Not valid as `S1` does not conform to `T2`.

```
a: A[T1, T2] is new A [T1, T2, S1]
```

Not valid as `A` with 3 generic parameters does not conform to `A` with 2 generic parameters.

Tuple conformance. All tuples are of the same type – tuple type. It means that we need to consider (similar to generic instantiations) by-element conformance of element types.

```
a: (T1, T2) is (new T1, new T2)
```

Valid as types are identical.

```
a: (T1, T2) is (new S1, new T2)
```

Valid as `S1` conforms to `T1`.

```
a: (T1, T2) is (new T1, new S1)
```

Not valid as `S1` does not conform to `T2`.

```
a: (T1, T2) is (new S1, new T2, new S1)
```

Valid as all elements of the longer tuple, which has corresponding elements in the shorter one, conform to them.

Last but not the least is **unit type conformance**. All unit types are of the same type – 'unit', similar to tuple conformance. So, we need to look at a member after a member to check if they conform to each other. The difference from tuples that tuples have an order of elements in the tuple but unit types not. But every member of the unit type has a name. And search by name identifies the subset of members which will define the conformance. So, if we have two unit types A and B then A conforms to B if for every member of A there is a member with the same name in B and its signature in A conforms to the signature of the corresponding member in B and B has not other members. Common sense logic brings the idea that to an empty unit any unit type will conform. Any 'thinner' unit type will always accommodate in terms of conformance the 'thicker' one. Empty unit means any unit!

```
var A is unit end
var B is unit
  foo (T1, T2): T3
  goo (T3)
  var attr: T1 := (T1)
  // It has a setter with an argument
  // of type T1
end
var C is unit
           foo (S1, T2): T3
           goo (T3)
      end
var D is unit
           foo (S1, T2): T3
           goo (T3)
           var attr: T1 := (S1)
           // it has setter with an argument
           // of type S1
           too (T1, T2, T3)
         end
A := B  // Valid as any type conforms
        // to an empty type
B := C  // Not valid as C lacks member
        // called attr
B:= D   // Valid as all D members fit all
        // B members in terms of conformance
        // and D has extra members – it is
        // thicker than B
```

6.2 Type convertibility

Here, conversion routines are considered as they also play important roles in assignments. There are two types of conversion routines: from-conversions and to-conversions. The first one is a procedure with one parameter and the second one is a function with no arguments. Let's examine the following example with units `A` and `T`.

```
unit A
   := (other: T) do ... end
   // This is a from-conversion procedure,
   // which has some algorithm how to
   // perform a conversion from objects of
   // type T into the objects of the current
   // type A. T is just some empty type.
   := (): T do ... end
   // This is a to-conversion function that
```

```
    //  creates a proper object of type T
    // and works well for assignments too.
    foo (arg: T) do end
    // Procedure 'foo' will be used to show
    // how converters work
end

unit T end
```

At first, let's create a valid object of type A, and then different conversions will be done using an assignment statement.

```
var a is new A
a := new T
```

Here, a can be assigned with an object of type T as it has a from-converter procedure.

```
a.foo (new A)
```

This call is valid as well as unit A has a to-conversion function to type T.

Here is a brief review of routines' signature conformance which also has similarity with generic instantiation conformance and uses tuple conformance. If we have routine foo with signature S1 and routine goo with signature S2 then S2 conforms to S1 if they have the same number of elements and every type element of signature S1 conforms to the appropriate element of signature S1. Let's consider the following example

```
unit A
    foo (T1; T2; T3): T4
end
unit B extend A
    override foo (U1; U2; U3): U4
end
```

In this example, the signature of foo from A is ((T1, T2, T3), T4), and foo from B has ((U1, U2, U3), U4) and the task is equal to tuple conformance. Tuple ((U1, U2, U3), U4) conforms to the tuple ((T1, T2, T3): T4) as they have the same number of elements – two in this case (for the procedure we may just drop the return type) and for the first element we again have tuples conformance case – whether (U1, U2, U3) conforms to (T1, T2, T3) and check if U4 conforms to T4.

Some notes about the name and structural type equivalence. Below is an example in Ada [2], which presents name equivalence – type Integer_1 is not compatible with type Integer_2 as they have different names! But structurally they are identical.

```
type Integer_1 is range 1 .. 10;
type Integer_2 is range 1 .. 10;
A : Integer 1 := 8;
B : Integer 2:= A; -- illegal!
```

We can choose between two different approaches. The first one is right below

```
a : 1 .. 10 is 8
b : 1 .. 10 is a
```

Here, a and b have the same type: range type 1..10 and a can be assigned to b.

In the second case when one likes to introduce new types, type Integer_1 is different from Integer_2 and they are not compatible.

```
unit Integer_1 extend Integer
require
    this in 1 .. 10
end
unit Integer_2 extend Integer
require
    this in 1 .. 10
end
var a is new Integer_1
```

```
var b: Integer_2 is a
```

Declaration of `b` leads to compile-time error as the type of `a` is not compatible with the type of `b`.

So, support of name equivalence is in place but the term name is treated a bit wider. `1..10` is treated as the type name, `A | B` is the type name too, and `(T1, T2, T3)` is also a type and its name is a tuple `(T1, T2, T3)`, type "**`as this`**" is compatible to the type of the unit where an attribute of such type was declared.

7. Duck typing

The popular thing is duck typing. It also can be interpreted in terms of the conformance test. As an ability to fly means that we can imagine a hypothetical unit `Flyable` with one abstract procedure `fly` and check if the object of interest conforms to this unit-based type or not. The trick is that we do not need to enforce to change the inheritance graph for that. We need just to construct such a unit on the fly, keep it anonymous, and just apply the proper check. Let's consider the following example which is used for other programming languages

```
unit Duck // It can fly
    fly do
      StandardIO.print("Duck is flying")
    end
end
unit Sparrow // It flies too
    fly do
      StandardIO.print("Sparrow is flying")
    end
end
unit Whale // It does not fly but swims
    swim do
      StandardIO.print("Whale is swimming")
    end
end
while animal in (Duck, Sparrow, Whale) do
    // Now it is necessary to check if the
    // object 'animal' conforms to the type
    // which is described as the anonymous
    // unit-based type which has only one
    // routine – fly with no arguments.
    // Routines are specified using their
    // signature only.
  if animal is unit fly () end
  do
    animal.fly
  end
end
```

Here are a few caveats. What is the static type of animal to be determined by the type inference process? If units `Duck`, `Sparrow`, and `Whale` have the nearest common ancestor, this unit will be the type of animal. If such unit was not explicitly mentioned thru extend directives, then Any will be such unit. So, the process terminates in any case. If there are several nearest common ancestors, then the process can be run for them recursively.

8.Conclusion

The paper presents the unified type system which supports the convergence of different models of programming, allows to have static typing with type inference, to have all types and values to be explicitly and fully defined using the same programming language. For that, the concept of the unit is used and it is defined as a combination of class and module concepts. Types compatibility if fully and explicitly defined using type conformance and type conversion. Both conformance and conversions are fully defined too. The approach which allows treating manifest constants as immutable objects of the proper type is introduced, it works well for basic types and user-defined

ones. It supersedes enumerations and sets the background to have the programming language which is fully defined using the language itself.

References / Список литературы

[1] Clemens A. Szyperski. Import is Not Inheritance. Why We Need Both: Modules and Classes. Lecture Notes in Computer Science, vol. 615, 2006, pp. 19-32.
[2] International Standard: ISO/IEC 8652:2012 Information technology – Programming Languages – Ada.
[3] Bertrand Meyer. Object-Oriented Software Construction, Second Edition. Pearson College Div., 2000, 1296 p.
[4] International Standard: ISO/IEC 10514-2:1998 Information technology – Programming Languages – Modula-2.

Information about authors / Информация об авторах

Алексей Валерьевич КАНАТОВ – инженер-исследователь, магистр, главный академический консультант в компании Хуавей с 2019 года. Сфера научных интересов: проектирование и разработка языков программирования, методики и схемы реализации объектно-ориентированного программирования при наличии множественного наследования и различных форм перекрытия имен.

Alexey Valer'evich KANATOV – engineer-researcher, Magister in computer science, chief academic consultant at Huawei since 2019. Research interests: design and implementation of programming languages, methods and approaches for implementation of object-oriented paradigm in the presence of multiple-inheritance and different forms of overloading.

Евгений Александрович ЗУЕВ – профессор университета Иннополис с 2015 года, зав. лабораторией языков программирования и компиляторов. Основная сфера профессиональных интересов – семантика языков программирования, реализация компиляторов ЯП.

Eugene ZOUEV is a professor in the Innopolis University, Russia, and the head of the laboratory for programming languages and compilers. The area of his professional interests is programming languages' semantics and compiler development.

DOI: 10.15514/ISPRAS-2022-34(3)-3

Diff tool for comparing .NET assemblies in the Rider IDE

V.I. Miroshnikov, ORCID: 0000-0002-6218-9406 <vladislaw.miroshnikov@gmail.com>
St. Petersburg State University,
7/9, University Embankment, Saint Petersburg, 199034, Russia

Abstract. A .NET developer occasionally needs to compare compiled programs or assemblies, e.g., when updating versions of third-party libraries or when working with their own binary files. However, the existing tools have some significant drawbacks, for example they don't support comparison of .NET Core assemblies. In this paper we reviewed different types of .NET assemblies and, taking into account their structure, developed and integrated into Rider IDE our own Assembly Diff tool which considers the disadvantages of the existing tools and expands the comparison possibilities. We presented several variants of comparison tool presentation and implementation and chose the most functional one in the form of a comparison tree, for which we developed and described special algorithms allowing to take into account semantic features of .NET types.

Keywords: Assembly Difference; assembly diff tool; .NET assembly diff; Rider; assembly comparison; Exe/Dll diff; comparing compiled assemblies

For citation: Miroshnikov V.I. Diff tool for comparing .NET assemblies in the Rider IDE. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 31-46. DOI: 10.15514/ISPRAS–2022–34(3)–3

Инструмент для сравнения .NET сборок в интегрированной среде разработки Rider

В.И. Мирошников, ORCID: 0000-0002-6218-9406 <vladislaw.miroshnikov@gmail.com>
Санкт-Петербургский государственный университет,
199034, Россия, г. Санкт-Петербург, пр. Университетский, 7/9

Аннотация. Разработчику .NET иногда требуется сравнить скомпилированные программы или сборки, например, при обновлении версий сторонних библиотек или при работе с собственными бинарными файлами. Однако существующие инструменты имеют ряд серьёзных недостатков, например, они не поддерживают сравнение сборок .NET Core. В данной работе мы рассмотрели различные типы сборок .NET и, учитывая их структуру, разработали и интегрировали в Rider IDE собственный инструмент Assembly Diff, который учитывает недостатки существующих инструментов и расширяет возможности сравнения. Мы представили несколько вариантов представления и реализации инструмента сравнения и выбрали наиболее функциональный в виде дерева сравнения, для которого разработали и описали специальные алгоритмы, позволяющие учитывать семантические особенности типов .NET.

Ключевые слова: Assembly Difference; assembly diff tool; .NET assembly diff; Rider; сравнение сборок; Exe/Dll diff; сравнение откомпилированных сборок

Для цитирования: Мирошников В.И. Инструмент для сравнения .NET сборок в интегрированной среде разработки Rider. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 31-46. DOI: 10.15514/ISPRAS–2022–34(3)–3

1. Introduction

The .NET [1] platform – is Microsoft's software platform for developing various applications. One of the distinguishing features of this platform is the ability to develop applications using several programming languages, e.g., C#, F#, Visual Basic .NET, thanks to Common Language Runtime

(CLR). Currently, there are two different platforms from Microsoft: *.NET Framework* and .NET Core/.NET. .NET Framework is an older version and supports only the Windows operating system, which differs it from .NET Core/.NET that is already cross-platform and corresponds to modern industry standards. Thus, according to Stack Overflow research [2], for 2021 .NET Core/.NET and .NET Framework are among the top three most popular frameworks.

One of the basic, structural and functional units of the .NET platform is assembly, which is used for version control, application deployment and configuration. **.NET assembly** – is a collection of .NET types (classes, interfaces, etc.) and resources (JSON, XML files, etc.) assembled to work together to form a logically functional unit. During the .NET development process, it is often necessary to compare the versions of the compiled programs. For example, sometimes this need arises when updating third-party libraries or debugging problems with dependency resolution of .NET applications, and sometimes – it can be our own binary files, for which we want to understand which version of code corresponds to any of the previously published binary files.

However, there are various ways to compare .NET assemblies. For example, a developer can use any decompiler to get the source code and then apply a text comparison tool. Nevertheless, such solutions are not very convenient because they don't allow to compare assemblies directly without additional tools. There are also "complete applications" providing comparison of assemblies, but they have own significant drawbacks, for example, lack of support for comparing .NET Core/.NET assemblies.

One of the existing development environments for the .NET platform is **Rider** [3] – JetBrains cross-platform development environment, which is one of the "most loved" [4] IDEs according to the mentioned above Stack Overflow research of 2021. In addition to the existing functionality in Rider, this paper proposes extending the capabilities to work with the .NET platform, namely – adding a tool for comparing compiled .NET assemblies. This should take into account the weaknesses of the existing solutions and improve the comparison capabilities. In this paper we consider various ways of comparing assemblies and also offer the implementation of one of them in the Rider IDE. We also need to consider the technical challenges of the task, which is not only about searching for source code differences provided in the C# language. For example, due to the large number of structural units and language features in .NET assemblies it is necessary to search and display "semantic" differences, both in the metainformation of the assembly and in the resources and code, which will be discussed in more detail in this paper.

This paper is organized as follows. Section 2 includes the different types of .NET assemblies, provides the structure and contents of the assembly, gives an overview of existing solutions and the Rider architecture required for the implementation. Section 3 provides the first version of the implementation based on a directory comparison, and discusses the alternative implementation as a comparison tree. Section 4 describes the final implementation based on the assembly diff tree, and gives various algorithms for constructing the tree and comparing types. Section 5 gives general conclusion of the paper and further plans.

2. Background

Assemblies can be created from one or more source code files. For simplicity, they can be thought of as archives, which have a specific structure and in particular contain source code data. Typically, a compiled assembly is presented on disk as a file with the extension .exe or .dll.

2.1 Types of .NET assemblies

According to Microsoft's official .NET [5] documentation, assemblies are divided into two types:

- Executable assemblies – assemblies that are represented as an executable file with the .exe extension. For the sake of brevity, this type of assembly is called an "Exe-assembly";
- Assemblies that are presented as a dynamic link library file – files with the extension .dll.

Hereafter, for the sake of brevity, this type of assembly is called a “Dll-assembly”.

There is also a division of assemblies into single-file and multi-file assemblies. A single-file assembly is the simplest type of assembly, with all its contents placed inside a single *.exe or *.dll file. Multi-file assembly, on the other hand, consists of a set of .NET modules, which are deployed as a single logical unit and are provided with single version number. Such assemblies can be created, for example, using command line compilers. One of the main advantages of this type of assemblies is that they allow combining modules written in different .NET compatible programming languages. A multi-file assembly can also be represented as a file with the extension *.exe or *.dll.

.NET assemblies also satisfy with the ECMA-335 standard [6]. This standard is a specification of common language infrastructure and .NET platform, defining architecture of .NET runtime system, arrangement of different libraries, structure, and types of assemblies, description of intermediate language CIL and others.

Let's take a closer look at Exe and Dll assemblies, as well as their various subtypes.

a) Exe-assembly

   According to ECMA-335 standard [6], the distinguishing features of Exe-assemblies are follows:

   - Possibility of “direct” invocation – the user can directly run a .NET application with the .exe extension;
   - Availability of its own address space and memory area – since the Exe-assembly is executable, it can be run as a separate operating system process with its own address space and memory area.

     As described in Section 1, there are two main .NET platforms: .NET Framework and .NET Core/.NET. Each of these platforms provides a different implementation of Exe-assemblies:

     1) .NET Framework

        In this platform there is a single and so called “classic” type of Exe-assembly. As a result of compilation, we get one file with the extension .exe, which remains to be run by the user. The limitation in the form of dependencies on other assemblies should be taken into account.

     2) .NET Core/.NET

        - Native host assembly – compilation results in both a file with extension .exe and a dynamic library file with extension .dll. There is a restriction though: an Exe-application cannot be started without a corresponding dynamic library file, because the DLL containing the application source code is loaded during the startup.
        - Single File assembly (or so-called standalone assembly) – an assembly presented as a single file, allows to combine all files that application depends on, resources, other assemblies into a single assembly. This type of assembly makes it much easier to deploy and distribute the application, but the size of such a file will be large, as it will include the runtime and platform libraries.

b) Dll-assembly

   In accordance with the ECMA-335 standard mentioned above [6], the distinguishing feature of this type of assembly is that it cannot be “directly” launched, so the DLL-assembly has no address space or memory area of its own. The assembly is dynamic in the sense that it can only be loaded or mounted in some other process, such as a console or web application. A Dll-assembly is also called a class library, because it actually contains the source code, but has no “entry point” of its own.

   It's worth noting that there is only one type of DLL-assembly in .NET Framework and

in .NET Core/.NET.

To summarize the overview of the different types of .NET assemblies, we also declare that a comparison of all the types described above should be supported in the diff tool.

2.2 Structure and content of the .NET assembly

According to the official documentation from Microsoft [7], any .NET assembly, both Exe and Dll, consists of the following:

- Assembly manifest – a key component of the assembly, without which the source code cannot be run. It includes name, version, a list of all assembly files, a list of references to other assemblies, and a list of references to types used by the assembly. Manifest allows the system to identify all the files included in an assembly, map references to types, resources, and assemblies to their files, and manage version control.
- Type metadata – used to define the location of types in an application file;
- Application code in the intermediate CIL language. The assembly file does not contain source code in C# or any other .NET compatible language, but instead contains IL code (a.k.a. bytecode) – the language of the .NET virtual machine.
- Assembly Resources. These can be various JSON, XML files as well as images and audio files.

Additionally, it is worth noting that there is such a thing as format of .NET assembly [8] – binary file format.

The format is fully defined and standardized in the ECMA-335 specification [6]. All .NET compilers and runtime environments use this format. For example, it defines that the assembly is processor- and operating system-independent.

This format makes the assembly diff tool platform-independent in the sense that there is no need for any checks on which processor or operating system a particular assembly was derived. Thanks to the unified assembly format, it is possible to compare any assemblies, even if one of them was derived, for example, from Windows and the other from macOS.

2.3 Existing solutions

The existing solutions can be divided into two groups:

- Solutions that allow to compare assemblies using several tools. This type of solutions are based on using some .NET decompiler (Ildasm, IlSpy, dotPeek) to get the C# source code and then applying any diff tool for text comparison.
- Complete solutions that allow to compare assemblies directly without using any third-party tools. These full-fledged diff tools provide both decompilation and source code difference display. However, many of them have some significant disadvantages. Some, for example, do not support comparison of .NET Core/.NET assemblies, which makes such tools not very relevant these days. Others do not support viewing decompiled C# code and only allow user to see added/removed types.

In our assembly diff tool, we take many of these disadvantages into account and offer solutions to fix them.

2.4 Rider architecture

As stated earlier, Rider [3] – a cross-platform development environment for the .NET platform from JetBrains. To understand the inner workings and architecture of the Rider environment, we refer to the corresponding article [9] published in the journal CODE Magazine.

Thus, the Rider consists of two main operating system processes running in parallel:

- Frontend process – responsible for rendering the user interface, based on the IntelliJ platform [10] and runs on a Java Virtual Machine.

- Backend process – a process without user interface, responsible for code analysis, code inspections, formatting and other logic. Based on the ReSharper [11], which is an extension to the Visual Studio development environment from JetBrains and runs on a .NET virtual machine.

One of the key points of the Rider IDE is that the two processes "communicate". For this purpose, a specially developed "Reactive Distributed" protocol [12], presented as an open-source library, is used. This protocol ensures the principle of reactivity: it provides entities that can react to changes through an event subscription mechanism. It also provides consistency in the state of the system at any given time, and importantly, it allows to work with distributed systems. This protocol enables the exchange of information between two processes.

The Rider architecture is shown schematically in Fig. 1.

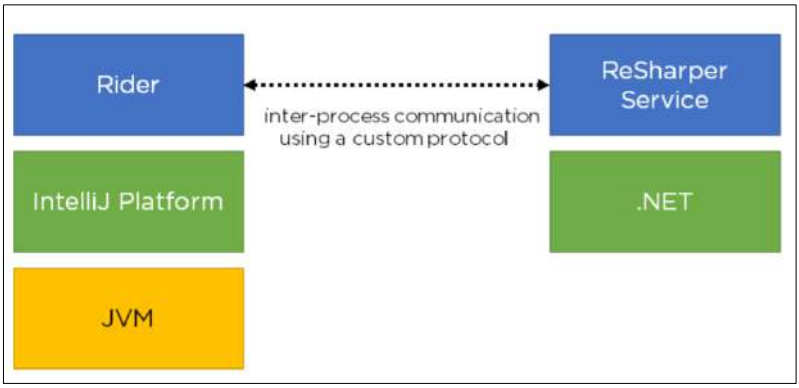


Fig. 1. Architecture of the Rider (from the related article [9])

3. Assembly diff methodology

3.1 Project decomposition

Assembly diff is decomposed into three key subtasks:

- "Disassemble" the assembly and find objects that are different and contain source code. This subtask is due to the fact that an assembly can consist of several source files and has a certain structure described in Section 2.2. Therefore, among various metadata tables, assembly resources and other contents it is necessary to find source code objects.
- Decompile found objects to C# code, as they will be represented in intermediate IL code in the assembly.
- Calculate and display the difference represented as C# code.

Thus, taking the Rider architecture into account, the following decision has been made:

- "Reading" the assembly and finding source code objects, as well as the decompilation process into C# code should be performed on the backend process and implemented in C# using the ReSharper capabilities.
- Difference calculation and display should be done on the frontend process and implemented in Kotlin/Java using the capabilities of the IntelliJ platform.
- The exchange of information between the two processes must be provided by the protocol described in Section 2.4.

3.2 Implementation based on directory comparison of the IntelliJ platform

The IntelliJ platform already includes the ability to compare arbitrary directories and archives. In the IntelliJ IDEA IDE, it is also possible to compare JAR files, which are a kind of analogue of .NET

assembly. JAR files are archives that contain source code, manifest files, and various resources such as audio files. The JAR file diff tool shows the difference between the decompiled versions of Java classes within them.

Based on the existing capabilities of the IntelliJ platform, we decided to make the first version of the .NET assembly diff tool, presented in Fig. 2. There is a table for the user that contains the .NET types – classes, interfaces, enumerations, etc. When you click on a row, you can see the difference of the decompiled C# code.

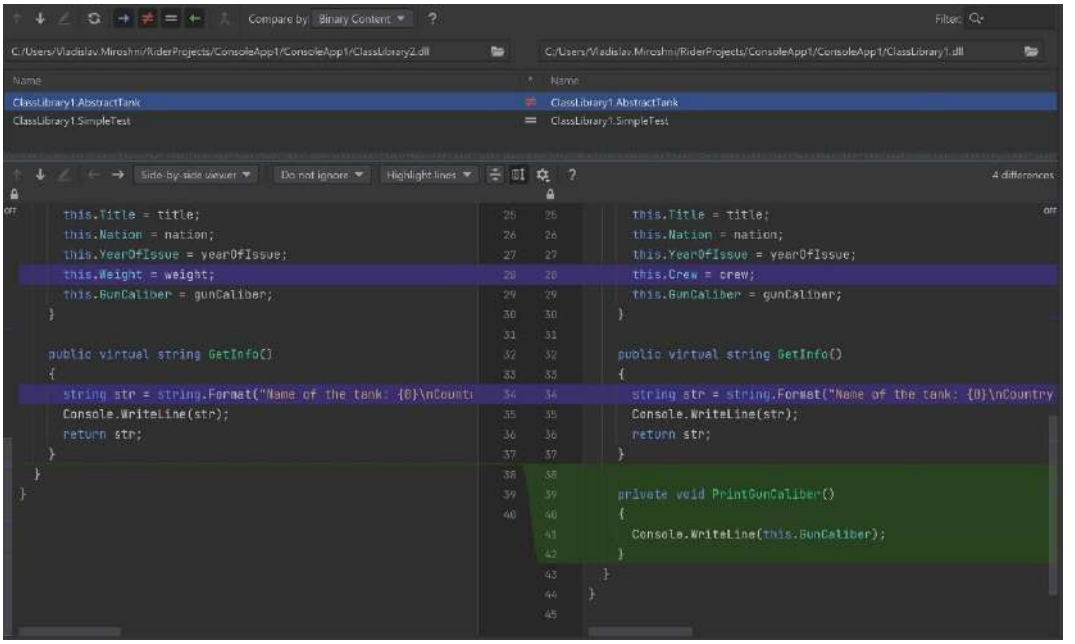


Fig. 2. Assembly diff tool based on directory comparison

However, the implementation of the diff tool is not limited to the one method presented above. We suppose it's appropriate to consider an alternative option. To do this, Assembly Explorer Rider subsystem should be reviewed.

3.3 Assembly Explorer subsystem

Assembly Explorer – an existing subsystem of the Rider IDE, which provides a wide range of opportunities to explore the contents of assemblies. An assembly is represented as a tree with assembly name, version, and platform in the root node, e.g., .NET Framework v.4.8 or .NET Core v5.0.

In the tree itself, the user can see the following information when the root node is expanded:

- Metadata – this subtree contains all metadata information, various metadata tables, special type tokens and more;
- References – this subtree contains information about the dependencies of the assembly, i.e., which other assemblies, packages, and modules the current assembly refers to;
- Resources – this subtree contains various resources, e.g., audio files, images, XML documents. By double-clicking on a resource node, you can view its contents.
- Namespaces and nested types – this subtree is responsible for the source code. You can view classes, its internal parts, such as methods, fields, constructors and other. Double-clicking on a node decompiles the source code into C#, opens in a new window and navigates through the document.

An example of an assembly tree is shown in Fig. 3.

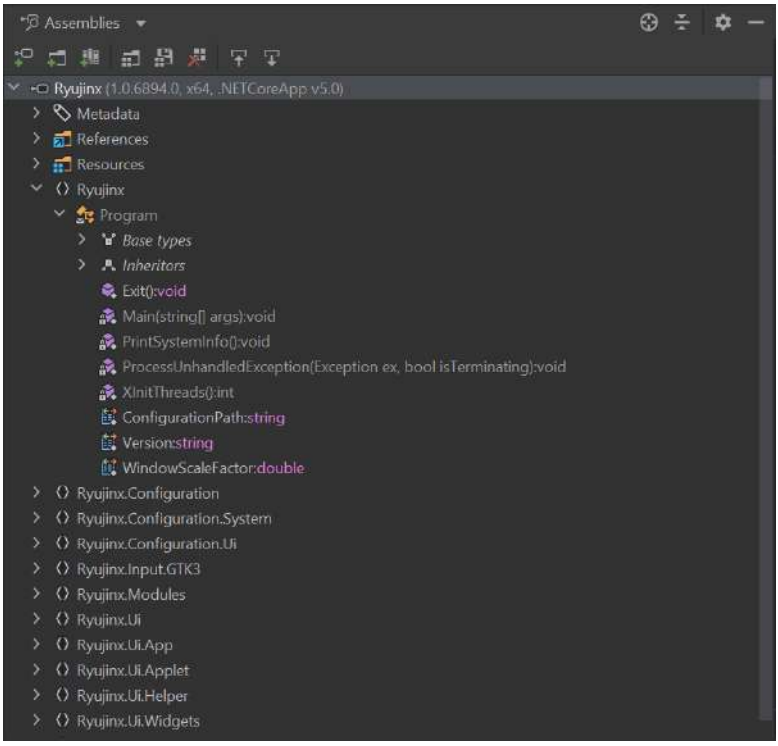


Fig. 3. Assembly tree in Assembly Explorer

It is also worth noting that this subsystem is implemented in other products of the JetBrains .NET ecosystem – ReSharper and dotPeek.

Considering the capabilities of the Assembly Explorer subsystem, we made the following key decision – to use a tree view in the diff tool implementation for the following reasons:

- The assembly diff tree view greatly extends the functionality of the tool in contrast to the version based on directory comparison. Such a tree view allows to compare not only source code differences, but also resources and references from other assemblies.
- Comparison of assemblies in the tree view will also allow to extend the scope of application of this tool. Integration with Assembly Explorer subsystem in future will allow to add assembly comparison functionality in other products – ReSharper and dotPeek. For this reason, choosing a method based on directory comparison will significantly reduce the scope of this functionality and possible work scenarios, as it will only allow comparing assemblies in Rider.

4. Implementation based on assembly diff tree

4.1 General operating principle

By choosing to represent the difference of the assemblies as a tree, we have implemented this approach.

Assembly diff tool works as follows:

1) When a user requests to compare two assemblies on the frontend, the frontend assembly tree is initialized, as well as various additional components. At the same time, an asynchronous request is sent to the backend to “read” assemblies and the “internal backend” tree is initialized, which

is created separately for each request. When parsing assemblies, among the various meta-information and other contents, objects are searched and divided into 3 subtrees: a subtree of references, resources and assembly types. This involves matching appropriate pairs to merge into a single tree node (e.g., the same method with a changed implementation from one assembly version to another), and also applies various algorithms, such as semantic .NET type comparison, which will be described in Section 4.3. The data found is sent to the frontend process using the protocol described in section 2.4 and displayed as a final assembly diff tree. It is worth noting that it is thanks to the reactive protocol that data consistency is ensured at any time between the frontend and backend trees.

2) By double-clicking on the tree node responsible for the .NET type (class, field, method, etc.), a request is sent to the backend, the desired backend tree node is found and then the IL code is translated into C# using the decompiler and the decompiled code is forwarded back to the frontend. Navigation is also provided: the frontend also receives the required parameter to which the carriage should be moved in the corresponding window.
3) The frontend calculates the C# code difference using Myers and LCS [13, 14] algorithms, shifts the carriage, and displays it to the user with the C# syntax highlighting.
4) A specially created protocol model in Kotlin DSL is used to “communicate” between processes. This model describes containers/protocol types that contain some data (such as decompiled class code or assembly paths). These protocol types are then serialized and deserialized when the two processes “communicate”.

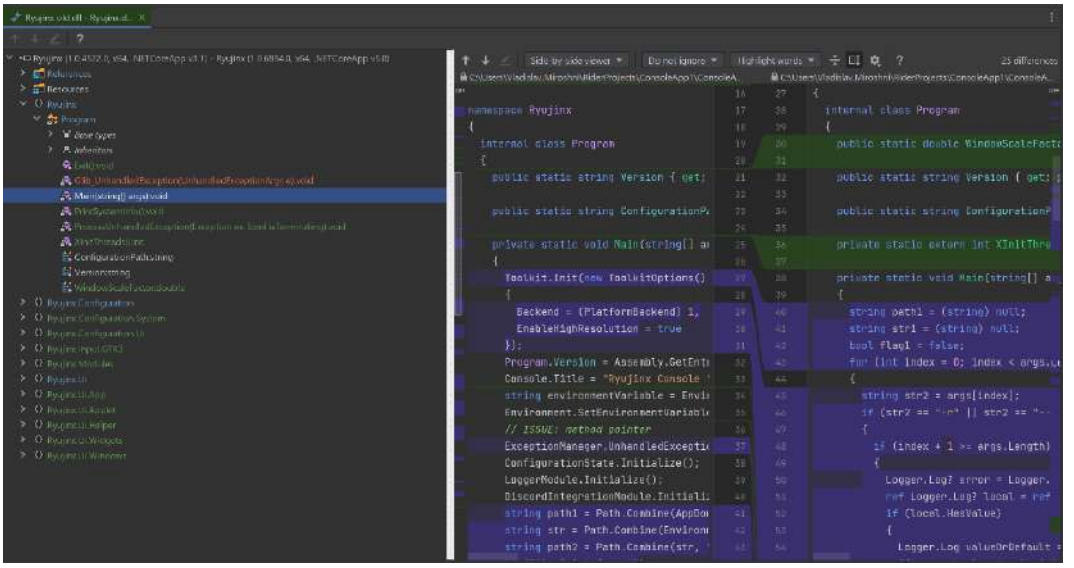


Fig. 4. Assembly diff tool work example

Additionally, it's worth mentioning that this diff tool supports comparison of both .NET Framework and .NET Core/.NET assemblies, including Exe and Dll assemblies and their subsets described in Section 2.1. This fact distinguishes this tool from existing solutions.

An example of the assembly diff tool is shown in the Fig. 4.

Consider the structure and principle of operation of individual subtrees.

4.2 Algorithm of assembly diff tree construction

The assembly diff tree is a key component in the diff tool, it allows to view difference in resources, references and .NET types themselves, as well as C# code differences.

On the backend, a separate *AssemblyDiffTreeHost* instance is created when an appropriate request is received. This instance is responsible for building and updating the tree.

The general algorithm for constructing a tree is as follows:

1) Once the pair of assembly paths is obtained, the assemblies are loaded and a root tree node is created, containing information about the two assemblies, differences in name, versions, platforms, and architecture on which the assembly was obtained.
2) After the assemblies are loaded among the various meta-information and other content, so-called "entry points" are searched and a level 1 of the tree is created from the root node of level 0. Level 1 of the tree contains a references node, a resource node and nodes representing containers for .NET types – in this case namespace nodes. The tree is thus divided into 3 subtrees, which are processed and built independently in asynchronous mode.
3) The following is true for each level of the tree: globally there are three groups of entities: nodes, providers and presenters. Each level of the tree is built as follows: for each node the corresponding provider is triggered, which builds a subtree of the next level, calculating children and creating the necessary instances of nodes. It is possible to say that provider of n-1 level collects nodes of n level of the tree. For example, when you visit node *ClassDiffNode* the corresponding provider will be called, which will spawn children, i.e., the next level containing types of the given class: methods, fields, events, nested classes and so on. Thus, an assembly diff tree is built by applying a breadth-first search (BFS) to traverse the tree in order of levels. At each level of the tree it is necessary to search and match the entity of the old assembly with the new one and merge it into one tree node (in case there is no pairing, the entity is considered as deleted or added). For each of the resource, references and types subtrees, its own matching algorithms are applied.
4) Presenters are used to compose the presentable item of the node: generating text, setting styles and colours, and selecting a suitable icon. The node data is then wrapped in a special protocol container and sent to the frontend to be unpacked and displayed to the user.

Consider the operation of the individual subtrees:

a) References subtree (Fig. 5)

   This subtree contains nodes of two types: *AssemblyReference* (a dependency on another assembly, e.g., *System.Runtime*) and *ModuleReference* (these can be references to some third-party library). All references of old and new assemblies are merged and further grouped by name without regard to version or additional tokens. In this way, each found old-new reference pair will be merged into one node and for it the version difference will be calculated to show to the user. If there is no pair for any reference, it is considered as added or deleted.

b) Resources subtree (Fig. 6)

   This subtree contains various resource nodes: images, XML, JSON files, etc. Similar to the references subtree, the resources are merged and grouped by name, then pairs are searched and merged. In this way the user will be able to identify which resource files have been added or removed from one version to another.

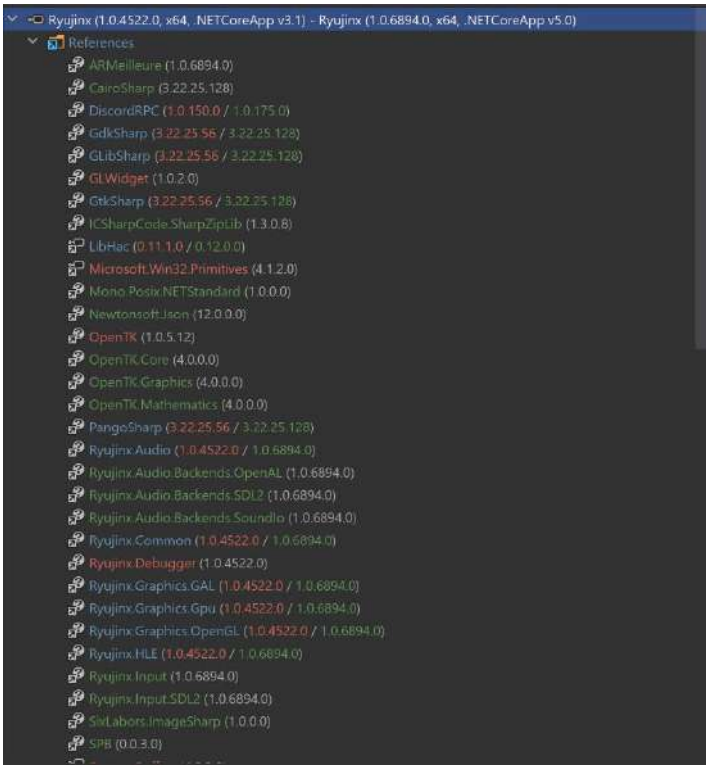


Fig. 5. Example of Reference subtree

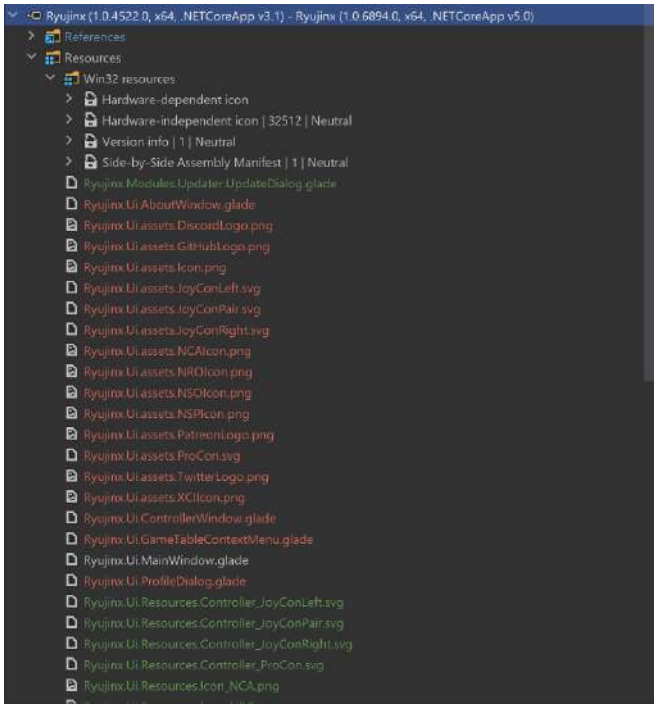


Fig. 6. Example of Resources subtree

c) Assembly types subtree (Fig. 7)

This subtree contains namespaces and nested types: classes, interfaces, enums, etc. with their own nested types.

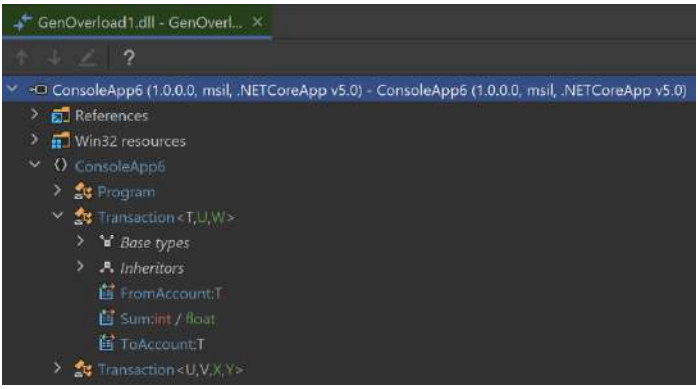


Fig. 7. Example of Assembly types subtree (including signature diff)

When constructing this subtree, the logic and general approach changes somewhat. The general algorithm is as follows:

1) The first stage is to search for namespaces from both assemblies and group them by the unique CLR *QualifiedName*, noting that no two namespaces with the same FQN can be in the same assembly. Thus, we will have two groups: the first will represent namespaces with no pairing, i.e., deleted or added namespaces. The second group will have old-new namespace pairs, meaning that we have found the right pair to merge into a single node in the tree. To determine the state of such a node (modified/unmodified) our Quick Check method first compares the number of nested types for a namespace pair. For example, if the number of classes inside the old and new namespaces does not match, then the node is considered to be changed. If the Quick Check fails, a Full Check is performed: for both namespaces *CLRName* of nested types is collected and compared as sets. There is one extreme case, where the namespaces do not differ in their subtypes and the difference is only at deeper levels (e.g., the method body within the class has changed). In this case, the state of the namespace node will be dynamically updated with a special trigger.
2) At the second stage, for each namespace, subtrees are constructed independently of each other. This level will contain the “children” of namespaces – classes, interfaces, structures, enums and records. It will also group by type *CLRName* and have two groups. The first group will contain unpaired nodes, i.e., nodes meaning that the type was removed or added depending on its location (from the old assembly or from the new one). The second group will have pairs to merge into a single tree node. The modified/unmodified state of the node is defined as follows:
    a. First, we apply a Quick Check for the two types by comparing their signatures. For example, if a class was *public* in the old assembly and *private* in the new assembly, then that node is identified as the changed node.
    b. If the signatures match, a Full Check is performed – a rendering of the IL code is performed for the two types. For example, for two classes we will generate all their IL code and then compare it to determine the state of the node. We believe that in this case the IL code rendering is the key point in determining the node state. We could consider another approach and generate C# code for the classes, but this approach would be slower compared to IL code rendering, which is what we have chosen to achieve the fastest performance in tree construction.

3) When constructing the next levels, we work with "children" of classes, interfaces, etc., i.e. fields, methods, properties, nested classes, etc. The general logic here is partly the same as in point 2: for each type, e.g., class, we collect its old and new nested types from assemblies and also group them by *CLRName*. However, here we have 3 groups:
   a. The first group consists of types for which there are no other types with the same name, i.e., they are treated as deleted or added depending on their location.
   b. The second group consists of pairs, i.e., we have found pairs of types with the same name. In this case the logic changes: first we need to determine if both types belong to some single assembly. If they do, it means that the types don't need to be merged into one tree node, they are two different types and two different nodes. This is possible because of the overloading mechanism in the .NET platform: the types in a given tree level can have the same name. For example, the old assembly had two overloaded methods named *Start* in the *Fabric* class, but the new assembly has no such methods in the same class. In this case, the two methods are considered deleted and treated as different tree nodes. If, however, each of the two types belongs to a different assembly, we have a merge into one node and state detection is performed. A Quick Check – a signature comparison – is also performed first. If this check fails, a Full Check – IL code rendering – is performed. But there are peculiarities here, depending on the .NET type being checked. For example, if we have a property, then in addition to rendering the IL code we need to find and render the IL code of the corresponding *Getter* and *Setter*, because they are located separately from the property in the assembly. Similar behaviour for events, where we need to search for the corresponding *Adder* and *Remover*. A method deserves special attention: if a method is an iterator and *yield return* is used in it or if a method is marked with *async* keyword, a special *StateMachine* class is generated in a compiler-generated class in IL code. In such a case to understand whether the method has changed or not, besides rendering the IL code of the method itself, it is necessary to search among various metadata and *CustomAttributes* and then render this *StateMachine*. The situation is similar when using lambda expressions: in this case, separate compiler-generated methods are created during compilation, and sometimes even entire classes if there is a closure.
   c. The third group contains tuples of 3 or more types with the same name. Here will only be types that can be overloaded in .NET: methods and operators. This group is of the greatest interest, including research interest. Here we immediately face the problem of type matching: suppose we have three overloaded methods *Start* with different signatures in the old assembly, and the new assembly has three overloaded methods *Start* with different signatures in the same class. The question arises: how exactly do we map these methods for further merging into one node? The problem could be solved by using a special token given to each type at compile time. If it were unique and didn't change, we could accurately find the right pairs of methods by this token, but it changes when recompiling. In this case we have to choose some other approach. To do that, we developed a special algorithm for semantic matching and comparison of .NET types, which we will discuss next.

Also at each level, after the providers are completed, the view for the node is computed using the appropriate node presenter. Particularly noteworthy is the generation of text with colour definitions. Thus, for nodes at each level of this subtree (excluding the namespace level as an unnecessary case), a signature difference calculation algorithm is applied. This algorithm computes and identifies parameter addition/removal, parameter type or return value changes, and generic parameter changes. The signature difference will be displayed to the user in the tree and he will not need to look through the decompiled code additionally.

We consider this tree-constructing algorithm to be unique, as it provides wide functionality in exploring differences of assemblies. It is worth noting that existing solutions do not have this kind

of tree-constructing behaviour and use approaches that, in our opinion, are not the most accurate. For example, some existing solutions consider all types with different signatures as different. In this case, assume there is a method in the old assembly and assume that in the new assembly only one parameter has been added to the signature in that method, but the body of the method itself has not changed. Then there will be two different nodes in the tree: the method with the old signature will be considered completely removed, and with the new signature completely added, despite the fact that the implementation of the method has not changed at all. However, in this case we suppose it's more correct to show the user a single node in the tree with the state changed, showing the addition of a parameter to the signature.

4.3 Algorithm of semantic comparison of .NET types

Suppose the following situation. An old assembly has a class with two overloaded methods:

```
class OverloadListing
{
  public void Add(int id, int age)
  {
    // Method body
  }
  // Other methods ...
  public void Add(string name, string
      email)
  {
    // Method body
  }
}
```

The new assembly has the same class with two overloaded methods, but these methods have different signatures:

```
class OverloadListing
{
  public void Add(string name, string[]
      emails)
  {
    // Method body
  }
  // Other methods ...
  public void Add(int id, float age)
  {
    // Method body
  }
}
```

We don't consider methods with different signatures as different and we don't think it's correct to show two deleted methods and two added methods in the tree in this case. In the case of several overloaded methods whose signatures have changed in the new assembly, we believe it is most correct to match methods with the most similar signatures. Thus, for the old method it's necessary to find the closest new method, taking into account the semantic differences.

When implementing such an algorithm, it is most important to propose an appropriate metric to determine the proximity of the signatures. In this example, it is obvious that the signature from the old assembly *public void (int, int)* is closer to the signature of *public void (int, float)* than to *public void (string, string[])*, because in the first case the type of one parameter has changed and in the second two have changed. Our metric works as follows: for the chosen pair of .NET types we create a *similarityCounter* that accumulates data about the proximity of signatures. Then multiple checks are performed: comparison of *CLRName* for two types, generic parameters (if exists), comparison of access modifiers, keywords, return value types, also compared types of passed parameters, their number, name of parameters, presence of additional modifiers *ref/in/out*, performed

CLRTypeConversion – determination of data types affinity (for example, we determine that type *float* is closer to type *double* or *int* than to *string*) and other checks. Each check, if passed, has its own weight by which the *similarityCounter* is incremented. For example, for some keywords there are the following checks:

```
public int
    CompareByMostCommonParams(Element
    first, Element second)
{
    //...
    if (first.IsStatic ==
        second.IsStatic)
        similarityCounter++;

    if (first.IsVirtual ==
        second.IsVirtual)
        similarityCounter++;
    //other keywords checks (override,
        sealed, abstract, etc.)
}
```

Using this metric, we compare all signatures from the first assembly with all signatures from the second assembly and use a greedy algorithm: sort all possible pairs and take the closest pair of signatures, then the closest of the remaining ones, and so on. If we have 3 overloaded methods in the old assembly and 5 in the new one, we'll find only 3 mappings, and 2 methods will be considered as added. It is quite possible that there are pairs with the same proximity value (e.g., there were signatures (*int*) and (*double*) and became (*string*) and (*object*)), then we can choose the matching arbitrarily. Comparing method bodies and rendering IL code is unnecessary in this algorithm, thus reducing the overhead.

This algorithm is in some sense a heuristic – we can define other proximity metrics too. We do not consider our chosen metric to be the only true one and leave the ability for research in this area.

5. Conclusion and Future work

Assembly Diff tool is a tool designed and integrated into the Rider IDE for comparing compiled .NET assemblies and provides extensive functionality for exploring assembly differences. The tool includes not only differences in .NET types but also references and resource differences and presents them in a convenient diff tree. The Assembly Diff tool considers and fixes the shortcomings of existing tools, expands the number of supported .NET assembly types, and uses specially designed algorithms in the comparison tree to compute signature differences and account for semantic features in type mapping. The Assembly Diff tool is used in Rider with a special assembly diff action (including the Ctrl+D shortcut) and is also integrated with the version control subsystem, allowing users to view assembly diffs in the Commit tab.

However, in the future we plan to integrate the Assembly Diff tool into other products: ReSharper and dotPeek, and to integrate with the NuGet Rider subsystem. This would allow users to conveniently explore the differences between the versions of the downloaded third-party libraries.

References

[1]. .NET platform from Microsoft. URL: https://dotnet.microsoft.com/, accessed 19-March-2022.
[2]. Stack Overflow frameworks and platforms survey for 2021 year. URL: https://insights.stackoverflow.com/survey/2021#most-popular-technologies-misc-tech, accessed 19-March-2022.
[3]. JetBrains Rider IDE. URL: https://www.jetbrains.com/ru-ru/rider/, accessed 19-March-2022.
[4]. Stack Overflow most loved collaboration tools survey for 2021 year. URL: https://insights.stackoverflow.com/survey/2021#most-loved-dreaded-and-wanted-new-collab-tools-love-dread, accessed 19-March-2022.

[5]. .NET assemblies' types according to Microsoft documentation. URL: https://docs.microsoft.com/en-us/dotnet/standard/assembly/, accessed 20-March-2022.
[6]. Standard ECMA-335: Common Language Infrastructure (CLI), 6th edition, June 2012. URL: https://www.ecma-international.org/wp-content/uploads/ECMA-335_6th_edition_june_2012.pdf, accessed 20-March-2022.
[7]. .NET Assembly contents according to Microsoft documentation. URL: https://docs.microsoft.com/en-us/dotnet/standard/assembly/contents, accessed 21-March-2022.
[8]. .NET Assembly format according to Microsoft documentation. URL: https://docs.microsoft.com/en-us/dotnet/standard/assembly/file-format, accessed 24-March-2022.
[9]. C. Woodruff and M. Balliauw. Building a .NET IDE with JetBrains Rider. CODE Magazine, 2018, November/December.
[10]. JetBrains IntelliJ Platform. URL: https://www.jetbrains.com/ru-ru/opensource/idea/, accessed 23-March-2022.
[11]. JetBrains ReSharper. URL: https://www.jetbrains.com/ru-ru/resharper/, accessed 23-March-2022.
[12]. Reactive Distributed communication framework. URL: https://github.com/JetBrains/rd, accessed 23-March-2022]
[13]. E.W. Myers. An o(nd) difference algorithm and its variations. Algorithmica, vol. 1, 1986, pp. 251-266.
[14]. J.W. Hunt and M.D. Mcilroy. An algorithm for differential file comparison. Computing Science Technical Report, vol. 41, Bell Laboratories, 1976, 9 p.

Information about the author / Информация об авторе

Vladislav Igorevich MIROSHNIKOV – a student and a researcher at St. Petersburg State University. His research and professional interests include IDE development, code refactoring, power consumption research of Android devices.

Владислав Игоревич МИРОШНИКОВ – студент и исследователь Санкт-Петербургского государственного университета. В сферу научных и профессиональных интересов входит разработка IDE, код-рефакторинг, исследование энергопотребления Android устройств.

DOI: 10.15514/ISPRAS-2022-34(3)-4

Виртуальные площадки в алгоритме излучательности

[1]*А.С. Щербаков, ORCID: 0000-0002-5360-4479 <alex.shcherbakov@graphics.cs.msu.ru>*
[1,2]*В.А. Фролов, ORCID: ORCID: 0000-0001-8829-9884 <vfrolov@graphics.cs.msu.ru>*
[2]*В.А. Галактионов, ORCID: 0000-0001-6460-7539 <vlgal@gin.keldysh.ru>*

[1] *Московский государственный университет имени М.В. Ломоносова,*
119991, Россия, Москва, Ленинские горы, д. 1
[2]*Институт прикладной математики имени М.В. Келдыша РАН (ИПМ РАН),*
125047, Россия, Москва, Миусская площадь, 4

Аннотация. Создание упрощенной геометрии для метода излучательности является трудоёмким процессом, который трудно автоматизировать в общем случае. В качестве альтернативного решения этой проблемы в данной работе предлагается модификация метода излучательности с использованием виртуальных площадок. Виртуальные площадки – это элементы геометрии, полученные кластеризацией некоторых точек исходной геометрии, для которых производится вычисление освещения. Они имеют нормаль, цвет и площадь, но не имеют геометрического представления, представляя собой облако точек внутри вокселя. По сравнению с оригинальным методом излучательности предложенный метод, не снижая производительности вычисления глобального освещения, увеличивает его точность.

Ключевые слова: глобальное освещение; вторичное освещение; излучательность; кластеризация

Для цитирования: Щербаков А.С., Фролов В.А., Галактионов В.А. Виртуальные площадки в алгоритме излучательности. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 47-60. DOI: 10.15514/ISPRAS–2022–34(3)–4

Virtual patches approach for radiosity

[1] *A.S. Shcherbakov, ORCID: 0000-0002-5360-4479 <alex.shcherbakov@graphics.cs.msu.ru>*
[1,2] *V.A. Frolov ORCID: 0000-0001-8829-9884 <vfrolov@graphics.cs.msu.ru>*
[2]*V.A. Galaktionov, ORCID: 0000-0001-6460-7539 <vlgal@gin.keldysh.ru>*

[1] *Lomonosov Moscow State University,*
GSP-1, Leninskie Gory, Moscow, 119991, Russia
[2] *The Keldysh Institute of Applied Mathematics Russian Academy of Sciences,*
Miusskaya sq. 4, Moscow 125047, Russia

Abstract. Geometry simplification for the radiosity method is a laborious process and it is difficult to automate in the general case. As an alternative solution to this problem, this paper proposes a modification of the radiosity method using approximation called “virtual patches”. Virtual patches are elements of the geometry obtained by clustering some points of the original geometry for which the illumination is calculated. They have a normal, color and area, but do not have a geometric representation, representing a cloud of points inside the voxel. In comparison with the original radiosity method, the proposed method, without reducing the performance of calculating global illumination, increases its accuracy.

Keywords: global illumination; indirect illumination; radiosity; clustering

For citation: Sherbakov A.S., Frolov V.A., Galaktionov V.A. Virtual patches approach for radiosity. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 47-60 (in Russian). DOI: 10.15514/ISPRAS–2022–34(3)–4

1. Введение

Метод излучательности [1] для вычисления глобального освещения диффузных поверхностей обладает неплохим балансом скорость/качество по сравнению с другими методами интерактивной глобальной освещённости благодаря пред-просчёту информации о видимости и сохранению энергии, но в то же время требует вычислительных затрат пропорциональных квадрату количества полигонов 3D сцены (или площадок).

Существуют различные подходы, снижающие вычислительную сложность этого метода. Часть из них являются развитием иерархической излучательности [2], другие предлагают альтернативные схемы вычисления освещения [3], замещающие стандартные подходы: решение матричного уравнения [1] и многократное умножение матрицы на вектор.

Несмотря на существование подобных подходов, они не решают представленную проблему в полной мере. Поэтому в существующих решениях создаётся упрощенная геометрия [4], для которой вычисляется освещение. Затем вычисленное освещение накладывается на исходную геометрию для последующей визуализации.

2. Обзор существующих работ

2.1 Глобальное освещение

Задача глобального освещения разделяется на две составляющие: первичное освещение и вторичное. Первичное – это свет, попавший на геометрию 3D-сцены непосредственно из источника света. Вторичное – свет, отраженный от поверхностей. Задача первичного освещения значительно проще с точки зрения количества требуемых вычислений и успешно решается при помощи различных техник карт теней [5] и трассировки лучей [6]. Отдельно выделяют методы, позволяющие создавать мягкие тени [7]. Вторичное освещение намного сложнее вычислить, так как каждая точка геометрии является вторичным источником света. Поэтому для вычисления глобального освещения с учетом вторичного используют различные аппроксимации.

Существует большое число методов глобального освещения. Можно разделить их на два основных вида: методы реального времени и методы, использующие пред-просчёт.

Методы реального времени не требуют подготовки 3D сцены, но они уступают в своей точности методам, использующим пред-просчёт. Наиболее актуальными из них на данный момент являются следующие методы.

Instant Radiosity [8]. Данный алгоритм генерирует набор вторичных источников света, используя информацию об освещенности сцены первичными источниками. При вычислении освещения используется освещение от всех сгенерированных источников. Данный метод прост в реализации, но крайне требователен к количеству вторичных источников света для достижения приемлемого качества освещения. Также при использовании карт теней для каждого вторичного источника требования к памяти становятся неприемлемыми для большинства современных GPU.

Reflective Shadow Map (RSM) [9]. Этот метод является развитием Instant Radiosity. Обычная карта теней дополняется каналами, которые хранят цвет и нормали геометрии, отбрасывающей тень. Эти данные используются для генерации вторичных источников света. Таким образом, при вычислении освещения достаточно сделать несколько выборок из RSM и добавить освещение от вторичных источников. При этом вторичные источники не имеют карт теней и освещение может “просачиваться” сквозь геометрию. Для уменьшения количества артефактов карт теней часто используют Backface culling - отбрасывание

треугольников, повернутых к источнику света. RSM делает такой подход неприменимым. Таким образом, он быстрее, чем Instant radiosity, но имеет ряд недостатков связанных с качеством вычисленного освещения.

Screen Space Directional Occlusion (SSDO) [10] является адаптацией методов SSAO [11], учитывающей вторичное освещение при вычислении Ambient Occlusion (AO). При вычислении AO точки, которые выступают окклюдерами для освещения от окружения, также становятся вторичными источниками света. Учёт света от этих источников даёт некоторое приближение глобального освещения на дистанции, ограниченной радиусом эффекта SSAO. Так как эта техника работает в экранном пространстве, то она может учесть освещение только от геометрии, находящейся в данный момент на экране. При вращении объектов или камеры вторичное освещение от отдельных объектов может появляться и пропадать.

GTAO [12] также работает в экранном пространстве и имеет некоторую аппроксимацию вторичного освещения. Авторы данного метода собрали большую статистику того, как свет переотражается для различной геометрии, и предложили аппроксимацию многократных отражений через многочлены. При этом в своём методе они аппроксимируют только яркость, не учитывая свет окружающей геометрии, поэтому в качестве цвета вторичных источников света берется цвет поверхности, на которую освещение этих источников падает. Сделано это из предположения, что цвет геометрии в небольшой окрестности скорее всего не различается значительно. Как и рассмотренному ранее методу SSDO, данному алгоритму присущи недостатки методов, работающих в экранном пространстве.

Light Propagation Volumes (LPV) [13]. В данном методе предлагается использование воксельной сетки в качестве так называемого объема распространения света. Инициализация этих объёмов производится при помощи алгоритма Reflective Shadow Map. Затем для объёмов производится вычисление переноса освещения между вокселями этого объёма. Данный метод подходит как для статической, так и для динамической геометрии. При этом он имеет следующие недостатки. Во-первых, ошибки вызванные дискретизацией сцены в объёмах распространения света. Отчасти эту проблему можно решить наличием нескольких каскадов объёмов. Во-вторых, невозможность поддержки матовых и зеркальных материалов для распространения света.

Voxel Cone Tracing (VCT) [14]. Суть этого алгоритма в том, что он заменяет дорогостоящую трассировку лучей через множество треугольников сцены трассировкой конусами вокселизированной сцены. При инициализации алгоритм строит воксельное представление сцены вокруг камеры и создаёт несколько MIP уровней для этого представления. Для вокселизированной сцены вычисляется освещение при помощи карты теней. Для вычисления самого освещения из пикселя выполняется трассировка конусами по полусфере. Количество и размер конусов зависит от материала поверхности, для которой вычисляется освещение. В ходе трассировки с удалением от исходной точки выбираются всё менее детальные уровни трехмерной текстуры с освещением. Данный метод более затратен для вычислений, чем подходы, рассмотренные выше, и имеет проблемы с просачиванием света сквозь геометрию или некорректным освещением геометрии в длинных узких коридорах. Обычно на практике этим алгоритмом вычисляют одно переотражение света, так как последующие переотражения требуют больше дорогостоящих вычислений.

Ray Tracing Global Illumination [6]. Это один из самых новых методов для использования в реальном времени, так как он опирается на аппаратную поддержку трассировки лучей. Для точек поверхности производится трассировка лучей в нескольких направлениях и вычисляется освещение методом Монте-Карло. Так как в реальном времени нет возможности трассировать достаточное количество лучей на пиксель для получения стабильного изображения, данный метод применяют совместно с алгоритмами подавления шума [15]. Также для увеличения производительности, трассировку можно проводить для меньшего разрешения экрана, совмещая её с алгоритмами увеличения разрешения [16]. Данный подход не подходит для видеокарт без поддержки трассировки лучей и требует выполнения

дорогостоящих операций, таких как трассировка луча и построение или изменение ускоряющих структур.

Spherical Harmonics (SH) [17] используются для аппроксимации освещения как сферической функции, зависящей от направления. Они могут применяться как полностью решение реального времени, так и как метод, требующий ручной расстановки проб, для которых будут вычислены гармоники. Для создания гармоники требуется отрисовать сцену в кубическую текстуру и для полученной текстуры вычислить приближение освещения гармоникой. На практике используют небольшое количество коэффициентов гармоники для уменьшения количества данных, требуемых для вычислений. Таким образом, гармоники могут представлять только диффузное освещение с ограниченной точностью. Для спекулярных отражений используют MIP-уровни кубической текстуры, полученной ранее. Это требует дополнительной фильтрации при построении MIP-уровней текстуры [18]. При автоматическом расположении проб для вычисления гармоник возможны проблемы с расположением пробы внутри геометрии, что будет приводить к просачиванию освещения через эту геометрию, либо некорректным значениям гармоник. Для использования данного метода необходимо отдельно решать задачу проверки видимости пробы из точек, для которых вычисляется освещение.

Irradiance Probes (IP) [19] может использовать сферические гармоники для представления irradiance внутри проб. При этом использование проб почти всегда заключается в наличии 3D сетки проб, которая используется для вычисления освещения. Для точки геометрии, которая покрывается сеткой проб, находятся 8 проб, которые её освещают, и с использованием трилинейной фильтрации вычисляется освещение для данной точки. Такой подход может давать просачивание света при использовании фильтрации, поэтому активно исследуются методы, уменьшающие такие артефакты.

Irradiance Cache (IC) [20] по сути является обобщением сетки Irradiance Probe. Данная техника предлагает различные структуры для хранения irradiance, как в экранном, так и в мировом пространстве. При этом стоит отметить, что иерархические или анизотропные структуры требуют больше времени для обращения к кэшу, хотя и снижают затраты памяти.

RTX Global Illumination (RTX GI) [21] – это метод, предложенный компанией Nvidia для видеокарт с аппаратной поддержкой трассировки лучей. Он совмещает в себе методы Irradiance Probes и трассировку лучей. Для построения проб используются не кубические текстуры, а атлас текстур с октаэдральной проекцией и атлас текстур с глубиной также с октаэдральной проекцией. Это требует большее количество памяти для диффузной составляющей, но использование атласа глубины позволяет исключить большую часть артефактов, связанных с протечками света сквозь геометрию. Увеличенное количество требуемых данных для каждой пробы приводит к уменьшенному количеству проб и, как следствие, к уменьшенному объему, содержащему пробы при сохранении их плотности. За счёт уменьшения количества артефактов данный метод значительно превосходит другие методы irradiance cache в качестве изображения, но накладывает значительные ограничения на используемое аппаратное обеспечение.

Из методов, требующих пред-просчёта, можно выделить следующие.

Lightmaps [22, 23]. Это наиболее простой из методов глобального освещения. На этапе предобработки сцены вычисляется освещение для всех статических поверхностей 3D-сцены. Для этого можно использовать методы реалистичного рендеринга, такие как трассировка путей. Вычисленное освещение сохраняется в текстуры, из которых затем формируется атлас. При визуализации, для наложения освещения производится выборка рассчитанного освещения из атласа. Данный метод обладает высокой производительностью, но имеет ряд существенных недостатков. Во-первых, освещение возможно использовать только для статической геометрии и для фиксированной конфигурации источников света. Во-вторых, может потребоваться большой объем дополнительной памяти для хранения атласа и

текстурных координат атласа в каждой вершине статической геометрии. Для современных сцен, содержащих миллионы треугольников, данный недостаток делает весь алгоритм неприменимым. В-третьих, таким образом можно вычислить только диффузное освещение, так как может быть вычислено только освещение, не зависящее от позиции камеры.

Spherical Harmonics Precomputed Light Transfer (SH PRT) [24]. Этот алгоритм использует описанные выше сферические гармоники для представления освещения. На этапе предобработки сферические гармоники используются для представления освещения от окружения. Так как на этом этапе известна финальная геометрия 3D объекта и предполагается, что она не будет изменяться в ходе визуализации, при вычислении гармоник можно учесть самозатенение света и переотражения для данного объекта. Данный метод поддерживает как диффузные отражения, так и спекулярные.

Нейросетевые методы [25, 26]. В последнее время ведутся работы по использованию нейронных сетей для вычисления глобального освещения. При этом используется архитектура многослойного перцептрона [25], либо архитектура U-network [26]. В настоящий момент данные подходы имеют слишком высокие требования ко времени выполнения, что делает их ограниченно применимыми на практике. Помимо этого, разработчик имеет ограниченный контроль над работой алгоритма, так как нет прямого метода по исправлению тех или иных артефактов. Для этого нужно расширять обучающую выборку, добавляя в неё примеры найденных артефактов.

Radiosity (излучательность) [1]. Данный метод один из наиболее реалистичных, так как использует физически корректные вычисления с двумя допущениями: все поверхности сцены матовые и их Двулучевая Функция Рассеивания (ДФР, BRDF) описывается моделью Ламберта. На этапе предобработки для сцены вычисляется матрица форм-факторов. Форм-фактор — это число, показывающее какая часть света от пришедшего на заданный полигон отразится на другой. При этом направление света, падающего на полигон, не имеет значения, так как ламбертовские материалы отражают свет равномерно во все стороны. Так как свет может переноситься между любыми двумя полигонами, количество форм-факторов равно N^2 для N полигонов сцены. Эти данные удобно представить в виде матрицы. Для корректного вычисления форм-факторов требуется вычислить интеграл:

$$F_{ij} = \frac{1}{A_i}\int_{x\in P_i}\int_{y\in P_j}\frac{cos\theta cos\theta'}{\pi r^2}V(x,y)dydx.$$

В общем случае, вычислить данный интеграл аналитически нельзя. Поэтому вычисляют его приближение методом Монте-Карло. Данная операция является очень затратной, поэтому её можно проводить только на этапе предобработки.

Отдельной проблемой при вычислениях является наличие больших треугольников. Поскольку при делении такого полигона значения форм-факторов для его частей, и как следствие, их освещение, могут значительно различаться метод излучательности требует некоторого подразбиения исходных полигонов 3D сцены. Следовательно, форм-факторы вычисляются для M полигонов, где M >= N. Так как современные сцены содержат миллионы полигонов, количество форм-факторов будет огромным, и матрицу такого размера нельзя будет эффективно использовать при визуализации реального времени.

Чтобы решить данную проблему, на практике используют иерархическую излучательность, кластеризацию полигонов и упрощенную геометрию. При этом иерархическая излучательность имеет проблемы с производительностью на GPU, так как этот метод требует выполнения несколько этапов вычислений для каждого уровня иерархии и синхронизацию между ними. Кластеризация полигонов позволяет разделить сцену на несколько под-сцен, и каждая из них обрабатывается независимо. При этому может теряться освещение, отраженное с одной под-сцены на другую. При использовании любой из этих оптимизаций требуется использовать упрощение геометрии.

Данный метод может быть применен в реальном времени для вычисления вторичного освещения на статической геометрии. Для динамической геометрии можно вычислять освещение при помощи сферических гармоник [17]. Источники освещения могут менять свою конфигурацию, и освещение будет меняться динамически.

Другое решение проблемы большого количества полигонов заключается в том, чтобы выполнить метод излучательности не на исходном наборе полигонов, а на некотором другом наборе полигонов (или площадок), который был бы существенно меньше, а дискретизация поверхностей была бы более-менее равномерной. После чего результаты перенести обратно на исходную геометрию.

2.2 Упрощение геометрии

Для метода излучательности применяют как ручное, так и автоматизированное упрощение геометрии. Ручное упрощение требует работы со стороны 3D художников и несколько итераций для уменьшения количества артефактов, связанных с упрощением.

Можно выделить несколько видов автоматического упрощения геометрии.

Модификация исходной геометрии [4]. Данная группа методов итеративно изменяет некоторые полигоны исходной сцены до достижения некоторого условия. Например, достижение необходимого числа полигонов или достижение нужных метрик для площадей полигонов. В данных методах выделяют следующие операции: удаление вершин, коллапс ребер, кластеризация вершин, удаление полигонов, изменение ребер, смещение вершин. Такие подходы не применимы для метода излучательности, так как в них нет контроля над нормалями полученных полигонов, что существенно влияет на результирующее освещение.

Ремешинг [27]. Идея этих алгоритмов заключается в том, чтобы создать геометрию аналогичную данной с меньшим количеством полигонов. Для достижения этой цели используется не модификация исходной геометрии, а создание векторных полей, которые затем используются для генерации новой геометрии. Такие подходы применимы для уменьшения сложности 3D объектов, но на низких количествах результирующих полигонов они дают результаты с артефактами, требующие ручной модификации. Чаще всего эти артефакты связаны с появлением отверстий и самопересекающейся геометрии. Таким образом, при использовании с методом излучательности ремешинг может давать просачивание света и ложные тени во вторичном освещении.

Нейросетевые методы [28]. Для аппроксимации геометрии можно использовать самоорганизующиеся карты Кохонена. В функцию, которую оптимизирует нейронная сеть, можно заложить зависимость от нормалей исходной и упрощенной геометрии и равномерность площадей финальных полигонов. Недостатком данного подхода является необходимость вручную задавать начальную геометрию карты для устранения возможных артефактов: исчезновения или появления отверстий, которым нет соответствия в исходных данных 3D сцены.

Также можно представить геометрию в виде набора заранее заданных примитивов. Один из таких методов связан с вокселизацией геометрии [29]. Вся сцена разбивается на воксели и те из них, которые содержат внутри себя геометрию, считаются заполненными. Такой подход будет давай артефакты для объектов и треугольников, которые не ориентированы по осям воксельной сетки. Улучшить ситуацию можно с использованием марширующих кубов [30]. Такой подход позволит увеличить количество вариаций допустимых направлений нормали и форм полигонов, но он не решает данную проблему полностью и в экстремальных случаях возможно грубое представление.

Level of Details (LoD) [31]. Многие инструменты для работы с 3D геометрией имеют возможность генерации уровней детализации для объектов и сцен. Основная задача упрощенной таким образом геометрии - заменять исходные полигоны, когда они находятся

на большом расстоянии от камеры во время визуализации и не требуется наличие большого количества деталей, которые превращаются в наборы субпиксельных треугольников. Такие автоматически сгенерированные объекты не применимы для метода излучательности, так как целью их создания является уменьшение количества треугольников, что приводит к появлению больших полигонов с неравномерным освещением.

Таким образом, существуют различные методы упрощения геометрии, но нет специализированного подхода, который ориентирован на метод излучательности. Так как создание упрощенной геометрии нельзя полностью автоматизировать, в данной работе предлагается альтернативный вариант метода излучательности, который не требует создания низкополигонального аналога сцены или работы художников.

2.3 Surfels (Surface Element)

Для вычисления освещения сложных сцен в высокоточном рендеринге используется метод surfels [32]. Surfel – по сути является вторичным источником света, но не точечным, а с некоторой площадью. Он содержит цвет, позицию, нормаль и площадь. При вычислении освещения проверяется видимость этих источников для требуемых точек сцены и, если нет загораживающих видимость объектов, добавляется освещение от данного surfel. При этом возможны оптимизации, связанные с кластеризацией таких источников и построения их иерархий. Этот подход не применяется для визуализации в реальном времени, но в данной работе были использованы его идеи для создания виртуальных патчей для метода излучательности.

3. Предложенный метод

В данной работе предлагается метод создания и использования виртуальных патчей, которые позволяют избежать использования упрощенной геометрии. Сцена разбивается на воксели. Внутри каждого вокселя генерируются семплы (точечные выборки) на исходной геометрии. Для каждой пары семплов вычисляется форм-фактор. Таким образом, мы получаем классический форм-фактор, посчитанный трассировкой лучей для одного семпла.

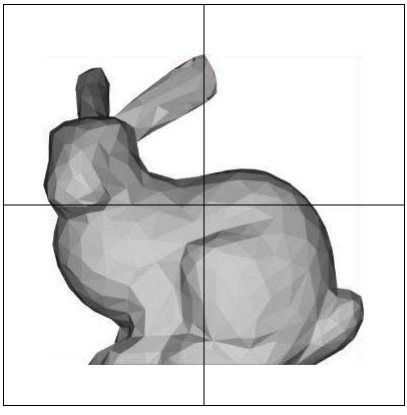


Рис. 1. Вокруг геометрии сцены генерируется сетка из 4-х вокселей
Fig.1. A grid of 4 voxels is generated around the geometry of the scene

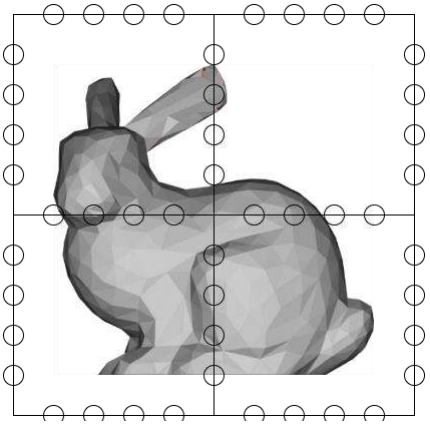


Рис. 2. На границах сетки равномерно распределяются начальные позиции для трассировки лучей
Fig.2. The initial positions for ray tracing are evenly distributed at the grid boundaries

После этого производится кластеризация семплов внутри вокселя, пока количество кластеров не будет ниже заданного значения. Семплы внутри кластеров объединяются в единый патч/площадку за счёт аддитивности форм-факторов. Таким образом, получаются патчи, не имеющие реальной геометрии.

Вначале для сцены вычисляется ограничивающий прямоугольный параллелепипед, выровненный по осям координат. Полученный объем разбивается на воксели (рис. 1). На гранях каждого вокселя равномерно выбираются точки (рис. 2). Из выбранных точек трассируются лучи внутрь вокселя. С помощью этих лучей находятся точки на исходной геометрии, которые влияют на освещение геометрии в других вокселях (рис. 3). При вычислениях учитываются только лучи, для которых найдены пересечения внутри рассматриваемого вокселя (рис. 4). Для точек пересечения внутри вокселя генерируются семплы (рис. 5).

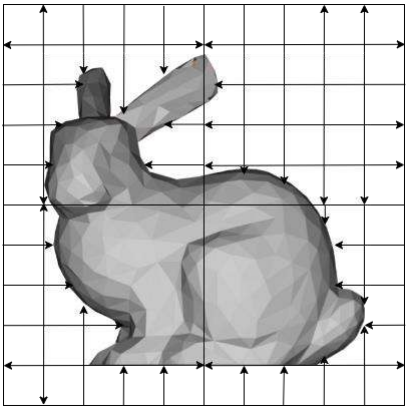


Рис. 3. Из сгенерированных точек выпускаются лучи вдоль нормали грани сетки
Fig.3. Rays are released from the generated points along the normal of the mesh face

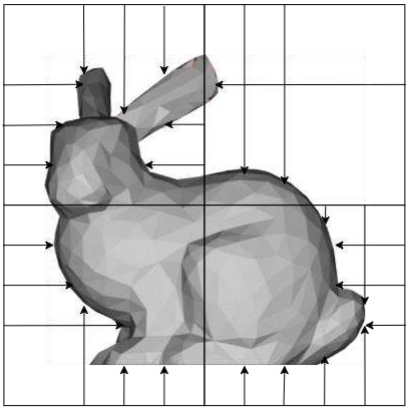


Рис. 4. Лучи, не имеющие пересечений с геометрией сцены или имеющие пересечение вне рассматриваемого вокселя, игнорируются
Fig.4. Rays that do not intersect with the geometry of the scene or have an intersection outside the voxel are ignored

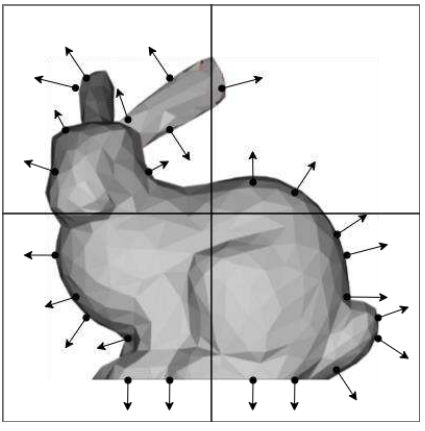


Рис. 5. На точках пересечения луча с геометрией генерируются семплы с заданной позицией, нормалью и цветом
Fig.5. Samples with the specified position, normal and color are generated at the points of intersection of the ray with the geometry

В качестве нормали и цвета берутся соответствующие значения исходной геометрии. Последний компонент семпла, площадь, требует большего количества вычислений. Берется площадь полигона S^h, попадающая в воксел, и делится на количество семплов K, попавших в этот полигон:

$$S_i^h = \frac{S^h}{K}.$$

Следующий этап пред-просчёта - это вычисление форм-факторов. Для каждой пары семплов проверяется их взаимная видимость и в случае успешной геометрии вычисляются форм-факторы:

$$F_{ij} = \frac{1}{S_i^h}\frac{cos\theta cos\theta'}{\pi r^2}V(i,j).$$

На этом этапе количество данных будет зависеть от количества семплов в каждом вокселе. Для хорошего качества освещения нужно несколько десятков семплов, что существенно увеличивает размер матрицы форм-факторов.

Мы использовали жадный подход к кластеризации. Внутри вокселя ищутся два семпла, которые наиболее близки по метрике:

$$M(i,j) = \left(1 + \left\|C_i - C_j\right\|_2\right)\left(2 - \left(N_i, N_j\right)\right)\left(1 + \sqrt{\sum_{h=0}^{N}\left(F_{ih} - F_{jh}\right)^2 + \left(F_{hi} - F_{hj}\right)^2}\right),$$

где C – цвет, а N – нормаль.

В данной метрике учитывается схожесть семплов по цвету, нормали и вычисленным форм-факторам. Это позволяет уменьшить количество артефактов, связанных с кластеризацией. Для найденных семплов проводится процесс слияния. Нормали, цвет и форм-факторы сливаются с учетом площадей, соответствующим семплам, по формулам:

$$N = normalize\left(\frac{N_i S_i^h + N_j S_j^h}{S_i^h + S_j^h}\right),$$

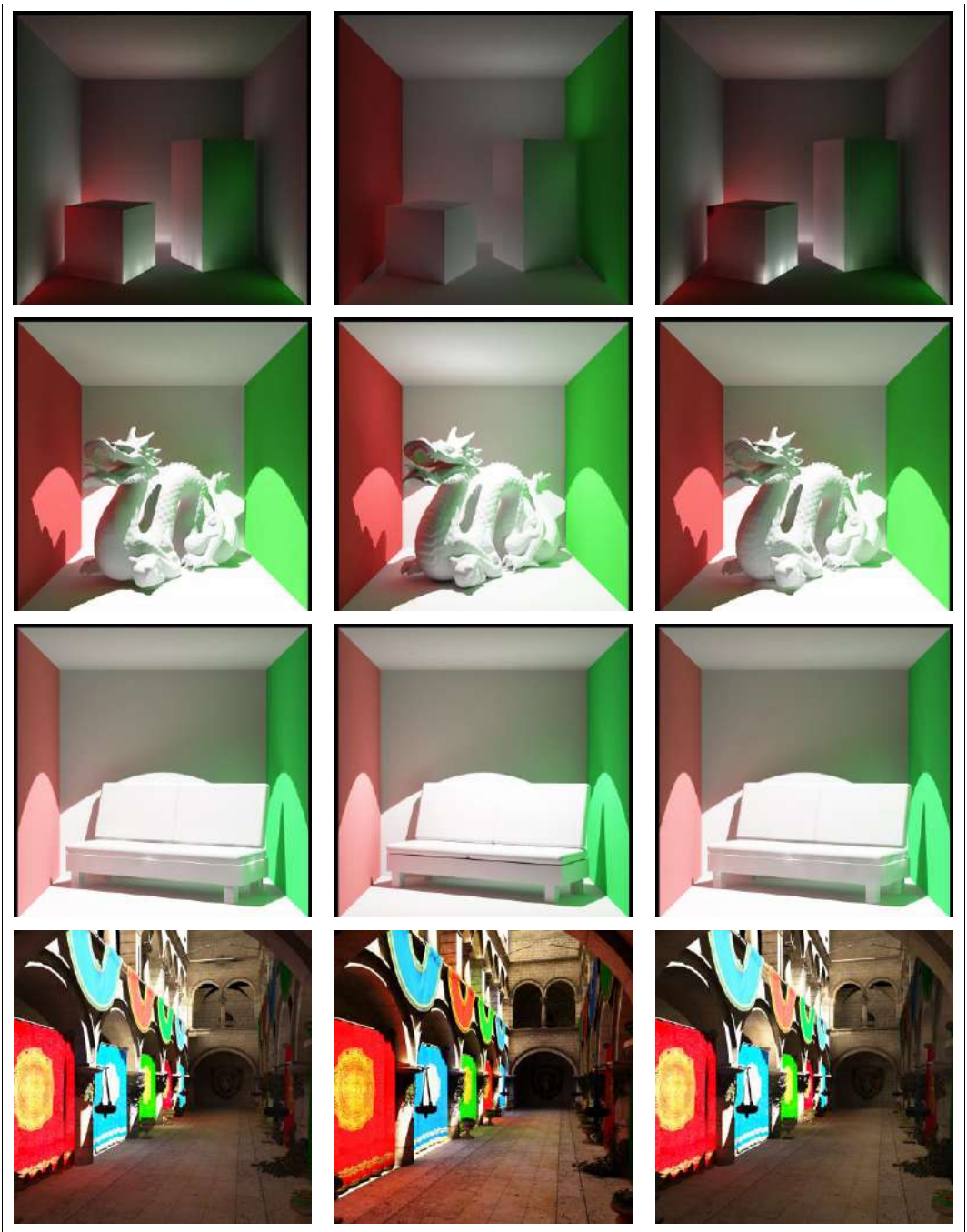
$$C = \frac{C_i S_i^h + C_j S_j^h}{S_i^h + S_j^h}.$$

Применимость такого подхода для форм-факторов описана в [1]. Данный процесс повторяется пока внутри воксела не останется желаемое количество патчей.

4. Сравнения результатов

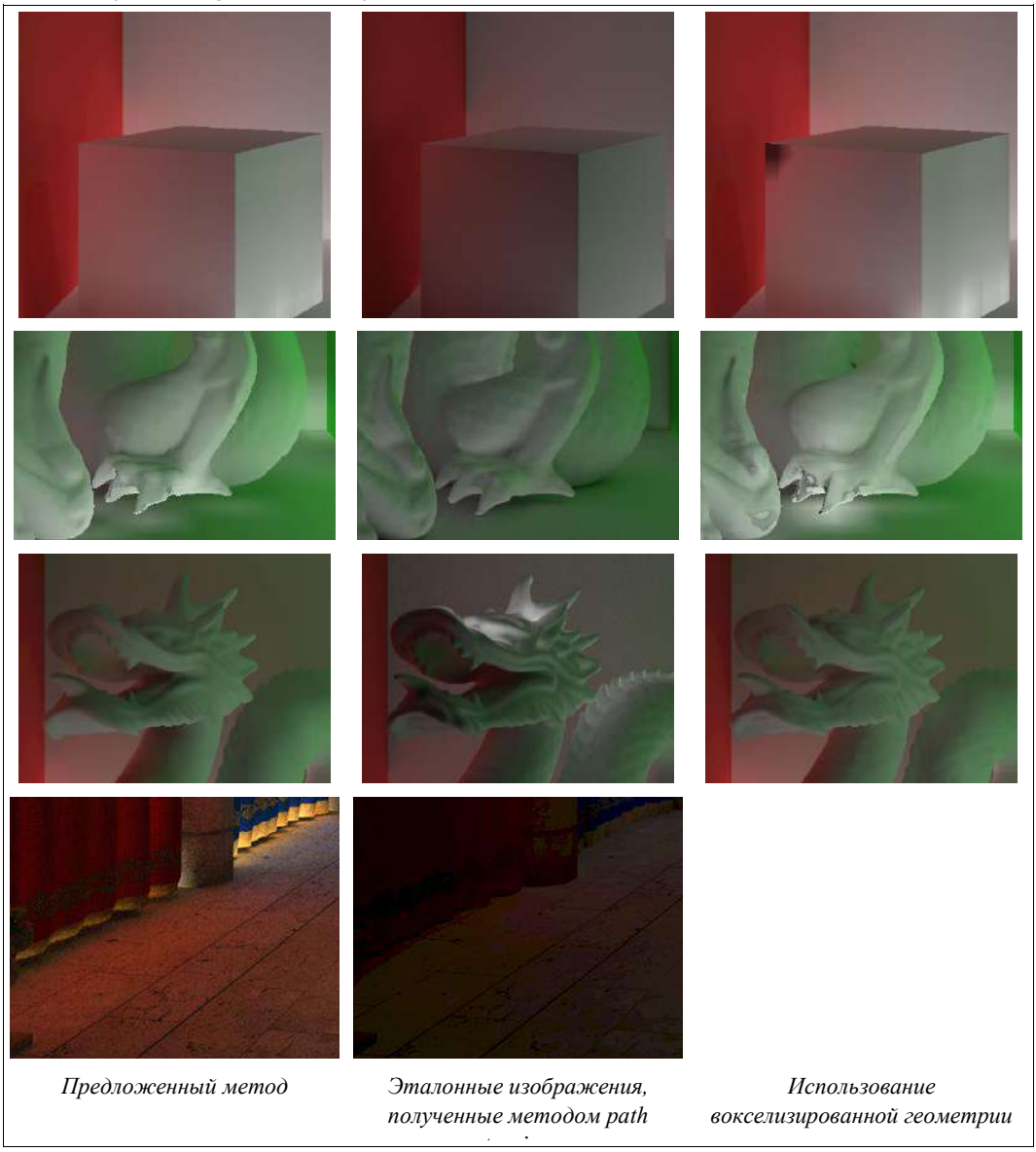
Для оценки качества были проведены сравнения изображений с эталонными, полученными методом трассировки путей. Предложенный метод даёт схожие отражения света на тестовых сценах (табл. 1). Были рассмотрены модели различной сложности, находящиеся в Cornell Box – кубической геометрии со стенами разного цвета. Также было проведено сравнение на сцене Sponza (нижний ряд в табл.1), где видно, что предложенный метод генерирует лучшее вторичное освещение от цветных флагов. Для первичного освещения были использованы карты теней с методом PCF [33] для мягких границ затенения. Сравнение полного освещения, полученного методом излучательности, не проводилось, так как дискретизация пространства даёт заметные артефакты освещения.

Табл. 1. Сравнения на различных сценах
Table 1. Comparison on different scenes



Предложенный метод	*Эталонные изображения, полученные методом path tracing*	*Использование вокселизированной геометрии*

Табл. 2. Артефакты при вокселизации сцены
Table 2. Artifacts during voxelization of the scene



Предложенный метод	*Эталонные изображения, полученные методом path*	*Использование вокселизированной геометрии*

Помимо этого, было проведено сравнение с излучательностью, использующей вокселизированное приближение сцены. Предложенный метод оказывается ближе к эталонному изображению по метрике MSE (табл. 3). На более сложных сценах метод излучательности имеет большую ошибку относительно трассировки путей, так как при дискретизации пространства теряется информация о деталях геометрии. Это особенно хорошо видно для ошибки на сцене Cry-Sponza. Для сцен с меньшим количеством деталей и, как следствие, более плавным изменением вторичного освещения метрика MSE меньше.

Табл. 3. Численное сравнение сцен с эталонным изображением, полученным методом трассировки путей (меньше — лучше)
Table 3. Numerical comparison of scenes with a reference image obtained by the path tracing method (less is better)

№ сцены (строки в Табл.1 и 2)	**MSE для предложенного метода**	**MSE для вокселизации**
1 (cornell box)	**4.90**	7.73
2 (dragon)	**9.24**	10.95
3 (cornell and sofa)	**2.68**	4.01
4 (Cry-Sponza)	**56.43**	61.96

Данный метод может быть улучшен с применением каскадов и нескольких масштабов воксельных сеток. При этом есть возможность добавить перенос освещения между разными каскадами для улучшения точности.

Важным улучшением, требуемым для любых реализаций методов излучательности, является решение этой задачи для различных масштабов сцены. Предложенный метод может использовать существующие подходы, связанные с кластеризацией [34], либо может быть расширен новыми вариантами с использованием воксельных сеток. Сцены большого масштаба могут быть разделены на части, к которым освещение применяется независимо. При этом данные для вычисления освещения могут загружаться по требованию при помощи механизмов стриминга данных.

При большом количестве деталей геометрии для более точного вторичного освещения метод виртуальных патчей может быть скомбинирован с различными техниками вычисления глобального освещения в экранном пространстве. При этом, так как предложенный метод учитывает освещение между удаленными друг от друга участками геометрии, метод в экранном пространстве может ограничиться вычислениями в небольшой окрестности, уменьшая тем самым количество промахов кеша при анализе освещения окружающей геометрии.

5. Заключение

В данной статье предложен метод виртуальных патчей, который используется при подготовке 3D-сцены для применения в методе излучательности. Этот метод позволяет избежать ручного и полуавтоматического упрощения геометрии, которое необходимо для вычисления вторичного освещения методом излучательности на реальных сценах. В то же время данный подход увеличивает качество финального освещения по сравнению с воксельным приближением сцены (табл. 2), которое также является полностью автоматическим методом упрощения геометрии. При этом увеличивается только вычислительная сложность этапа предобработки сцены и не требуется дополнительных вычислений при визуализации изображения. Количество данных, необходимых для вычисления освещения, также не увеличивается по сравнению с вокселизацией. Предложенный метод может быть скомбинирован с различными техниками и алгоритмами, которые позволяют расширить его для сцен произвольного размера и учесть вторичное освещение для высокодетализированной геометрии.

Список литературы / References

[1]. Sillion F. X., Puech C. Radiosity and global illumination. Morgan Kaufmann, San Francisco, 1994. 251p.
[2]. Hanrahan P., Salzman D., and Aupperle L. A rapid hierarchical radiosity algorithm. ACM SIGGRAPH Computer Graphics, vol. 25, issue 4, 1991, pp. 197-206.

[3]. Cohen M.F., Chen S.E. et al. A progressive refinement approach to fast radiosity image generation. In Proc. of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH ’88), 1988, pp. 75-84.
[4]. Andújar C. Geometry Simplification. Research report Nr LSI-99-2-R, Universitat Politècnica de Catalunya, Barcelona, 1999, 73 p.
[5]. Scherzer D., Wimmer M., Purgathofer W. A Survey of Real-Time Hard Shadow Mapping Methods. Computer Graphics Forum, vol. 30, issue 1, 2011. pp. 169-186.
[6]. Keller A., McGuire M. et al. Ray Tracing Gems. High-quality and real-time rendering with DXR and other APIs. Apress, 2019, 651 p.
[7]. Hasenfratz J.-M., Lapierre M. et al. A survey of RealTime Soft Shadows Algorithms. Computer Graphics Forum, vol. 22, issue 4, 2003, pp.753-774.
[8]. Keller A. Instant Radiosity. In Proc. of the 24th Annual Conference on Computer Graphics and Interactive Techniques, 1997, pp. 49-56.
[9]. Dachsbacher C., Stamminger M. Reflective shadow maps. In Proc. of the 2005 Symposium on Interactive 3D Graphics and Games, 2005. pp. 203-231.
[10]. Grosch T., Ritschel T. Screen-Space Directional Occlusion. In GPU Pro: Advanced Rendering Techniques, A K Peters/CRC Press, 2010, 16 p.
[11]. Mittring, M. Finding next gen: Cry Engine 2. In ACM SIGGRAPH 2007 courses, 2007, pp. 97-121.
[12]. Jimenez. J, Wu X.-C. et al. Practical Real-Time Strategies for Accurate Indirect Occlusion. Technical Memo ATVI-TR-19-01, Activision, 2016.
[13]. Kaplanyan A. and Dachsbacher C. 2010. Cascaded Light Propagation Volumes for Real-Time Indirect Illumination. In Proc. of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D '10), 2010, pp. 99-107.
[14]. Crassin C., Neyret F. et al. Interactive Indirect Illumination Using Voxel Cone Tracing: a Preview. In Proc. of the Symposium on Interactive 3D Graphics and Games (I3D '11), 2011, p. 207.
[15]. Mara M., McGuire M. et al. An efficient denoising algorithm for global illumination. In Proc. of High Performance Graphics (HPG ’17), 2017, article no. 3, 7 p.
[16]. Shi W., Caballero J. et al. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1874-1883.
[17]. Green R. Spherical harmonic lighting: The gritty details. Technical report, Sony Computer Entertainment America, 2003, 47 p.
[18]. Manson, J. and Sloan, P.-P. Fast Filtering of Reflection Probes. Computer Graphics Forum, vol. 35, issue 4, 2016, pp. 119-127.
[19]. McGuire M., Mara M. et al. Real-Time Global Illumination Using Precomputed Light Field Probes. In Proc. of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D '17), 2017, article no. 2, 11 p.
[20]. Tabellion E. and Lamorlette A. An Approximate Global Illumination System for Computer-Generated Films. ACM Transactions on Graphics, vol. 23, issue 3, 2004, pp 469-476
[21]. RTX Global Illumination (RTXGI). URL: https://developer.nvidia.com/rtxgi.
[22]. Flatt D. AMD Radeon Rays Integrated into Unity's GPU Progressive Lightmapper. URL: https://blogs.unity3d.com/2018/03/29/amd-radeon-rays-integrated-into-unitys-gpu-progressive-lightmapper/.
[23]. Bush J. Quake Lightmaps. URL: https://jbush001.github.io/ 2015/06/11/quake-lightmaps.html.
[24]. Sloan P.-P., Kautz J., and Snyder J. Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments. ACM Transactions on Graphics, vol. 21, issue 3, 2002, pp 527-536.
[25]. Ren P., Wang J. et al. Global Illumination with Radiance Regression Functions. ACM Transactions on Graphics, vol. 32, issue 4, 2013, article no. 130, 12 p.
[26]. Thomas M. M., Forbes A. Deep Illumination: Approximating Dynamic Global Illumination with Generative Adversarial Network. arXiv:1710.09834, 2017. 10 p.
[27]. Alliez P., Ucelli G. et al. Recent Advances in Remeshing of Surfaces. In Shape Analysis and Structuring. Mathematics and Visualization. Springer, 2008, pp 53-82.
[28]. Cheng B., Liu Q. et al. Building Simplification Using Backpropagation Neural Networks: A Combination of Cartographers' Expertise and Raster-Based Local Perception. GIScience & Remote Sensing, vol. 50, issue 5, 2013, pp. 527-542.

[29]. Щербаков А.С. Расчет освещённости при помощи метода излучательности на графических процессорах для интерактивных приложений. Сборник тезисов лучших выпускных работ факультета ВМК МГУ 2017 года, Москва, 2017, стр. 95–97 / Shcherbakov A.S. Illuminance Calculation Using the Radiance Method on GPUs for Interactive Applications. Collection of abstracts of the best graduation papers of the faculty of the CMC MSU in 2017, Moscow, 2017, pp. 95–97 (in Russian).
[30]. Lorensen W., Cline H. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. ACM SIGGRAPH Computer Graphics, vol. 21, issue 4, 1987, pp 163-169.
[31]. Anders K.-H. Level of Detail Generation of 3D Building Groups by Aggregation and Typefication. In Proc. of the XXII International Cartographic Conference, 2005, 8 p.
[32]. Pfister H., Zwicker M. et al. Surfels: Surface Elements as Rendering Primitives. In Proc. of the 27th Annual Conference on Computer Graphics and Interactive Techniques, 2000, pp. 335-342.
[33]. Bunnell M., Pellacini F. URL: Shadow Map Antialiasing https://developer.nvidia.com/gpugems/gpugems/part-ii-lighting-and-shadows/chapter-11-shadow-map-antialiasing
[34]. Rai P., Shubha S. A Survey of Clustering Techniques. International Journal of Computer Applications, vol. 7, no. 12, 2010, article no. 1, 5 p.

Информация об авторах / Information about authors

Александр Станиславович Щербаков — аспирант факультета Вычислительной математики и кибернетики, лаборатории компьютерной графики и мультимедиа. Сфера научных интересов: компьютерная графика, глобальное освещение, программирование GPU.

Alexandr Shcherbakov is a graduate student of the Faculty of Computational Mathematics and Cybernetics, Laboratory of Computer Graphics and Multimedia. Research interests: computer graphics, global illumination, GPU programming.

Владимир Александрович ФРОЛОВ – кандидат физико-математических наук, старший научный сотрудник ИПМ РАН, научный сотрудник факультета ВМК МГУ. Сфера научных интересов: реалистичная компьютерная графика, моделирование освещённости, разработка программных систем оптического моделирования, параллельные и распределённые вычисления.

Vladimir FROLOV – PhD in computer graphics, senior researcher at Keldysh Institute of Applied Mathematics and researcher in computer graphics at Moscow State University. Research interests: realistic computer graphics, light transport simulation, elaboration of optical simulation software systems, GPU computing.

Владимир Александрович ГАЛАКТИОНОВ – доктор физико-математических наук, профессор, заведующий отделом компьютерной графики и вычислительной оптики. Сфера научных интересов: компьютерная графика, оптическое моделирование, создание программных систем оптического моделирования.

Vladimir GALAKTIONOV – Doctor of Science in physics and mathematics, Professor, Head of Computer graphics department. Research interests: computer graphics, optical simulation, elaboration of optical simulation software.

DOI: 10.15514/ISPRAS-2022-34(3)-5

Метод аппаратной реализации сверточной нейронной сети на основе системы остаточных классов

[1] *М.В. Валуева, ORCID: 0000-0002-4732-3216 <mriya.valueva@mail.ru>*
[1] *Г.В. Валуев, ORCID: 0000-0003-2049-7213 <mail@gvvaluev.ru>*
[2,3] *М.Г. Бабенко, ORCID: 0000-0001-7066-0061 <mgbabenko@ncfu.ru>*
[3,4,5] *А. Черных, ORCID: 0000-0001-5029-5212 <chernykh@cicese.mx>*
[5] *Х.М. Кортес Мендоса, ORCID: 0000-0001-7209-8324 <kortesmendosak@susu.ru>*

[1] *Северо-Кавказский центр математических исследований СКФУ*
355017, Россия, г. Ставрополь, ул. Пушкина, 1
[2] *Северо-Кавказский федеральный университет,*
355017, Россия, г. Ставрополь, ул. Пушкина, 1
[3] *Институт системного программирования им. В.П. Иванникова РАН,*
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
[4] *Центр научных исследований и высшего образования,*
Мексика, 22860, Нижняя Калифорния, Энсенада, ш. Тихуана-Энсенада, 3918
[5] *Южно-Уральский государственный университет,*
454080, Россия, Челябинск, проспект Ленина, 76

Аннотация. Сверточные нейронные сети (СНС) показывают высокую точность при решении задачи распознавания образов, но обладают высокой вычислительной сложностью, что приводит к медленной обработке данных. Для увеличения быстродействия СНС в данной работе предлагается метод аппаратной реализации СНС с вычислениями в системе остаточных классов с модулями специального вида $\mathbf{2^{\alpha}}$ и $\mathbf{2^{\alpha}-1}$. В статье представлено аппаратное моделирование предлагаемого метода на FPGA на примере СНС LeNet-5, обученной на базах изображений MNIST, FMNIST и CIFAR-10. Моделирование показало, что применение предлагаемого подхода позволяет увеличить тактовую частоту и производительность устройства примерно на 11% –12%, по сравнению с традиционным подходом на основе позиционной системы счисления. Тем не менее, увеличение скорости работы устройства достигнуто за счет увеличения аппаратных затрат. Предлагаемый в статье метод может быть применен системах распознавания образов, когда необходимо обеспечить высокую скорость обработки данных.

Ключевые слова: сверточная нейронная сеть; система остаточных классов; распознавание образов; field-programmable gate array (FPGA)

Для цитирования: Валуева М.В., Валуев Г.В., Бабенко М.Г., Черных А., Кортес Мендоса Х.М. Метод аппаратной реализации сверточной нейронной сети на основе системы остаточных классов. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 61-74. DOI: 10.15514/ISPRAS–2022–34(3)–5.

Благодарности: Работа выполнена при поддержке Российского научного фонда, проект №19-71-10033.

Method for Convolutional Neural Network Hardware Implementation Based on a Residue Number System

[1] *M.V. Valueva, ORCID: 0000-0002-4732-3216 <mriya.valueva@mail.ru>*
[1] *G.V. Valuev, ORCID: 0000-0003-2049-7213 <mail@gvvaluev.ru>*
[2,3] *M.G. Babenko, ORCID: 0000-0001-7066-0061 <mgbabenko@ncfu.ru>*
[3,4,5] *A. Tchernykh, ORCID: 0000-0001-5029-5212 <chernykh@cicese.mx>*
[5] *J. M. Cortes-Mendoza, ORCID: 0000-0001-7209-8324 <kortesmendosak@susu.ru>*

[1] *North-Caucasus Center for Mathematical Research NCFU*
1, Pushkin St., Stavropol, 355017, Russia
[2] *North-Caucasus Federal University,*
1, Pushkin st., Stavropol, 355017, Russia
[3] *Ivannikov Institute for System Programming of the Russian Academy of Sciences*
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia
[4] *CICESE Research Center,*
3918, Ensenada-Tijuana Highway, Ensenada, 22860, Mexico
[5] *South Ural State University*
76, Lenin prospekt, Chelyabinsk, 454080, Russia

Abstract. Convolutional Neural Networks (CNN) show high accuracy in pattern recognition solving problem but have high computational complexity, which leads to slow data processing. To increase the speed of CNN, we propose a hardware implementation method with calculations in the residue number system with moduli of a special type $\mathbf{2^{\alpha}}$ and $\mathbf{2^{\alpha}-1}$. A hardware simulation of the proposed method on Field-Programmable Gate Array for LeNet-5 CNN is trained with the MNIST, FMNIST, and CIFAR-10 image databases. It has shown that the proposed approach can increase the clock frequency and performance of the device by 11%-12%, compared with the traditional approach based on the positional number system.

Keywords: convolutional neural network, residue number system, pattern recognition, field-programmable gate array (FPGA)

For citation: Valueva M.V., Valuev G.V., Babenko M.G., Tchernykh A., Cortes-Mendoza J. M. Method for Convolutional Neural Network Hardware Implementation Based on a Residue Number System. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 61-74 (in Russian). DOI: 10.15514/ISPRAS–2022–34(3)–5

Acknowledgements. This work was supported in part by the Russian Science Foundation, project number 19-71-10033.

1. Введение

Одним из основных методов распознавания визуальных образов являются сверточные нейронные сети (СНС), поскольку они показывают высокую точность. Они широко используются в области медицины [1-3], в системах видеонаблюдения [4, 5], в робототехнике [6] и других областях. Но высокая точность распознавания достигается за счет увеличения количества слоев сети, а, следовательно, и ее вычислительной сложности [7]. Это приводит к необходимости разработки специализированных аппаратных ускорителей нейросетевых вычислений.

Одним из эффективных путей улучшения технических характеристик цифровых устройств является оптимизация вычислений на арифметическом уровне. Например, авторы работ [8-10] предлагают выполнять вычисления в системе остаточных классов (СОК) для построения эффективной аппаратной реализации глубоких нейронных сетей.

В данной работе предлагается метод аппаратной реализации СНС с вычислениями в СОК с модулями вида 2^α и 2^α-1. Так же в работе представлена реализация предлагаемого метода на программируемой пользователем вентильной матрице (field-programmable gate array,

FPGA) и произведено сравнение с реализаций СНС в традиционной позиционной системе счисления (ПСС).

Оставшаяся часть статьи организована следующим образом. В разд. 2 описан предлагаемый метод применения СОК для аппаратной реализации СНС. В разд. 3 представлены результаты аппаратного моделирования предлагаемого метода на примере СНС LeNet-5. Был произведен анализ результатов моделирования, которые представлены в разд. 4. Выводы по проведенному исследования находятся в разд. 5.

2. Предлагаемый метод

Любое целое число $0 \le A < M$ может быть однозначно представлено в СОК в виде остатков от деления на попарно взаимно простые модули $\{m_1, m_2, \dots, m_n\}$ как $A = \{a_1, a_2, \dots, a_n\}$, так что $a_i = |A|_{m_i}$ [11], при этом $M = \prod_{i=1}^{n} m_i$ называется динамическим диапазоном СОК. Для представления отрицательных чисел в СОК, динамический диапазон системы делится на две примерно равные части, при этом должно выполняться одно из следующих условий:

$$\begin{aligned} -\tfrac{M-1}{2} \le A \le \tfrac{M-1}{2} \text{ для нечетных } M, \\ -\tfrac{M}{2} \le A \le \tfrac{M}{2} - 1 \text{ для четных } M, \end{aligned} \quad (1)$$

Арифметические операции над числами, представленными в СОК, производятся параллельно по каждому модулю:

$$A * B = (|a_1 * b_1|_{p_i}, |a_2 * b_2|_{p_2}, \dots, |a_n * b_n|_{p_n}), \quad (2)$$

где $*$ означает операцию сложения, вычитания или умножения.

Число $A = \{a_1, a_2, \dots, a_n\}$ может быть преобразовано из СОК в позиционную систему счисления (ПСС) с использованием Китайской теоремы об остатках [11]

$$A = \left| \sum_{i=1}^{j} \left(\left| M_i^{-1} \right|_{m_i} a_i \middle| M_i \right) \right|_M, \quad (3)$$

где $M_i = \frac{M}{m_i}$, а $\left|M_i^{-1}\right|_{m_i}$ – мультипликативный обратный элемент для M_i.

Первым блоком, при выполнении вычислений в СОК, является перевод чисел из ПСС в СОК. Для преобразования числа необходимо вычислить остатки от деления по каждому модулю. Данная операция является ресурсозатратной. Использование модулей вида $\{2^{\alpha_1}, 2^{\alpha_2} - 1, \dots, 2^{\alpha_n} - 1\}$ позволяет избежать данной операции. Операция вычисления остатка от деления по модулю вида 2^{α} осуществляется простым оставлением α младших бит исходного числа, с отбрасыванием оставшихся старших значащих бит. Введем для устройства, выполняющего вычисление остатка от деления по модулю 2^{α} обозначение $MOD_{2^{\alpha}}$. Для модулей вида $2^{\alpha} - 1$ вычисление остатка от деления α-битных чисел по модулю $2^{\alpha} - 1$ [12], обозначим данное устройство как $MOD_{2^{\alpha}-1}$. При сложении по модулю $2^{\alpha} - 1$ больше двух слагаемых используют устройство для сложения нескольких чисел по модулю $MOMA_{2^{\alpha}-1}$, которое состоит из дерева сумматоров с сохранением переноса и циклическим переносом старшего бита (EAC-CSA) [13, 14], а результат передается на сумматор Когге-Стоуна с циклическим переносом старшего бита (EAC-KSA) [14, 15]. Таким образом, устройство $PNS \to RNS$ для перевода числа DR-битного числа A в СОК с модулями вида $\{2^{\alpha_1}, 2^{\alpha_2} - 1, \dots, 2^{\alpha_n} - 1\}$ состоит из одного устройства $MOD_{2^{\alpha}}$ и $n - 1$ устройств $MOD_{2^{\alpha}-1}$. На выходе устройства формируется число $\{a_1, a_2, \dots, a_n\}$, представленное в СОК и имеющее разрядности $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ соответственно. Далее выполняются вычисления в СОК параллельно по каждому вычислительному каналу, соответствующему модулю системы.

Все вычисления в предлагаемой архитектуре СНС производятся параллельно с использованием одного вычислительного канала по модулю 2^α и $n-1$ вычислительных каналов по модулю вида $2^\alpha - 1$.

Предположим, что на вход сверточного слоя поступает набор из D карт признаков I_{in} состоящих из R строк, C столбцов. Это означает, что вход сверточного слоя можно описать как трехмерную функцию $I(x,y,z)$, где $0 \le x < R$, $0 \le y < C$ и $0 \le z < D$ пространственные координаты, а амплитуда I в любой точке с координатами (x,y,z) это интенсивность пикселей в этой точке. Процедура получения одной карты признаков в сверточном слое по модулю m_l может быть представлена в виде:

$$\left|I_f(x,y)\right|_{m_l} = \left| |b|_{m_l} + \left| \sum_{i=-t}^{t} \sum_{j=-t}^{t} \sum_{z=0}^{D-1} \left|W_{i,j,z}\right|_{m_l} \cdot |I(x+i,y+j,z)|_{m_l} \right|_{m_l} \right|_{m_l}, \quad (4)$$

где I_f – карта признаков после свертки, $W_{i,j,z}$ – это коэффициенты 3D-фильтра размерности $d \times d$ для обработки D двухмерных массивов, $t = \left\lfloor \frac{d}{2} \right\rfloor$ и b – смещение [16].

Пусть $F = \{F_0, F_1 \dots, F_{D-1}\}$ набор из D векторов размерности d^2, которые соответствуют фрагментам размерности $d \times d$ карт признаков I, поступающих на вход сверточного слоя. Аналогично, представим маску фильтра как набор из D векторов размерности d^2 и обозначим как $W = \{W_0, W_1 \dots, W_{D-1}\}$. Тогда операция свертки по модулю m_l для вычисления одного значения R карты признаков I_f может быть представлена в виде суммы одномерных сверток и прибавлении смещения

$$|R|_{m_l} = \left| |b|_{m_l} + \left| \sum_{i=0}^{D-1} \sum_{j=0}^{d^2-1} \left|W_{i,j}\right|_{m_l} \cdot \left|F_{i,j}\right|_{m_l} \right|_{m_l} \right|_{m_l}, \quad (5)$$

Одномерная свертка может быть реализована с помощью фильтра с конечной импульсной характеристикой (FIR) [17], который состоит из блоков умножения с накоплением (MAC). Блоки MAC состоят из генератора частичных произведений PPG [13], содержащего вентили AND, и дерева сумматоров. Результаты одномерной свертки суммируются с помощью сумматора со множественным входом. Если расчеты производятся по модулю 2^α, то биты старше α не участвуют в вычислениях. Для выполнения вычислений по модулю $2^\alpha - 1$ используется техника циклического переноса старших бит EAC. На рис. 1 представлена схема устройства $CONV_{2^\alpha}$ для свертки фрагмента изображения по модулю 2^α. Устройство свертки $CONV_{2^\alpha-1}$ по модулю $2^\alpha - 1$ имеет аналогичную структуру.

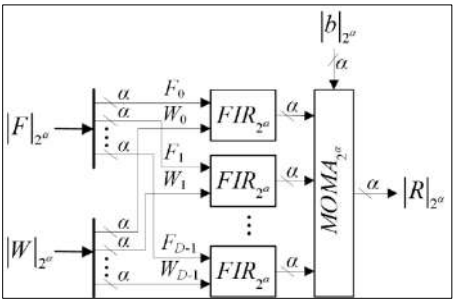


Рис. 1. Схема устройства $CONV_{2^\alpha}$ для свертки по модулю 2^α фрагмента карты признаков
Fig. 1. The $CONV_{2^\alpha}$ device circuit for modulo 2^α convolution of feature map fragment

На выходе сверточного слоя применяется функция активации. Наиболее распространенной функцией является линейный выпрямитель (ReLU) [18], которая имеет вид $max(0,R)$ и

сводится к определению знака числа. Количество карт признаков на выходе сверточного слоя соответствует количеству масок фильтров.

Операция определения знака числа является ресурсозатратной в СОК, так как требует вычисления позиционной характеристики числа [19]. Вычисление позиционной характеристики числа по формуле (3) требует вычисления остатка от деления на число M с разрядностью полного диапазона системы DR. На практике, одним из самых эффективных подходов является модификация КТО, называемая КТО с дробными величинами (КТОд) [20]. Согласно КТОд позиционную характеристику числа A' можно вычислить по формуле:

$$A' = \left| \sum_{i=1}^{n} a_i \tilde{k}_i \right|_{2^N}, \tag{6}$$

где $\tilde{k}_i = \left\lfloor 2^N \frac{|P_i^{-1}|_{p_i} P_i}{P} \right\rfloor$. Для гарантированного точного перевода чисел из СОК в ПСС достаточно выбрать N равным:

$$N = \lceil \log_2(P\mu) \rceil - 1, \tag{7}$$

где $\mu = -n + \sum_{i=1}^{n} p_i$.

Для определения знака числа в СОК по его позиционной характеристике A' необходимо выполнить проверку следующих условий:

- если $0 \le A' < 2^{N-1}$, тогда число A положительное;
- если $2^{N-1} < A' < 2^N$, тогда число A отрицательное.

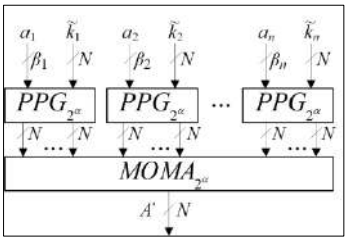


Рис. 2. Архитектура устройства NPC_{CRTf} для вычисления позиционной характеристики числа, представленного в СОК, с помощью КТОд

Fig. 2. Architecture of the NPC_{CRTf} device for calculating the positional characteristic of a number represented in RNS using CRTf

На рис. 2 представлена схема устройства, которое вычисляет позиционную характеристику числа по формуле (6) с помощью метода КТОд. Введем для него обозначение NPC_{CRTf}. На вход устройства поступает число $\{a_1, a_2, \dots, a_n\}$, представленное в СОК с модулями $\{m_1, m_2, \dots, m_n\}$ и разрядностями $\{\beta_1, \beta_2, \dots, \beta_n\}$ по каждому модулю соответственно. Также на вход подаются коэффициенты $\tilde{k}_i$, имеющие разрядность N бит, они являются константами и могут быть вычислены предварительно. Генерация частичных произведений осуществляется с помощью устройств PPG_{2^α}. Далее N-битные частичные произведения складываются с помощью устройства $MOMA_{2^\alpha}$. Устройства PPG_{2^α} и $MOMA_{2^\alpha}$ являются N-битными, то есть $\alpha = N$. Если старший значащий бит (Most Significant Bit, MSB) числа A' равен 0, то число A отрицательное, если равен 1, то положительное.

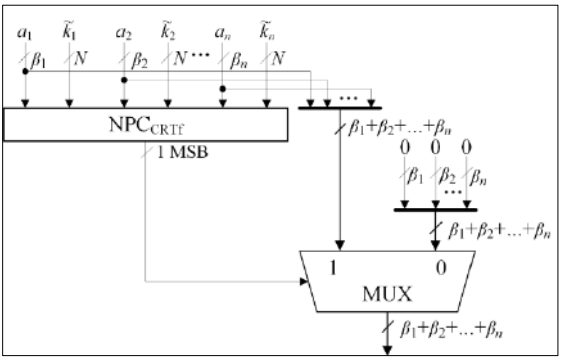


Рис. 3. Архитектура устройства ReLU для вычисления функции активации ReLU
Fig. 3. ReLU device architecture for computing the ReLU activation function

На рис. 3 представлена архитектура устройства ReLU для вычисления функции активации ReLU в СОК. Здесь знак числа определяется с помощью старшего значащего бита позиционной характеристики числа, который передается в качестве управляющего сигнала на вход мультиплексора MUX, принимающего решение какое значение подавать на выход устройства.

НС обычно использует большое количество фильтров в сверточном слое. Это приводит к резкому увеличению объема обрабатываемых данных в сети. Для их уменьшения используется слой выборки (pooling). Чаще всего используется выборка максимальных элементов в некоторой рассматриваемой окрестности (max pooling). На вход слоя поступает массив карт признаков, состоящий из D карт, содержащих R строк и C столбцов. Следовательно, вход данного слоя можно описать как трехмерную функцию $P_{in}(x_{in}, y_{in}, z)$, где $0 \le x_{in} < R$, $0 \le y_{in} < C$ и $0 \le z < D$ – пространственные координаты, а амплитуда P_{in} в любой точке с координатами (x_{in}, y_{in}, z) – интенсивность пикселей в данной точке. Процедуру максимального элемента из окрестности размером $s \times s$ с шагом s для z-ой карты признаков можно описать формулой:

$$P_{out}(x_{out}, y_{out}, z) = \max_{\substack{s \cdot x_{out} \le x_{in} \le s \cdot (x_{out}+1), \\ s \cdot y_{out} \le y_{in} \le s \cdot (y_{out}+1)}} \{P_{in}(x_{in}, y_{in}, z)\} \quad (8)$$

где $P_{out}(x_{out}, y_{out}, z)$ – набор из D карт признаков на выходе сверточного слоя, $0 \le x_{out} < \frac{R}{s} - 1$, $0 \le y_{out} < \frac{C}{s} - 1$.

Операция сравнения двух чисел в СОК сводится к сравнению их позиционных характеристик. Таким образом устройство для сравнения чисел в СОК по методу КТОд состоит из двух устройств NPC_{CRTf}, вычисляющих позиционную характеристику числа, и устройства COMP, выполняющего сравнение двух чисел в ПСС. На рис. 4 представлено устройство MAX, которое выполняет выбор большего числа из двух $A = \{a_1, a_2, \ldots, a_n\}$ и $B = \{b_1, b_2, \ldots, b_n\}$, представленных в СОК. Выход устройства COMP, выполняющего сравнение позиционных характеристик, подается в качестве управляющего сигнала на вход мультиплексора MUX, который принимает решение какое из двух чисел в СОК передать на выход устройства.

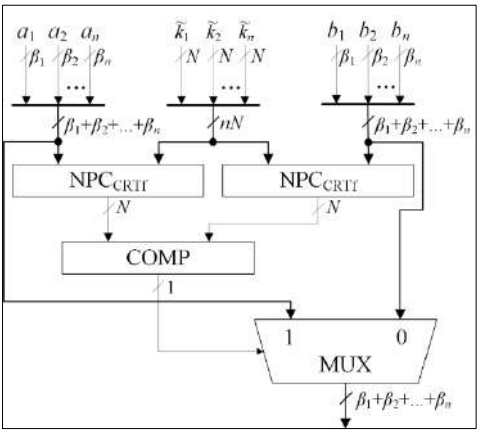


Рис. 4. Архитектура устройства MAX для выбора большего числа из двух в СОК
Fig. 4. MAX device architecture for choosing largest out of two numbers in RNS

Наиболее часто на слое max pooling используется $s = 2$. То есть рассматривается окрестность размерности 2×2, и вычисления производятся с шагом 2. Так как слой max pooling всегда идет после сверточного слоя, то позиционные характеристики чисел были рассчитаны при вычислении функции активации ReLU и поступают на вход устройства. Выбор наибольшего числа производится с помощью дерева устройств MAX.

Заключительными слоями сети являются полносвязные слои нейронов, выполняющие функцию классификатора **Ошибка! Источник ссылки не найден.**. Пусть $X = \{x_i\}$, – вектор, подаваемый на вход $(p-1)$-го слоя, где $0 \le i < m$ и m– общее число входов. Каждый элемент вектора умножается на соответствующий весовой коэффициент $W_j = \{w_{ji}\}$, $0 \le i < m$, $0 \le j < n$, n– количество нейронов $(p-1)$-го слоя и результат суммируется

$$y_j = b_j + \sum_{i=0}^{m-1} w_{ji} x_i. \qquad (9)$$

Обозначим функцию активации для $(p-1)$-го слоя как $\phi(t)$, тогда результатом $(p-1)$ -го слоя является вектор $\{h_j\}$, $0 \le j < n$, элементы которого вычисляются как $h_j = \phi(y_j)$.

Результат вычислений $(p-1)$-го слоя подается на вход p-го слоя, производим процедуру умножения на соответствующие весовые коэффициенты, складываем и подаем на функцию активации, результатом вычислений является вектор $\{z_k\}$, $0 \le k < l$, l– количество нейронов p-го слоя. Результат последнего слоя нормализуется с помощью функции softmax, таким образом на выходе полносвязных слоев формируется вектор $\{g_k\}$, $0 \le k < l$, элементы которого вычисляются следующим образом $g_k = \frac{e^{z_k}}{\sum_{i=0}^{l-1} e^{z_i}}$. На выходе полносвязного слоя формируется вектор, количество элементов которого соответствует количеству классов и отображает вероятность принадлежности образа, подаваемого на вход сети, к каждому классу.

На рис. 5 представлена архитектура устройства FC_{2^α}, выполняющего вычисления по формуле (9) для j-го нейрона по модулю 2^α. Устройство $FC_{2^\alpha-1}$, выполняющее вычисления для j-го нейрона по модулю $2^\alpha - 1$ имеет аналогичную архитектуру, но использует технику EAC. Результаты работы устройств FC_{2^α} и $FC_{2^\alpha-1}$ подаются на вход устройства ReLU.

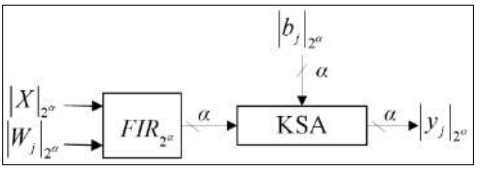


Рис. 5. Архитектура устройства FC_{2^α} полносвязного слоя с вычислениями по модулю 2^α
Fig. 5. Device architecture FC_{2^α} for calculating the output of one neuron of a fully connected layer with calculations modulo 2^α

Последним блоком системы, выполняющей вычисления в СОК, является преобразование результата обратно в ПСС. Для перевода числа из СОК в ПСС, согласно КТОд, необходимо умножить позиционную характеристику A' на модуль M. При этом результатом алгоритма являются старшие биты начиная с бита под номером $N+1$. Таким образом:

$$A = \frac{A'M}{2^N} \tag{10}$$

В (10) операция деления при аппаратной реализации игнорируется, так как на выход подаются старшие значащие биты начиная с $(N+1)$-го. В программной реализации эта операция эквивалентна сдвигу на N разрядов вправо. Тогда в устройстве $RNS \to PNS$ для обратного преобразования чисел из СОК в ПСС позиционная характеристика числа вычисляется с помощью устройства NPC_{CRTf}, а умножение на динамический диапазон системы производится с помощью $(N+DR)$-разрядного генератора частичных произведений PPG_{2^α} и сумматора $MOMA_{2^\alpha}$ такой же разрядности.

На рис. 6 представлена архитектура СНС с вычислениями в СОК. На вход поступает изображение в виде последовательности пикселей. Сперва в блоке $PNS \to RNS$ производится преобразование данных в СОК. Затем они поступают в оперативное запоминающее устройство (ОЗУ) и передаются в другие блоки СНС (сверточные слои, слои max pooling и полносвязные слои). Весовые коэффициенты СНС хранятся в постоянном запоминающем устройстве. Перевод весовых коэффициентов в СОК производится с помощью блоков $PNS \to RNS$, затем данные в формате СОК поступают на сверточные и полносвязные слои. Результат работы СНС переводится в ПСС с помощью блока $RNS \to PNS$ и поступает на выход устройства.

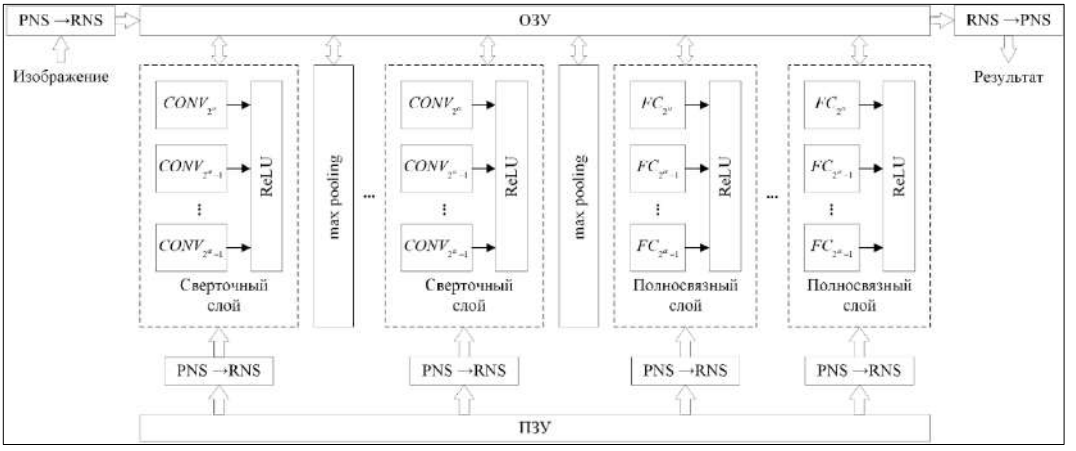


Рис. 6. Предлагаемая архитектура СНС с вычислениями в СОК
Fig. 6. Proposed CNN architecture with computations in RNS

Представленный метод для аппаратной реализации СНС с вычислениями в СОК позволяет выполнять вычисления в сверточных и полносвязных слоях параллельно с числами меньшей размерности, чем в ПСС, что увеличивает быстродействие системы. В следующем разделе представлен пример применения на практике предложенного метода.

3. Эксперимент и результаты

Рассмотрим применение предлагаемого метода аппаратной реализации СНС с использованием СОК на примере архитектуры LeNet-5 [22]. Обучение производилось с помощью библиотек машинного обучения TensorFlow [23] и Keras [24], использовался язык программирования Python. Функция активации гиперболический тангенс (tanh) заменена на ReLU. Параметры архитектуры LeNet-5 представлены в табл. 1.

Табл. 1. Параметры архитектур СНС LeNet-5
Table 1. Parameters of CNN LeNet-5 architecture

Слой	**Размер маски фильтра**	**Количество фильтров**	**Функция активации**
свертка	5×5×3	6	ReLU
max pooling	2×2	1	
свертка	5×5×6	16	ReLU
max pooling	2×2	1	
свертка	5×5×16	120	ReLU
полносвязный	84 нейрона		ReLU
полносвязный	10 нейронов		softmax

Для обучения СНС использовались базы изображений MNIST [22], FMNIST [25] и CIFAR-10 [26]. База MNIST содержит изображения рукописных цифр от 0 до 9 размера 28×28 в оттенках серого и состоит из 60000 изображений для обучения и 10000 изображений для тестирования. База FMNIST содержит 10 классов изображений одежды и обуви (футболка, брюки, свитер, платье, пальто, сандалии, рубашка, кроссовки, сумка, ботинки) размера 28×28 в оттенках серого, состоит из 60000 изображений для обучения и 10000 изображений для тестирования. База CIFAR-10 содержит 10 классов изображений (самолет, автомобиль, птица, кот, олень, собака, лягушка, лошадь, корабль, грузовик) размера 32×32 формата RGB, состоит из 50000 изображений для обучения и 10000 изображений для тестирования.

Весовые коэффициенты СНС являются вещественными числами. Для аппаратной реализации СНС требуется представить их в целочисленном виде. Для этого весовые коэффициенты необходимо умножить на 2^p и округлить результат к большему. После выполнения вычислений, полученный результат масштабируется на 2^{-p} и округляется к меньшему. Коэффициент масштабирования p показывает сколько бит необходимо для представления весовых коэффициентов в устройстве. От выбора p зависит точность работы СНС и объем памяти, необходимый для хранения весовых коэффициентов.

Был проведен эксперимент для выявления зависимости точности распознавания СНС от разрядности весовых коэффициентов, который показал, что разрядность весовых коэффициентов может быть уменьшена без потери точности распознавания. Для архитектуры СНС LeNet-5, обученной на базах MNIST и FMNIST, достаточно разрядности весовых коэффициентов 8 бит, а архитектуре, обученной на базе CIFAR-10, необходимо 12 бит. Таким образом, для архитектуры с 8-битным представлением весовых коэффициентов требуется в 4 раза меньший объем памяти, по сравнению с 32-битным представлением. А для архитектуры с 12-битным представлением весовых коэффициентов будет использоваться в 2,67 раз меньший, по сравнению с 32-битным представлением.

Табл. 2. Результаты аппаратного моделирования СНС LeNet-5
Table 2. Hardware simulation results of CNN LeNet-5

Параметр	Набор данных	Система счисления	
		ПСС	СОК
Тактовая частота, МГц	MNIST и FMNIST	50	56
	CIFAR-10	53	59
Количество LUT	MNIST и FMNIST	593291	647821
	CIFAR-10	612196	557297
Количество LUTRAM	MNIST и FMNIST	483	2212
	CIFAR-10	483	2212
Количество BRAM	MNIST и FMNIST	63,0	181,0
	CIFAR-10	69,5	200,5
Энергопотребление, Вт	MNIST и FMNIST	9,326	12,833
	CIFAR-10	11,718	13,518
Производительность, кадр/с	MNIST и FMNIST	272	305
	CIFAR-10	221	246

Было проведено сравнение предлагаемого метода аппаратной реализации СНС с вычислениями в СОК и традиционной архитектуры СНС в ПСС. Устройство с вычислениями в ПСС является 32-разрядным. Для организации вычислений в СОК был выбран набор модулей $\{2^{12}, 2^{11}-1, 2^{10}-1\}$. Аппаратное моделирование было проведено в среде Xilinx Vivado 2018.3 на целевой плате Virtex-7 xc7v2000tfhg1761-2L со стратегией оптимизации AreaOpimized_high. Результаты аппаратного моделирования представлены в табл. 2. Для оценки эффективности устройств были рассмотрены временные и аппаратные затраты устройств. К временным затратам относится тактовая частота устройства, измеряющаяся в МГц, и производительность, измеряющаяся как количество обработанных кадров в секунду (кадр/с). Для наборов данных MNIST и FMNIST размер кадра составляет 28×28 пикселей, а для набора данных CIFAR-10 размер кадра 32×32 пикселя. Под аппаратными затратами подразумевается количество занятых просмотровых таблиц (Look-Up-Table, LUT), памяти с произвольным доступом (Random Access Memory, RAM) LUTRAM и Block RAM (BRAM), а также энергопотребление устройства, которое измеряется в Вт.

Сравнение и обсуждение результатов аппаратного моделирования представлено в следующем разделе.

4. Обсуждение результатов

В табл. 2 представлены результаты аппаратного моделирования, которые показали, что использование предлагаемого метода на основе СОК позволяет увеличить тактовую частоту и производительность устройства примерно на 11% –12% по сравнению с ПСС. Тем не менее, предлагаемый метод требует больше аппаратных ресурсов. Энергопотребление устройства, разработанного по предлагаемому методу на 15% –38% выше, по сравнению с известным методом на основе ПСС. Кроме того, количество блоков памяти так же увеличилось примерно в 3–4 раза. Устройство на основе предлагаемого метода с 8-битным представлением весовых коэффициентов занимает на 9% больше LUT, но устройство с 12-битным представлением весовых коэффициентов использует на 9% меньше LUT по сравнению с методом на основе ПСС.

Основываясь на результатах аппаратного моделирования, можно сделать вывод, что предлагаемый метод аппаратной реализации СНС с вычислениями в СОК может быть

успешно применен в практических приложениях распознавания визуальных образов, где скорость обработки информации играет ключевую роль. Если требуется минимизировать аппаратные затраты, то традиционный метод на основе ПСС предпочтительнее.

Предложенный в работе метод, может быть адаптирован к другим архитектурам нейронных сетей, что расширяет область его практической значимости.

5. Заключение

В данной работе предложен метод аппаратной реализации СНС с вычислениями в СОК с модулями вида $\{2^{\alpha_1}, 2^{\alpha_2}-1, \ldots, 2^{\alpha_n}-1\}$. Проведено аппаратное моделирование на FPGA на примере СНС LeNet-5 и баз изображений MNIST, FMNIST и CIFAR-10. Результаты моделирования показали, что применение предлагаемого подхода к аппаратной реализации СНС позволяет увеличить тактовую частоту и производительность устройства примерно на 11% –12%, по сравнению с традиционным подходом на основе ПСС. Преимущество в скорости работы устройства достигнуто за счет увеличения аппаратных затрат. Предложенный метод может быть применен в таких практических приложениях как распознавание изображений, анализ речи и создание робототехнических систем.

Список литературы / References

[1] Ashiq F., Asif M. et al. CNN-Based Object Recognition and Tracking System to Assist Visually Impaired People. IEEE Access, vol. 10, 2022, pp. 14819-14834.

[2] Moon C.-I., Lee O. Skin Microstructure Segmentation and Aging Classification Using CNN-Based Models. IEEE Access, vol. 10, 2022, pp. 4948-4956.

[3] Mondal A.K., Bhattacharjee A. et al. xViTCOS: Explainable Vision Transformer Based COVID-19 Screening Using Radiography. IEEE Journal of Translational Engineering in Health and Medicine, vol. 10, 2022, article no. 1100110, pp. 1-10.

[4] Elharrouss O., Almaadeed N. et al. FSC-Set: Counting, Localization of Football Supporters Crowd in the Stadiums. IEEE Access, vol. 10, 2022, pp. 10445-10459.

[5] Vieira J.C., Sartori A. et al. Low-cost CNN for Automatic Violence Recognition on Embedded System. IEEE Access, vol. 10, 2022, pp. 25190-25202.

[6] Wong C.-C., Chien M.-Y. et al. Moving Object Prediction and Grasping System of Robot Manipulator. IEEE Access, vol. 10, 2022, pp. 20159-20172.

[7] Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks. In Proc. of the 25th International Conference on Neural Information Processing Systems, vol. 1, 2012, pp, 1097–1105.

[8] Nakahara H., Sasao T. A deep convolutional neural network based on nested residue number system. In Proc. of the 25th International Conference on Field Programmable Logic and Applications (FPL), 2015, pp. 1-6.

[9] Nakahara H., Sasao T. A High-speed Low-power Deep Neural Network on an FPGA based on the Nested RNS: Applied to an Object Detector. In Proc. of IEEE International Symposium on Circuits and Systems (ISCAS), 2018, pp. 1-5.

[10] Salamat S., Imani M. et al. RNSnet: In-Memory Neural Network Acceleration Using Residue Number System. In Proc. of IEEE International Conference on Rebooting Computing (ICRC), 2018, pp. 1-12.

[11] Omondi A., Premkumar B. Residue Number Systems: Theory and Im-plementation. London, Imperial College Press, 2007. 296 p.

[12] Chervyakov N.I., Lyakhov P.A. et al. Residue Number System-Based Solution for Reducing the Hardware Cost of a Convolutional Neural Network. Neurocomputing, vol. 407, 2020, pp. 439-453.

[13] Parhami B. Computer arithmetic: algorithms and hard-ware designs. Oxford University Press, 2010. 492 p.

[14] Vergos H.T., Dimitrakopoulos G. On Modulo 2^n+1 Adder Design. IEEE Transactions on Computers, vol. 61, issue 2, 2012, pp. 173-186.

[15] Kogge P.M., Stone H.S. A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations. IEEE Transactions on Computers, vol. C-22, issue 8, 1973, pp. 786–793.

[16] Chervyakov N.I., Lyakhov P.A., Valueva M.V. Increasing of Convolutional Neural Network Performance Using Residue Number System. In Proc. of the International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), 2017, pp. 135–140.
[17] Tung C., Huang S.A High-Performance Multiply-Accumulate Unit by Integrating Additions and Accumulations into Partial Product Reduction Process. IEEE Access, vol. 8, 2020, pp. 87367-87377.
[18] Habibi Aghdam H., Jahani Heravi E. Guide to Convolutional Neural Networks. Springer, 2017, 305 p.
[19] Valueva M., Valuev G. et al. Construction of Residue Number System Using Hardware Efficient Diagonal Function. Electronics, vol. 8, issue 6, 2019, article no 694, pp. 1-16.
[20] Chervyakov N.I., Molahosseini A.S. et al. Residue-to-binary conversion for general moduli sets based on approximate Chinese remainder theorem. International Journal of Computer Mathematics, vol. 94, issue 9, 2017, pp. 1833–1849.
[21] Haykin S.S. Neural networks: a comprehensive foundation. Prentice Hall, 1999. 842 p.
[22] LeCun Y., Bottou L. et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE, vol. 86, issue 11, 1998, pp.2278-2324.
[23] Abadi M., Agarwal A. et al. TensorFlow: Large-scale machine learning on heterogeneous systems. arXiv preprint arXiv:1603.04467, 1916, 19 p.
[24] Xiao H., Kashif R., Vollgraf R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747, 2017, 6 p.
[25] Krizhevsky A. Learning Multiple Layers of Features from Tiny Images. Technical Report TR-2009, University of Toronto, 2009, 60 p.

Информация об авторах / Information about authors

Мария Васильевна ВАЛУЕВА получила степени бакалавра и магистра прикладной математики и компьютерных наук в СКФУ в 2016 и 2018 годах соответственно, где в настоящее время работает над кандидатской диссертацией. Работает в Северо-Кавказском центре математических исследований младшим научным сотрудником. Ее исследовательские интересы включают высокопроизводительные вычисления, модулярнуюную арифметику, искусственные нейронные сети и цифровую обработку изображений.

Maria Vasilyevna VALUEVA – received the bachelor's and master's degrees in applied mathematics and computer science from NCFU in 2016 and 2018, respectively, where she is currently pursuing the Ph.D. degree. She works with the North-Caucasus Center for Mathematical Research as a Junior Researcher. Her research interests include high performance computing, modular arithmetic, artificial neural networks, and digital image processing.

Георгий Вячеславович ВАЛУЕВ получил степень магистра прикладных компьютерных наук в СКФУ в 2020 году, где в настоящее время обучается в аспирантуре. Работает в Северо-Кавказском центре математических исследований младшим научным сотрудником. Его исследовательские интересы включают высокопроизводительные вычисления, цифровую обработку изображений искусственные нейронные сети.

Georgii Vyacheslavovich VALUEV received the master's degree in applied computer science from NCFU), in 2020, where he is currently pursuing the Ph.D. degree. He works at the North-Caucasus Center for Mathematical Research as a Junior Researcher. His research interests include high performance computing, digital image processing and artificial neural networks.

Михаил Григорьевич БАБЕНКО – кандидат физико-математических наук. Сфера научных интересов: облачные вычисления, высокопроизводительные вычисления, система остаточных классов, нейронные сети, криптография.

Mikhail BABENKO - Ph.D. in Physics and Mathematics. His research interests include cloud computing, high-performance computing, residue number systems, neural networks, cryptography.

Андрей Николаевич ЧЕРНЫХ получил степень доктора наук в Институте системного программирования им. В.П. Иванникова РАН. Он является профессором Центра научных исследований и высшего образования в Энсенаде, Нижняя Калифорния, Мексика. В научном плане его интересуют многоцелевая оптимизация распределения ресурсов в облачной среде, проблемы безопасности, планирования, эвристики и метаэвристики, интернет вещей и т.д.

Andrei TCHERNYKH received his doctor of science degree at Ivannikov Institute for System Programming of the Russian Academy of Sciences. He is holding a full professor position in computer science at CICESE Research Center, Ensenada, Baja California, Mexico. He is interested in grid and cloud research addressing multi-objective resource optimization, both, theoretical and experimental, security, uncertainty, scheduling, heuristics and meta-heuristics, adaptive resource allocation, and Internet of Things.

Хорхе Марио КОРТЕС МЕНДОСА получил степень бакалавра компьютерных наук в Автономном университете Пуэблы в 2008 г., степень магистра в 2011 г. и степень доктора философии в области компьютерных наук в 2018 году в Исследовательском центре CICESE. Он является членом Национальной системы исследователей Мексики (SNI) с 2020 года. Его основные интересы включают облачные вычисления, балансировку нагрузки, распределенные вычисления и планирование.

Jorge Mario CORTES-MENDOZA is received his Bachelor’s degree in Computer Science from the Autonomous University of Puebla (Benemérita Universidad Autónoma de Puebla, México) in 2008, the Master’s degree in 2011 and the Ph.D. degree in 2018 from CICESE Research Center in Computer Science. He is a member of the National System of Researchers of Mexico (SNI) since 2020. His main interests include cloud computing, load balancing, distributed computing, and scheduling.

DOI: 10.15514/ISPRAS-2022-34(3)-6

Модификация алгоритма обнаружения и локализации ошибки в системе остаточных классов

[1] *А.В. Гладков, ORCID: 0000-0002-9454-7618 <agladkov@ncfu.ru>*
[2] *В.А. Кучуков, ORCID: 0000-0002-1839-2765 <vkuchukov@ncfu.ru>*
[1,3] *М.Г. Бабенко, ORCID: 0000-0001-7066-0061 <mgbabenko@ncfu.ru>*
[3,4,5] *А.Н. Черных, ORCID: 0000-0001-5029-5212 <chernykh@cicese.mx>*
[1] *В.В. Бережной, ORCID: 0000-0002-8327-2735 <vvberezhnoi@ncfu.ru>*
[6] *А.Ю. Дроздов, ORCID: 0000-0001-5607-2749 <alexander.y.drozdov@gmail.com>*

[1] *Северо-Кавказский федеральный университет,*
355017, Россия, г. Ставрополь, ул. Пушкина, 1
[2] *Северо-Кавказский центр математических исследований СКФУ*
355017, Россия, г. Ставрополь, ул. Пушкина, 1
[3] *Институт системного программирования им. В.П. Иванникова РАН,*
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
[4] *Центр научных исследований и высшего образования,*
Мексика, 22860, Нижняя Калифорния, Энсенада, ш. Тихуана-Энсенада, 3918
[5] *Южно-Уральский государственный университет,*
454080, Россия, Челябинск, проспект Ленина, 76
[6] *Московский физико-технический институт,*
141701, Россия, Долгопрудный, Институтский пер., 9

Аннотация. В статье рассмотрена модификация алгоритма обнаружения и локализации ошибки в системе остаточных классов (СОК). Классическая избыточная СОК с одним контрольным основанием позволяет обнаружить ошибку, но не локализовать её. Для локализации одиночной ошибки вводят два контрольных основания. Благодаря накладываемым на основания СОК ограничениям уи разработанному алгоритму дается достичь исправления ошибок при одном контрольном основании, передаваемом по надежному каналу связи. Проведено моделирование классического и предложенного подходов с использованием Verilog на ASIC в среде RTL и физического синтеза Cadence Genus Synthesis Solution Предложенный алгоритм позволяет значительно сократить используемое при аппаратной реализации оборудование, незначительно увеличив время работы. На основе предложенного алгоритма разработана система распределенного хранения данных.

Ключевые слова: избыточная система остаточных классов; обнаружение ошибок; аппаратное моделирование

Для цитирования: Гладков А.В., Кучуков В.А., Бабенко М.Г., Черных А.Н., Бережной В.В., Дроздов А.Ю. Модификация алгоритма обнаружения и локализации ошибки в системе остаточных классов. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 75-88. DOI: 10.15514/ISPRAS–2022–34(3)–6.

Благодарности: Работа выполнена при поддержке Российского научного фонда, проект №19-71-10033.

Modified Error Detection and Localization in the Residue Number System

[1] *A.V. Gladkov, ORCID: 0000-0002-9454-7618 <agladkov@ncfu.ru>*
[2] *V.A. Kuchukov, ORCID: 0000-0002-1839-2765 <vkuchukov@ncfu.ru>*
[1,3] *M.G. Babenko, ORCID: 0000-0001-7066-0061 <mgbabenko@ncfu.ru>*
[3,4,5] *A.N. Tchernykh, ORCID: 0000-0001-5029-5212 <chernykh@cicese.mx>*
[1] *V.V. Berezhnoy, ORCID: 0000-0002-8327-2735 <vvberezhnoi@ncfu.ru>*
[6] *A.Yu. Drozdov,* ORCID: *0000-0001-5607-2749 <alexander.y.drozdov@gmail.com>*

[1] *North-Caucasus Federal University,*
1, Pushkin st., Stavropol, 355017, Russia
[2] *North-Caucasus Center for Mathematical Research NCFU,*
1, Pushkin st., Stavropol, 355017, Russia
[3] *Ivannikov Institute for System Programming of the Russian Academy of Sciences,*
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia
[4] *Centro de Investigación Científica y de Educación Superior,*
3918, Ensenada-Tijuana Highway, Ensenada, 22860, Mexico
[5] *South Ural State University,*
76, Lenin prospekt, Chelyabinsk, 454080, Russia
[6] *Moscow Institute of Physics and Technology,*
9 Institutskiy per., Dolgoprudny, Moscow Region, 141701

Abstract. This article presents the design of the modified error detection and localization algorithm in the Residue Number System (RNS). Classical redundant RNS with one control modulus can detect one error but not localize it. Two control moduli are used to localize a single error. Presented algorithm can achieve an error correction with a single control modulus transmitted over a reliable communication channel. The proposed approach was verified using Verilog on ASIC in RTL and physical synthesis tool Cadence Genus Synthesis Solution. It significantly reduces the area of the hardware implementation increasing the packing density and more efficient use of silicon resource. It slightly increases the running time compared with the classical algorithm. Distributed data storage was developed to study efficiency of the proposed algorithm.

Keywords: redundant residue number system; error detection; hardware simulation

For citation: Gladkov A.V., Kuchukov V.A., Babenko M.G., Tchernykh A.N., Berezhnoy V.V., Drozdov A.Yu. Modified Error Detection and Localization in the Residue Number System. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 75-88 (in Russian). DOI: 10.15514/ISPRAS–2022–34(3)–6

Acknowledgements. This work was supported in part by the Russian Science Foundation, project number 19-71-10033.

1. Введение

Надежное хранение и обработка данных является необходимым условием работоспособности компьютерных систем. Для обеспечения достоверности восстановления информации в литературе предложен ряд подходов: дублирование данных и вычислительных каналов [1], схемы разделения секрета [2], коды обнаружения и коррекции ошибок, такие как коды стирания [3], избыточная система остаточных классов [4, 5]. Все эти коды используют избыточность для обнаружения и исправления ошибок. При этом также важна достоверность арифметической и логической обработки информации.

Дублирование данных. Дублирование является самым простым в реализации и наиболее распространенным методом введения избыточности [6]. Использование дублирования и двойного просчета позволяет за счет сравнения проверить правильность или неправильность вычислений. Тройной просчет за счет мажоритарной обработки уже позволяет исправить

ошибку. Однако такой подход фактически уменьшает производительность вычислительной системы минимум в два раза [7].

Помехоустойчивое кодирование. Теория помехоустойчивого кодирования для каждого конкретного канала позволяет выбрать наиболее эффективный метод обнаружения и исправления ошибок. Существуют два взаимодополняющих метода борьбы с помехами [8].

- Кодирование для исправления ошибок – приемник обнаруживает и исправляет ошибки;
- Кодирование для обнаружения ошибок – приемник распознает ошибки и, в случае необходимости, производит запрос на повторную передачу ошибочного блока.

Среди корректирующих кодов наибольшее распространение получили блочные двоичные коды, т.е. передача двоичного сообщения производится блоками, причем каждый блок содержит двоичных символов. Кодирование и декодирование каждого блока производится независимо от других блоков [8]. Коды стирания преобразуют сообщение из k символов в сообщение из n символов, $k < n$, что исходное сообщение может быть восстановлено по любым k' символам. Простейшим таким кодов является код проверки на четность.

В общем случае коды обнаружения и коррекции ошибок содержат две группы цифр – информационную и контрольную. Однако неарифметичность позиционных кодов не позволяет контролировать результаты арифметических операций контрольных оснований [7], таким образом присутствует неравноправность информационной и контрольной частей кода. Непозиционных системы счисления, такие как система остаточных классов, лишены этого недостатка.

Далее в разд. 2 рассмотрены особенности обнаружения и коррекции ошибок в избыточной системе остаточных классов. В разд. 3 предложена модификация алгоритма обнаружения и исправления ошибки в системе остаточных классов с одним избыточным основанием. В разд. 4 приведены результаты моделирования рассматриваемых алгоритмов на ASIC. В разд. 5 предложен вариант применения алгоритма для системы надежного распределенного хранения. В разд. 6 представлены основные результаты работы и направления дальнейшей работы.

2. Избыточная система остаточных классов

Пусть задана система остаточных классов (СОК) с взаимно простыми основаниями $p_1, p_2, \dots, p_n$.

Тогда рабочим диапазоном системы будет $P = p_1 \cdot p_2 \cdot \dots \cdot p_n$.

Если основания удовлетворяют условию $p_1 < p_2 < \dots < p_n$, то система называется упорядоченной.

Чтобы обеспечить возможность обнаружения и локализации ошибок вводят избыточные основания $p_{n+1}, \dots, p_{n+k}$.

Полным диапазоном СОК будет $\overline{P} = P \cdot p_{n+1} \cdot \dots \cdot p_{n+k}$. В данном случае критерием корректности числа $A = (\alpha_1, \alpha_2, \dots, \alpha_n, \alpha_{n+1})$ будет выполнение условия $A < P$, в противном случае число содержит ошибку.

Для избыточной СОК используют обозначение (k, n), где k – количество рабочих оснований, n – общее количество оснований.

Рассмотрим в общем случае СОК с одним избыточным основанием $\mathrm{p_{n+1}}$, для которой $p_i < p_{n+1}$, $i = 1,2, \dots, n$. Тогда любое искажение цифры по одному какому-либо разряду превращает это число в неправильное и позволяет тем самым обнаружить наличие ошибки [7]. При этом накладывают ограничения, что ошибка возникает только по информационным основаниям.

Одним из методов локализации ошибок модулярного кода является метод проекций. Проекцией A_i числа $A = (\alpha_1, \alpha_2, \dots, \alpha_{n+1})$ по основанию p_i будет число, полученное вычеркиванием цифры α_i в представлении A.

Теорема 1. Если в упорядоченной системе остаточных классов проекция A_i числа $A = (\alpha_1, \alpha_2, \dots, \alpha_i, \dots \alpha_n, \alpha_{n+1})$ по основанию p_i удовлетворяет условию

$$A_i > \frac{\overline{P}}{p_{n+1}},$$

то цифра α_i правильная, если возможна лишь одиночная ошибка [7].

Введение только одного контрольного основания в общем случае не позволяет локализовать ошибку. Для коррекции ошибки можно использовать метод на основе Китайской теоремы об остатках (КТО) и методе проекций. Запишем его в виде алгоритма:

Алгоритм 1. Восстановление числа на основе метода проекций и КТО.

Input: $X' = (x'_1, x'_2, \dots, x'_n, x'_{n+1})$,

$p_1, p_2, \dots, p_{n+1}$, $P = \prod_{i=1}^{n} p_i$, $\overline{P} = p_n \cdot P$

$w_i = \left|\overline{P}_i^{-1}\right|_{p_i} \cdot \overline{P}_i$, где $\overline{P}_i = \overline{P}/p_i$, для всех $i \in [1, n+1]$.

$w_{i,j} = \left|\overline{P}_{i,j}^{-1}\right|_{p_j} \cdot \overline{P}_{i,j}$, где $\overline{P}_{i,j} = \overline{P}_i/p_j$, для всех $i \in [1, n+1]$ и $j \neq i$.

Output: X

```
1. S=0
2. for i = 1 to n + 1
   2.1. S = S + x'_i · w_i
3. S = S mod P̄
4. if S < P
   4.1. X = S
   4.2. return X
5. else
   5.1. for i = 1 to n + 1
      5.1.1. S_i = 0
      5.1.2. for j = 1 to n + 1
         5.1.2.1. if i ≠ j then
            5.1.2.1.1. S_i = S_i + x'_i · w_{i,j}
      5.1.3.   X = S_i mod P̄_i
      5.1.4.  if X < P then
         5.1.4.1. return X
```

Рассмотрим пример некорректной локализации ошибки. Пусть задана СОК {3,5,7,11,13} с четырьмя информационными основаниями и одним контрольным. Тогда рабочим диапазоном будет $P = 1155$, полный диапазон $\overline{P} = 15015$.

Возьмем число $X = 4 = (1,4,4,4,4)$ и введем ошибку по второму основанию, получим $X' = (1,0,4,4,4) = 6010$. Поскольку $X' = 6010 > P = 1155$ можно сделать вывод, что X' содержит ошибку. Тогда проекции $X'_1 = 1005 = (0,4,4,4)$, $X'_2 = 4 = (1,4,4,4)$, $X'_3 = 1720 = (1,0,4,4)$, $X'_4 = 550 = (1,0,4,4)$, $X'_5 = 235 = (1,0,4,4)$. Очевидно, что проекции X'_1, X'_2, X'_4, X'_5 удовлетворяют условию корректности, откуда следует что одно избыточное основание позволяет только обнаружить ошибку, но не локализовать её.

Для обеспечения возможности коррекции ошибок введем два контрольных основания p_{n+1}, p_{n+2}.

Теорема 2. Если в системе $p_1, p_2, \dots, p_n, p_{n+1}, p_{n+2}$ с двумя контрольными основаниями задано неправильное число $A' = (\alpha'_1, \alpha'_2, \dots, \alpha'_i, \dots, \alpha'_n, \alpha'_{n+1}, \alpha'_{n+2})$, то необходимым и достаточным условием ошибочности цифры α'_i в A' является правильности его проекции A'_i по основанию p_i [7].

Рассмотрим аналогичный пример с двумя контрольными основаниями. Возьмем СОК $\{3,5,7,11,13,17\}$, рабочий диапазон у которой будет $P = 1155$, а полный диапазон $\overline{P} = 255255$. Возьмем число $X = 4 = (1,4,4,4,4,4)$ и введем ошибку по второму основанию, получим $X' = (1,0,4,4,4,4) = 51055$. Поскольку $X' = 51055 > P = 1155$ можно сделать вывод, что X' содержит ошибку. Тогда проекции $X'_1 = 51055$, $X'_2 = 4$, $X'_3 = 14590$, $X'_4 = 4645$, $X'_5 = 11785$, $X'_6 = 6010$. Очевидно, что только $X'_2 = 4 < P$, значит ошибка произошла по второму основанию и исходное число равно 4.

Однако недостатком исправления ошибки с использованием двух контрольных оснований является сложность восстановления чисел с на основе КТО по каждой проекции. Для решения этой проблемы рассмотрим модификацию алгоритма обнаружения и коррекции ошибок.

3. Модификация алгоритма обнаружения и локализации ошибки в системе остаточных классов

Пусть задана система остаточных классов с одним контрольным основанием $p_1 < p_2 < p_3 < \cdots < p_n < p_{n+1}$, при этом считается, что контрольное основание надежное и не может содержать ошибки. Тогда введем алгоритм

Алгоритм 2. Восстановление числа на основе метода проекций и КТО.

Input: $X' = (x'_1, x'_2, \dots, x'_n, x'_{n+1})$,

$p_1, p_2, \dots, p_{n+1}$, $P = \prod_{i=1}^{n} p_i$, $\overline{P} = p_n \cdot P$

$\overline{P}_i = \overline{P}/p_i, i \in [1, n+1]$.

$w_i = \left|\overline{P}_i^{-1}\right|_{p_i}$.

Output: X

1. S=0
2. for $i = 1$ to $n+1$
   2.1. $S = S + x'_i \cdot w_i \cdot \overline{P}_i$
3. $X = S \bmod \overline{P}$
4. if $X < P$
   4.1. return X
5. else
   5.1. $k = 1$
   5.2. $X = S \bmod \overline{P}_1$
   5.3. while $X > P$ AND $k \le n$ do
      5.3.1. $k = k + 1$
      5.3.2. $X = S \bmod \overline{P}_k$
   5.4. return X

Теорема 3. Если $p_{n+1} > p_n \cdot p_{n-1}$, то алгоритм 2 корректен.

Доказательство

Пусть ошибка произошла по k-му основанию, где $k \in [1, n]$. Для доказательства теоремы достаточно показать, что для всех $X \in [0, P)$, $j \in [1, n]$ и $j \neq k$, выполняется следующее неравенство:

$$P \le |X + e_k \cdot w_k \cdot \bar{P}_k|_{\bar{P}_j} \tag{1}$$

Вычислим $|X + e_k \cdot w_k \cdot \bar{P}_k|_{\bar{P}_j}$ используя формулу $|a|_m = a - \left\lfloor \frac{a}{m} \right\rfloor \cdot m$, получим:

$$|X + e_k \cdot w_k \cdot \bar{P}_k|_{\bar{P}_j} = X + e_k \cdot w_k \cdot \bar{P}_k - \left\lfloor \frac{X + e_k \cdot w_k \cdot \bar{P}_k}{\bar{P}_j} \right\rfloor \cdot \bar{P}_j \tag{2}$$

Так как $\frac{\bar{P}_k}{\bar{P}_j} = \frac{p_j}{p_k}$, то представим выражение $\left\lfloor \frac{X+e_k \cdot w_k \cdot \bar{P}_k}{\bar{P}_j} \right\rfloor$ в, следующем, виде:

$$\left\lfloor \frac{X + e_k \cdot w_k \cdot \bar{P}_k}{\bar{P}_j} \right\rfloor = \left\lfloor \frac{X}{\bar{P}_j} + \frac{e_k \cdot w_k \cdot \bar{P}_k}{\bar{P}_j} \right\rfloor = \left\lfloor \frac{X}{\bar{P}_j} + \frac{e_k \cdot w_k \cdot p_j}{p_k} \right\rfloor \quad (3)$$

Оценим значение величины $\frac{X}{\bar{P}_j}$ для всех $j \in [1, n]$ и $j \neq k$, получим

$$\frac{X}{\bar{P}_j} \leq \frac{P-1}{\bar{P}_j} = \frac{P}{\bar{P}_j} - \frac{1}{\bar{P}_j} \quad (4)$$

Так как $\frac{P}{\bar{P}_j} = \frac{p_j}{p_{n+1}}$ и по условию теоремы $p_{n+1} > p_n \cdot p_{n-1}$, то $\frac{p_j}{p_{n+1}} < \frac{1}{p_k}$ для всех $j \in [1, n]$ и $j \neq k$, следовательно, Eq. (4) примет вид:

$$\frac{X}{\bar{P}_j} < \frac{1}{p_k} \quad (5)$$

Учитывая неравенство (5), выражение (3) примет вид:

$$\left\lfloor \frac{X + e_k \cdot w_k \cdot \bar{P}_k}{\bar{P}_j} \right\rfloor = \left\lfloor \frac{e_k \cdot w_k \cdot p_j}{p_k} \right\rfloor \quad (6)$$

Подставляя выражения (2) и (6) в неравенство (1), получим:

$$P \leq X + e_k \cdot w_k \cdot \bar{P}_k - \left\lfloor \frac{e_k \cdot w_k \cdot p_j}{p_k} \right\rfloor \cdot \bar{P}_j \quad (7)$$

Разделим левую часть неравенства (7) на положительное число P, получим:

$$1 \leq X + e_k \cdot w_k \cdot \frac{\bar{P}_k}{P} - \left\lfloor \frac{e_k \cdot w_k \cdot p_j}{p_k} \right\rfloor \cdot \frac{\bar{P}_j}{P} \quad (8)$$

Так как $\frac{\bar{P}_k}{P} = \frac{p_{n+1}}{p_k}$ и $\frac{\bar{P}_j}{P} = \frac{p_{n+1}}{p_j}$, то выражение (8) примет вид:

$$1 \leq X + e_k \cdot w_k \cdot \frac{p_{n+1}}{p_k} - \left\lfloor \frac{e_k \cdot w_k \cdot p_j}{p_k} \right\rfloor \cdot \frac{p_{n+1}}{p_j} \quad (9)$$

Умножим правую и левую часть неравенства (7) на положительное число $\frac{p_k \cdot p_j}{p_{n+1}}$, получим:

$$\frac{p_k \cdot p_j}{p_{n+1}} \leq X \cdot \frac{p_k \cdot p_j}{p_{n+1}} + e_k \cdot w_k \cdot p_j - \left\lfloor \frac{e_k \cdot w_k \cdot p_j}{p_k} \right\rfloor \cdot p_k \quad (10)$$

Так как

$$e_k \cdot w_k \cdot p_j - \left\lfloor \frac{e_k \cdot w_k \cdot p_j}{p_k} \right\rfloor \cdot p_k = \left| e_k \cdot w_k \cdot p_j \right|_{p_k},$$

то формула (10) примет вид:

$$\frac{p_k \cdot p_j}{p_{n+1}} \leq X \cdot \frac{p_k \cdot p_j}{p_{n+1}} + \left| e_k \cdot w_k \cdot p_j \right|_{p_k} \quad (11)$$

Учитывая, что $X \in [0, P-1)$ и $1 \leq \left| e_k \cdot w_k \cdot p_j \right|_{p_k} \leq p_k - 1$, то неравенство (11) выполняется если выполняется для всех $j \in [1, n]$ и $j \neq k$, следующее, условие:

$$\frac{p_k \cdot p_j}{p_{n+1}} \leq 1 \quad (12)$$

Так как модули СОК удовлетворяют условию $p_1 < p_2 < p_3 < \cdots < p_n$, то из неравенства (12), следует, что необходимым и достаточным условием выполнения неравенство (1), является: $p_{n+1} \geq p_n p_{n-1}$.

Теорема доказана.

Рассмотрим пример обнаружения и исправления ошибки в СОК с одним контрольным основанием.

Выберем систему оснований СОК, удовлетворяющую теореме 3: $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7, n = 4$. Контрольное основание $p_5 = 37 > 7 \cdot 5$. Искомое число $X = 53 = (1,2,3,4,16)$.

Параметры СОК:

$P = 2 \cdot 3 \cdot 5 \cdot 7 = 210$ — рабочий диапазон.

$\overline{P} = p_5 \cdot P = 35 \cdot 210 = 7770$ — полный диапазон СОК.

$\overline{P}_1 = \frac{\overline{P}}{p_1} = 3885, \overline{P}_2 = \frac{\overline{P}}{p_2} = 2590,$

$\overline{P}_3 = \frac{\overline{P}}{p_3} = 1554, \overline{P}_4 = \frac{\overline{P}}{p_4} = 1110,$

$\overline{P}_5 = \frac{\overline{P}}{p_5} = 210.$

$w_1 = \left|\overline{P}_1^{-1}\right|_{p_1} = 1, w_2 = \left|\overline{P}_2^{-1}\right|_{p_2} = 1,$

$w_3 = \left|\overline{P}_3^{-1}\right|_{p_3} = 4, w_4 = \left|\overline{P}_4^{-1}\right|_{p_4} = 2,$

$w_5 = \left|\overline{P}_5^{-1}\right|_{p_5} = 3.$

Ввведем вектор ошибки $E = (0,0,1,0,0)$, $X' = E + X = (1,2,4,\ 4,16)$.

Вычислим

$$S = \left|\sum_{i=1}^{n+1} w_i \overline{P}_i x_i'\right|_{\overline{P}} =$$

$$= |1 \cdot 3885 \cdot 1 + 1 \cdot\ 2590 \cdot 2 + 4 \cdot 1554 \cdot 4 + 2 \cdot 1110 \cdot 4 + 3 \cdot\ 210 \cdot 16|_{7770} = 6269.$$

Так как $S = 6269 > 210$, следовательно, есть ошибка, вычислим ее:

$|S|_{\overline{P}_1} = 2384 > P$, $|S|_{\overline{P}_2} = 1089 > P$, $|S|_{\overline{P}_3} = 53 < P$, следовательно $X = 53$.

4. Моделирование и анализ

Для анализа и моделирования были выбраны наборы модулей СОК с 4, 5, 6 рабочими основаниями, рабочий диапазон которых покрывает диапазон в 8, 16, 24 и 32 бита

Избыточность данных является важным вопросом. В избыточной (k, n) СОК, используя любые k остатков из n, мы можем восстановить данные. Тогда избыточность можно выразить выражением

$$\frac{\sum_{i=1}^{n} d_i}{\sum_{i=1}^{k} d_i} - 1,$$

где d_i размерность остатков по основанию p_i.

Для моделирования работы алгоритма 1 были выбраны наборы с двумя контрольными основаниями и вычислена их избыточность. Результаты представлены в табл. 1.

Табл. 1. Избыточность ИСОК с двумя контрольными основаниями
Table 1. The redundancy of the RRNS with two control moduli.

Набор оснований	Размерность, бит	Избыточность
{3, 5, 7, 11, 13, 17}	10	0.75
{13, 17, 19, 23, 29, 31}	16	0.53
{59, 61, 67, 71, 73, 79}	24	0.54
{251, 257, 263, 269, 271, 277}	32	0.51
{2, 3, 5, 7, 11, 13, 17}	11	0.69
{5, 7, 11, 13, 17, 19, 23}	16	0.53
{23, 29, 31, 37, 41, 43, 47}	24	0.44
{79, 83, 89, 97, 101, 103, 107}	32	0.40
{2, 3, 5, 7, 11, 13, 17, 19}	14	0.59
{3, 5, 7, 11, 13, 17, 19, 23}	17	0.48
{11, 13, 17, 19, 23, 29, 31, 37}	24	0.39
{31,37, 41, 43, 47, 53, 59, 61}	32	0.34

Аналогично для моделирования работы алгоритма 2 были выбраны наборы с одним контрольным основанием, удовлетворяющие условиям теоремы 3 и вычислена их избыточность. Результаты представлены в табл. 2.

Табл. 2. Избыточность ИСОК с двумя контрольными основаниями
Table 2. The redundancy of the RRNS with one control modulus.

Набор оснований	Рабочий диапазон, бит	Избыточность
{3, 5, 7, 11, 79}	10	0.58
{13, 17, 19, 23, 439}	16	0.47
{59, 61, 67, 71, 4759}	24	0.50
{251, 257, 263, 269, 70753}	32	0.49
{2, 3, 5, 7, 11, 79}	11	0.54
{5, 7, 11, 13, 17, 223}	16	0.42
{23, 29, 31, 37, 41, 1523}	24	0.41
{79, 83, 89, 97, 101, 9803}	32	0.40
{2, 3, 5, 7, 11, 13, 149}	14	0.47
{3, 5, 7, 11, 13, 17, 223}	17	0.38
{11, 13, 17, 19, 23, 29, 673}	24	0.36
{31, 37, 41, 43, 47, 53, 2503}	32	0.34

В среднем избыточность предложенного алгоритма на 14% меньше, чем у СОК с двумя контрольными основаниями.

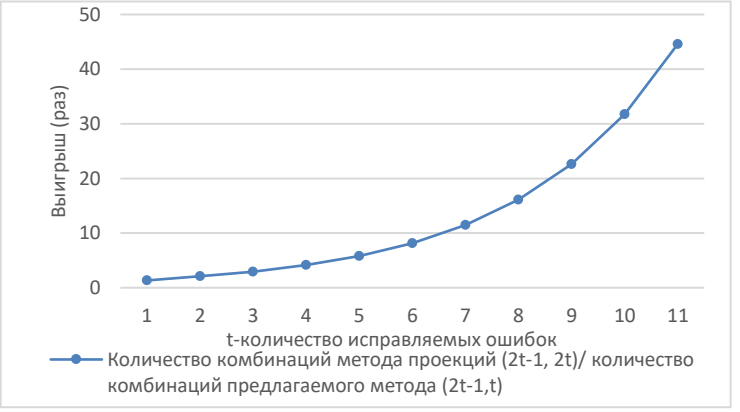


Рис. 1. Анализ количества комбинаций предлагаемого метода по сравнению с методом проекций
Fig. 1. The ratio of the number of combinations of the projection method to the number of combinations of the proposed method.

Метод проекций, представленный алгоритмом 1 требует большого количество восстановлений чисел с использованием КТО для каждой проекции. На рис. 1 показано уменьшение количества комбинаций предлагаемого метода по сравнению с методом проекций.

Моделирование предложенных алгоритмов было реализовано на языке Verilog на ASIC в среде RTL и физического синтеза Cadence Genus Synthesis Solution с использованием библиотеки osu018_stdcells.

Все значения, которые могли быть предвычислены записаны в константы. В качестве критериев, по которым проводился анализ, были выбраны площадь (Cell Area) и задержка комбинаторного пути (Arrival). Также был получен показатель количества используемых ячеек (Cell Count), однако он связан с площадью, но в то же время при одинаковом количестве ячеек общая площадь может быть различной ввиду различной сложности ячеек используемой библиотеки.

Результаты моделирования представлены в таблице 3. В среднем предложенный алгоритм 2 выполняется среднем в 1,5 раза дольше за счет последовательности действий 3 и 5 алгоритма, но при этом площадь микросхемы в среднем в 3,5 раза меньше. Это связано с ресурсоемкими вычислениями проекций, которые, однако, могут быть вычислены параллельно. При этом стоит обратить внимание на 32-битный рабочий диапазон. В этом случае для 5 и 6 рабочих основаниях время прохождения сигнала примерно одинаковое, а площадь реализации алгоритма 1 в 6 раз больше.

Табл. 3. Результаты моделирования на ASIC
Table 3. The results of the simulation

Показатель	**Количество рабочих модулей**	**Алгоритм**	**Покрываемый рабочий диапазон, бит**			
			8	**16**	**24**	**32**
Время, пс	4	1	15447	22338	34747	51685
		2	25097	39429	67242	76830
	5	1	15383	23642	30307	51164
		2	21944	36698	59120	44727
	6	1	19083	23047	31911	48572
		2	32558	41193	53634	48738
Площадь	4	1	156978	320430	652211	956196
		2	60759	132332	206485	280213
	5	1	184405	335056	657195	1006255
		2	63763	131767	154479	166433
	6	1	302090	411108	652668	1011104
		2	96815	155115	187973	184302

5. Система распределенного хранения данных

Одним из возможных применений данного алгоритма может быть система распределенного хранения данных, в которой выполняется подготовка исходных файлов для надежного распределенного хранения, посредством перевода в систему остаточных классов, удовлетворяющую теореме 3 и для восстановления полученных файлов, принятых из распределенной среды в случае ошибки или неполучения одной из частей файла по алгоритму 2.

Существует большое количество систем хранения данных, описанных в патентах США, России, Китая [заявка WO2019199288, опубл. 17.10.2019], [заявка US2013173916, опубл. 04.07.2013], [патент RU2656836, опубл. 06.06.2018], [патент CN103957264, опубл. 30.07.2014].

Предложенная система может быть реализована схемой, представленой на рис. 2. Исходное значение X поступает на входы блоков $mod\, p_i$ нахождения остатков по модулю p_i, $i \in$

$[1, n+1]$, которые могут быть выполнены как с использованием вычислительных устройств, например, интегральных схем или FPGA, так и в виде памяти, которая в ответ на значение исходного числа X подает на выход блока $mod\, p_i$ остаток x_i от деления на модуль p_i. При этом модули удовлетворяют условиям $p_1 < p_2 < p_3 < \cdots < p_n < p_{n+1}$ и $p_{n+1} > p_n \cdot p_{n-1}$.

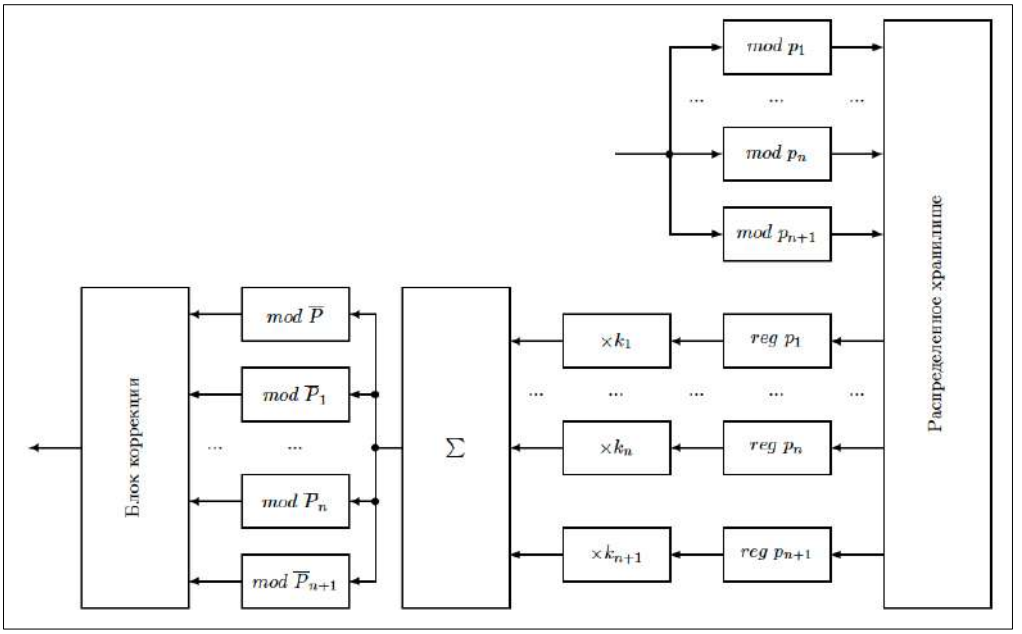


Рис. 2. Схема системы распределенного хранения данных
Fig. 2. Block diagram of the distributed data storage system

Данные с выходов блоков нахождения остатков по модулю p_i передаются в распределенное хранилище, которая может быть использована как для хранения, так и обработки данных, поскольку особенностью системы остаточных классов является возможность выполнения арифметических операций сложения и умножения независимо по каждому модулю. При этом распределенное хранилище может быть представлено одним или несколькими облачными провайдерами, или внутренними частными облаками/хранилищами, что позволит распределить данные при хранении, тем самым повысить надежность восстановления данных в случае выхода из строя одного или нескольких хранилищ за счет избыточной структуры системы остаточных классов.

После хранения данные поступают из распределенного хранилища на входы регистров хранения остатков по модулю p_i, выходы которых соединены со входами соответствующих блоков умножения на $k_i = w_i \overline{P_i}$, выходы которых подключены к входами сумматора произведений, значение суммы поступает на входы блока нахождения остатков по модулю $\overline{P}$ и блоков нахождения остатков по модулю $\overline{P_i}$, результаты с которых поступают на вход блока коррекции ошибки, выход которого является выходом системы.

На рис. 3 показана структурная схема блока коррекции ошибки. Значение $|S|_{\bar{P}}$ поступает на блок сравнения $|\mathrm{T}|_{\bar{P}}$ с рабочим диапазоном, в котором проверяется логическое выражение $|S|_{\bar{P}} < P$, если оно истинно, то сигнал с выхода блок сравнения $|S|_{\bar{P}}$ с рабочим диапазоном поступает на управляющий вход мультиплексора, пропуская корректное значение $|S|_{\bar{P}}$ через первый информационный вход на выход блока коррекции ошибки. Второй информационный вход мультиплексора $|S|_{\bar{P}}$ подключен к выходу мультиплексора $|S|_{\bar{P}_1}$.

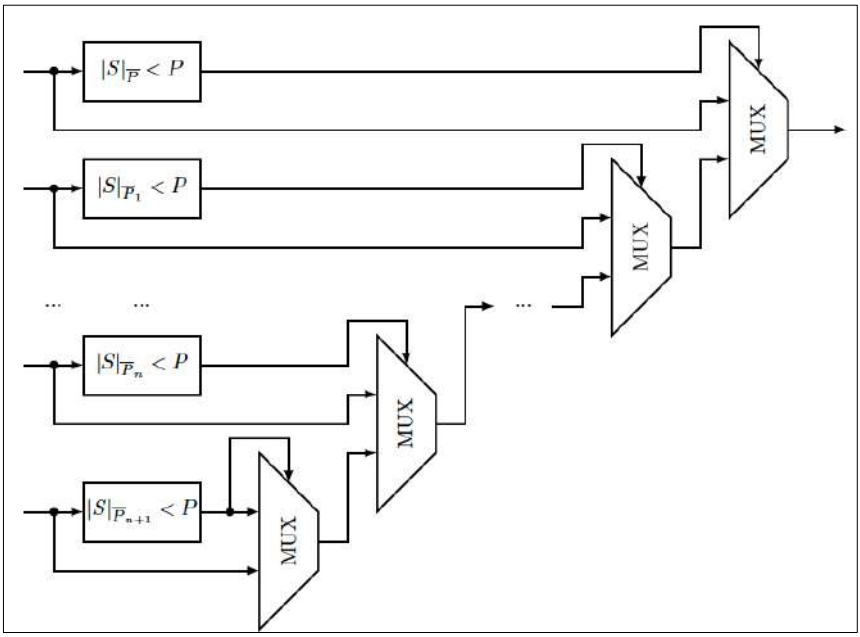


Рис. 3. Структурная схема блока коррекции
Fig. 3. Block diagram of the correction unit

Значения $|S|_{\bar{P}_i}$ с выходов блоков нахождения остатков по модулю $\bar{P}_i$ поступают на входы соответствующих блоков сравнения $|S|_{\bar{P}_i}$ с рабочим диапазоном, в которых проверяются логические выражения $|S|_{\bar{P}_i} < P$, если оно не выполняется, т.е. равно 0, то сигнал 0 с выхода блок сравнения $|S|_{\bar{P}_i}$ с рабочим диапазоном поступает на управляющий вход мультиплексора $|S|_{\bar{P}_i}$, на первый информационный вход которого подается значения $|S|_{\bar{P}_i}$ с выхода блока нахождения остатков по модулю $\bar{P}_i$. Выход мультиплексора $|S|_{\bar{P}_i}$ подключен ко второму информационному входу мультиплексора $|S|_{\bar{P}_{i-1}}$. Второй информационный вход мультиплексора $|S|_{\bar{P}_{n+1}}$ подключен к выходу блок сравнения $|S|_{\bar{P}_{n+1}}$.

Блок коррекции ошибки осуществляет вывод первого значения $|S|_{\bar{P}}$ или $|S|_{\bar{P}_i}$, которое меньше рабочего диапазона.

6. Заключение

Преимуществом данного алгоритма является снижение аппаратных издержек коррекции ошибок модулярных чисел, полученных из систем распределенного хранения данных, что связано с отсутствием необходимости вычисления $S = \sum_{i=1}^{n+1} k_i x_i'$ для каждой проекции $\bar{P}_i$, S вычисляется один раз, и в дальнейшем необходимо вычисления только остатка от деления.

Реализация всей системы возможна с использованием программируемых логических интегральных схем (ПЛИС), специализированных интегральных схем, а также в виде алгоритма работы ЭВМ и может использоваться как отдельное устройство, так и как сопроцессор для выполнения подготовки файлов к надежному распределенному хранению данных.

При этом предложенный алгоритм в среднем выполняется в 1,5 раза дольше за счет последовательной архитектуры некоторых операций, но при этом площадь микросхемы в среднем в 3,5 раза меньше.

При этом стоит обратить внимание на 32-битный рабочий диапазон. В этом случае для 5 и 6 рабочих основаниях время прохождения сигнала примерно одинаковое, а площадь реализации алгоритма 1 в 6 раз больше.

Направлением дальнейших исследований может стать реализация предложенного алгоритма для криптографических алгоритмов с размерностью 64, 128, 256 бит.

Список литературы / References

[1]. Ghemawat S., Gobioff H., Leung S.T. The Google file system. In Proc. of the 18th ACM Symposium on Operating Systems Principles, 2003, pp. 29-43.

[2]. Gomathisankaran M., Tyagi A., Namuduri K. HORNS: A homomorphic encryption scheme for Cloud Computing using Residue Number System. In Proc. of the 45th Annual Conference on Information Sciences and Systems, 2011, pp. 1-5.

[3]. Lin H.Y., Tzeng W.G. A secure erasure code-based cloud storage system with secure data forwarding. IEEE transactions on parallel and distributed systems, vol. 23, issue 6, 2011, pp. 995-1003.

[4]. Celesti A., Fazio M. et al. Adding long-term availability, obfuscation, and encryption to multi-cloud storage systems. Journal of Network and Computer Applications. vol. 59, 2016, pp. 208-218.

[5]. Chervyakov N., Babenko M. et al. AR-RRNS: Configurable reliable distributed data storage systems for Internet of Things to ensure security. Future Generation Computer Systems, vol. 92, 2019, pp. 1080–1092.

[6]. Li W., Yang Y., Yuan D. A Novel Cost-Effective Dynamic Data Replication Strategy for Reliability in Cloud Data Centres. In Proc. of the IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, 2011, pp. 496–502.

[7]. Акушский И.Я., Юдицкий Д.И. Машинная арифметика в остаточных классах. М., Советское радио, 1968 г., 440 стр. / Akushsky I.Ya., Yuditsky D.I. Computer arithmetic in residual classes. Moscow, Soviet Radio, 1968, 440 p. (in Russian).

[8]. Вернер М. Основы кодирования. М., Техносфера, 2004 г., 288 стр. / Werner M. Information und Codierung. Vieweg+Teubner Verlag, Wiesbaden, 2002, 213 p.

Информация об авторах / Information about authors

Андрей Владимирович ГЛАДКОВ – младший научный сотрудник. Сфера научных интересов: высокопроизводительные вычисления, система остаточных классов.

Andrei Vladimirovich GLADKOV – Research Assistant. His research interests include high-performance computing, residue number systems.

Виктор Андреевич КУЧУКОВ – младший научный сотрудник. Сфера научных интересов: высокопроизводительные вычисления, система остаточных классов, нейронные сети, цифровая обработка сигналов.

Viktor Andreevich KUCHUKOV – Research Assistant. His research interests include high-performance computing, residue number systems, neural networks, digital signal processing.

Михаил Григорьевич БАБЕНКО – кандидат физико-математических наук. Сфера научных интересов: облачные вычисления, высокопроизводительные вычисления, система остаточных классов, нейронные сети, криптография.

Mikhail Grigoryevich BABENKO - PhD in Physics and Mathematics. His research interests include cloud computing, high-performance computing, residue number systems, neural networks, cryptography.

Андрей Николаевич ЧЕРНЫХ получил степень доктора наук в Институте системного программирования РАН. Он является профессором Центра научных исследований и высшего образования в Энсенаде, Нижняя Калифорния, Мексика. В научном плане его интересуют многоцелевая оптимизация распределения ресурсов в облачной среде, проблемы безопасности, планирования, эвристики и метаэвристики, интернет вещей и т.д.

Andrei Nikolaevitch TCHERNYKH received his PhD degree at Ivannikov Institute for System Programming of the Russian Academy of Sciences. He is holding a full professor position in computer science at CICESE Research Center, Ensenada, Baja California, Mexico. He is interesting in grid and cloud research addressing multiobjective resource optimization, both, theoretical and experimental, security, uncertainty, scheduling, heuristics and meta-heuristics, adaptive resource allocation, and Internet of Things.

Виктор Васильевич БЕРЕЖНОЙ – кандидат технических наук, доцент. Сфера научных интересов: модулярная арифметика, параллельные вычислительные системы, нейронные сети.

Victor Vasilyevich BEREZHNOY – PhD in Technical Sciences, Associate Professor. Research interests: modular arithmetic, parallel computing systems, neural networks.

Александр Юльевич ДРОЗДОВ. Доктор технических наук, профессор. Главный научный сотрудник, руководитель лаборатории, заместитель заведующего кафедр РЭПИ, ФРТК МФТИ. Основатель и руководитель лаборатории моделирования и проектирования архитектур специальных вычислительных систем МФТИ. Основатель и руководитель конструкторского цента Микроэлектроники факультета радиотехники и кибернетики МФТИ. Научные интересы: системы автоматизации проектирования, вычислительные машины, комплексы и компьютерные сети, системы автоматизации проектирования, принципы работы современных архитектур микропроцессоров, компилятора и системного ПО.

Alexander Yulievich DROZDOV. Doctor of Technical Sciences, Professor. Chief Researcher, Head of the Laboratory, Deputy Head of the Departments of REPI, FRTK MIPT. Founder and head of the Laboratory for Modeling and Designing Architectures of Special Computing Systems at MIPT. Founder and head of the Microelectronics Design Center of the Faculty of Radio Engineering and Cybernetics of the Moscow Institute of Physics and Technology. Scientific interests: design automation systems, computers, complexes and computer networks, design automation systems, operating principles of modern microprocessor architectures, compiler and system software.

DOI: 10.15514/ISPRAS-2022-34(3)-7

Mobile Learning Platform focused on Learning Monitoring and Customization: Usability Evaluation Based on a Laboratory Study

[1] *H. del Ángel-Flores, ORCID: 0000-0001-8035-9301 <hdelangel.mca19@lania.edu.mx>*
[2] *E. López-Domínguez, ORCID: 0000-0002-6167-6309 <eduardo.lopez.dom@cinvestav.mx>*
[3] *Y. Hernández-Velázquez, ORCID: 0000-0002-5767-532X <yeseniahv@gmail.com>*
[1] *S. Domínguez-Isidro, ORCID: 0000-0002-9546-8233 <saul.dominguez@lania.edu.mx>*
[4] *M.A. Medina-Nieto, ORCID: 0000-0001-6391-4799 <maria.medina@uppuebla.edu.mx>*
[4] *J. de la Calleja, ORCID: 0000-0002-6846-3162 <jorge.delacalleja@uppuebla.edu.mx>*

[1] *Laboratorio Nacional de Informática Avanzada,*
Veracruz, México
[2] *Center for Research and Advanced Studies of the National Polytechnic Institute,*
CDMX, Mexico
[3] *Universidad Veracruzana,*
Veracruz, México
[4] *Polytechnic University of Puebla,*
Puebla, México

Abstract. The learning customization and monitoring are considered key aspects of the teaching-learning processes. Some works have proposed mobile learning systems that provide teachers and students learning monitoring and personalization services. One of the main requirements of these kinds of systems in terms of software quality is usability; however, few works have addressed the usability issues using laboratory studies with users in real domains. In this work, we present a usability evaluation of the learning monitoring and personalization services of a mobile learning platform based on a laboratory study in which nine teachers and ten students participated. In our usability evaluation, the aspects evaluated were effectiveness, efficiency, and level of user satisfaction as proposed by the ISO/IEC 25000 family of standards. The results show that the teachers presented effectiveness, efficiency, and satisfaction considered satisfactory, while the students presented effectiveness and satisfaction classified as satisfactory and acceptable efficiency. The usability evaluation described in this work can serve as a reference for developers seeking to improve learning monitoring and personalization services development.

Keywords: mobile learning; learning customization; learning monitoring; software quality; usability evaluation

For citation: del Ángel-Flores H., López-Domínguez E., Hernández-Velázquez Y., Domínguez-Isidro S., Medina-Nieto M.A., de la Calleja J. Mobile Learning Platform focused on Learning Monitoring and Customization: Usability Evaluation Based on a Laboratory Study. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 89-110. DOI: 10.15514/ISPRAS–2022–34(3)–7

del Ángel-Flores H., López-Domínguez E., Hernández-Velázquez Y., Domínguez-Isidro S., Medina-Nieto M.A., de la Calleja J. Mobile Learning Platform focused on Learning Monitoring and Customization: Usability Evaluation Based on a Laboratory Study. *Trudy ISP RAN/Proc. ISP RAS*, vol. 34, issue 3, 2022, pp. 89-110

Платформа мобильного обучения, ориентированная на мониторинг и настройку обучения: оценка удобства использования на основе лабораторного исследования

[1] *Х. дель Анхель-Флорес, ORCID: 0000-0001-8035-9301 <hdelangel.mca19@lania.edu.mx>*
[2] *Э. Лопес-Домингес, ORCID: 0000-0002-6167-6309 <eduardo.lopez.dom@cinvestav.mx>*
[3] *Е. Эрнандес-Веласкес, ORCID: 0000-0002-5767-532X <yeseniahv@gmail.com>*
[1] *С. Домингес-Исидро, ORCID: 0000-0002-9546-8233 <saul.dominguez@lania.edu.mx>*
[4] *М.А. Медина-Ниэто, ORCID: 0000-0001-6391-4799 <maria.medina@uppuebla.edu.mx>*
[4] *Х. де ла Каллеха, ORCID: 0000-0002-6846-3162 <jorge.delacalleja@uppuebla.edu.mx>*

[1] *Национальная лаборатория перспективной информатики,*
Мексика, Веракрус
[2] *Центр перспективных исследований Национального политехнического института,*
Мексика, CDMX
[3] *Университет Веракрус,*
Мексика, Веракрус
[4] *Политехнический университет Пуэблы,*
Мексика, Пуэбла,

Аннотация. Индивидуализация обучения и мониторинг считаются ключевыми аспектами процессов преподавания и обучения. В некоторых работах предлагались мобильные системы обучения, которые предоставляют учителям и учащимся услуги мониторинга и персонализации обучения. Одним из основных требований к такого рода системам с точки зрения качества программного обеспечения является удобство использования; однако лишь в нескольких работах рассматривались вопросы удобства использования с использованием лабораторных исследований с пользователями в реальных доменах. В этой работе мы представляем оценку удобства использования сервисов мониторинга и персонализации обучения мобильной обучающей платформы на основе лабораторного исследования, в котором приняли участие девять учителей и десять студентов. В нашей оценке удобства использования оценивались такие аспекты, как эффективность, результативность и уровень удовлетворенности пользователей, как это предлагается в семействе стандартов ISO/IEC 25000. Результаты показывают, что учителя оценили эффективность, результативность и удовлетворенность как удовлетворительные, в то время как учащиеся оценили эффективность и удовлетворенность как удовлетворительные и эффективность как приемлемую. Оценка удобства использования, описанная в этой работе, может служить справочным материалом для разработчиков, стремящихся улучшить мониторинг обучения и разработку сервисов персонализации.

Ключевые слова: мобильное обучение; индивидуализация обучения; мониторинг обучения; качество программного обеспечения; оценка удобства использования

Для цитирования: дель Анхель-Флорес Х.**,** Лопес-Домингес Э., Эрнандес-Веласкес Е., Домингес-Исидро С., Медина-Ниэто М.А., де ла Каллеха Х. Платформа мобильного обучения, ориентированная на мониторинг и настройку обучения: оценка удобства использования на основе лабораторного исследования**.** Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 89-110. DOI: 10.15514/ISPRAS–2022–34(3)–7.

1. Introduction

Currently, learning monitoring and customization are considered essential aspects of the teaching-learning processes [1], [2]. Learning monitoring is any procedure that provides feedback and information on student progress, which leads to their self-assessment or reflection of the learning process. Furthermore, learning monitoring helps identify students' competencies, what they know, and what they do [2]. The purpose of learning monitoring is to advise students, offer guidance, correct mistakes, help them overcome difficulties in the learning process, and keep track of the process followed by students [2].

On the other hand, learning customization is based on the idea that students learn in different ways and at different rates. It considers the student's knowledge, needs, abilities, and perceptions in the learning process. Therefore, it is considered learner-centered learning training [1]. Both learning monitoring and personalizing complement the teaching-learning process.

However, in practice, it is difficult to carry them out within the framework of traditional education since it is difficult for a teacher to keep track of each of his students and even more difficult to personalize his education based on his skills and abilities. In this context, some works have proposed mobile learning systems [3] - [18]. Nevertheless, the mobile learning platform presented in [18] is characterized by providing the teachers and students with various learning monitoring and personalization services that include mobile learning objects, considering the students' learning styles and context information. This platform comprises three main components: a mobile learning object generator system (SiGOAM), a mobile learning object repository (MLOR), and a mobile application (MoApp) focused on the student. The integration of these three components allows and facilitates the teacher the systematized implementation of various strategies for learning monitoring and customization. In terms of software quality, one of the main requirements that this type of system must consider is usability, which refers to the degree to which a software product can be used by a certain group of users to achieve clearly defined objectives with effectiveness, efficiency, and satisfaction [19], [20].

However, there is limited research work [7], [21]-[27] focused on the usability evaluation of mobile learning systems that identify and address usability issues using laboratory studies with students and teachers in a real domain. Some of the possible problems derived from a lack of usability evaluation in these types of systems are potential errors in the interface design, unacceptable ease of use, errors that the user makes when interacting with the software in a real environment, among others [28].

This work presents a usability evaluation of the learning monitoring and personalization services of the mobile learning platform proposed in [18] based on a laboratory study in which nine teachers and ten students participated. In our usability evaluation, the aspects evaluated were effectiveness, efficiency, and level of user satisfaction as proposed by the standards ISO/IEC 25010 and ISO/IEC 25022 [19], [20]. Based on the results obtained, it was determined that users with a teacher profile presented 87.11% effectiveness, 80.08% efficiency, and 7.53 satisfaction concerning SiGOAM and MLOR. These three results are considered satisfactory.

On the other hand, users with a student profile presented 80.00% effectiveness, 77.30% efficiency, and 7.37 satisfaction regarding the mobile application (MoApp). Therefore, the efficiency and satisfaction scores are classified as satisfactory, and the efficiency as acceptable. Based on the feedback from users who participated in the usability evaluation, improvements and extensions were carried out to achieve a higher degree of usability of the mobile learning platform. The usability evaluation described in this work can serve as a reference for developers seeking to improve learning monitoring and personalization services development.

This paper is organized as follows: Section 2 presents the state-of-the-art usability evaluations applied to mobile learning tools focused on learning monitoring and/or customization. Section 3 describes the usability evaluation carried out, and the description of the evaluation instruments used in the said evaluation. Section 4 presents the analysis of the results obtained by the usability evaluation. Section 5 describes the improvements and extensions made to the platform, considering the results obtained from the usability evaluation. Finally, the conclusions and future work are presented in section 6.

2. Related Work

Some works proposed in the specialized literature have carried out usability evaluations of mobile learning systems [7], [21]-[27]. These works used diverse evaluation approaches and aspects,

including different evaluation instruments. A comparative analysis shown in Table 1 is presented below.

Table 1. Comparative table of related works to usability evaluations.

		[7]	[21]	[22]	[23]	[24]	[25]	[26]	[27]
General characteristics	Web		✓			✓			
	Mobile	✓		✓	✓		✓	✓	✓
	Learning monitoring services			✓	✓		✓	✓	✓
	Learning reinforcement services	✓	✓	✓			✓	✓	
	Learning customization services	✓			✓	✓			
	Field study	✓			✓				✓
	Laboratory study		✓	✓		✓	✓	✓	
Nielsen Decalogue	Show system status				✓				✓
	Maintain consistency between system and reality				✓				✓
	Give the user full control				✓				✓
	Stick to standards and be consistent				✓				✓
	Prevent errors				✓				✓
	Let the user choose instead of requiring them to remember things				✓				✓
	Ensure flexibility and efficiency			✓	✓		✓		✓
	Take care of aesthetics and moderation				✓				✓
	Ensure effective error handling				✓				✓
	Provide support and documentation				✓				
ISO/IEC 25010	Effectiveness	✓	✓	✓		✓	✓	✓	
	Efficiency		✓					✓	
	Satisfaction	✓	✓	✓		✓	✓	✓	
	Freedom from risk								
	Context coverage								
Evaluation instruments	Questionnaire proposed at work		✓	✓	✓		✓		✓
	Observation								✓
	Usability Scaling System (SUS)	✓	✓					✓	
	USE questionnaire					✓			
	Video recording	✓						✓	
	Group interview			✓					

Based on the works reviewed, Table 1 shows a comparative analysis in three aspects: learning services evaluated, usability aspects/characteristics evaluated, and evaluation instruments used. The

target systems of the usability evaluations proposed in the works reviewed offer different services that were grouped into learning monitoring, reinforcement, and customization services. In this aspect, we identified three works [7], [23-24] that carry out usability evaluations of learning customization services; however, two works only assess the characteristics of effectiveness and satisfaction proposed by the ISO/IEC 25010 standard.

On the other hand, the usability aspects evaluated in the studies reviewed were grouped into the characteristics mentioned in the Nielsen decalogue proposed in [29-30] and those suggested by the ISO/IEC 25010 standard [19]. Although the ISO / IEC 25010 standard proposes five characteristics to be evaluated, it was found that for usability evaluations of mobile learning systems, it is frequent to evaluate the characteristics of effectiveness and satisfaction, and only [21] and [26] evaluate three characteristics: effectiveness, efficiency, and satisfaction. Regarding the evaluation instruments used, the works proposed in [21], [22], and [25]-[26] used questionnaires that allow the evaluation of characteristics proposed in the ISO/IEC 25010 standard.

On the other hand, the works proposed in [7], [21], and [26] used the instrument called the usability scale system proposed in [29] that allows calculating the degree of user satisfaction with the evaluated software. Other evaluation instruments identified in the works are the observation by the evaluators making notes [27], video recording of the test carried out [26], and group interview [22]. Finally, we note that there is a lack of works that carry out a usability evaluation of learning monitoring and customization services based on a laboratory study with users to evaluate the characteristics of effectiveness, efficiency, and satisfaction proposed by ISO/IEC 25010.

3. Usability Evaluation Description

In this work, we carry out a usability evaluation of the learning monitoring and personalization services of the mobile learning platform proposed in [18] based on a laboratory study in which nine teachers and ten students participated. In the subsequent sections, we present the details of the usability evaluation carried out, starting with the description of the mobile learning platform, the definition of the laboratory studies, the description of the instruments generated for the evaluation, and ending with the details on the execution of the evaluation.

3.1 Mobile Learning Platform

The mobile learning platform presented in [18] is characterized by the related works proposed in the specialty literature by the following aspects:

a) Consider learning styles while incorporating mechanisms to obtain students' physical activity to provide recommendations of learning objects related to student preferences and conditions;
b) Provide recommendations of learning objects that were useful to other students with equal or similar learning styles;
c) Allow monitoring of the studied learning objects by students; and

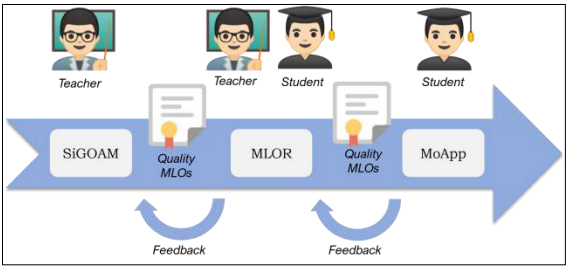


Fig. 1. Mobile Learning Platform components: SiGOAM, MLOR, and MoApp

d) Integrate three tools in a single platform: a mobile learning object generator system (SiGOAM), a mobile learning object repository (MLOR), and a mobile application (MoApp); which allows

carrying a complete flow from when the professor designs and creates a mobile learning object, until the student consults it from a mobile device, and subsequently obtains feedback and self-assessment, see Fig. 1.

The integration of these three components allows and facilitates the professor to implement strategies for learning monitoring and customization systematically.

3.1.1 SiGOAM Description

SiGOAM offers the professor multiple services grouped into four modules to generate quality MLOs: analysis, design, development, and product-oriented tests. The analysis module allows the professor to obtain a guide for acquiring information and the production of the activities composing the MLO, see Fig. 2. In the design module, the professor can esthetically build the educational content obtained in the analysis module. On the other hand, the development module allows the professor to generate a functional MLO prototype. Finally, in the module on product-oriented tests, the professor can assess the technological, pedagogical, and usability aspects of the MLO to obtain feedback from the students and improve it. The learning objects generated by SiGOAM have the following general structure: introduction, lessons, examples, exercises, and evaluations.

Fig. 2. Interface to consult and add resources [18]

3.1.2. MLOR Description

The MLOR component allows teachers to store, catalog, consult, and visualize MLOs generated by SiGOAM, as well as classify MLOs according to the learning style and context defined by the professor when building them. The MLOR also allows the professor to monitor every mobile learning object consulted by students, including the time invested in each of them; see Fig. 3.

Fig. 3. Detail of learning objects studied [18]

3.1.3. AppMo Description

Finally, the mobile application (AppMo) allows the student to consult, view, interact and recommend MLOs based on their learning style and specific context. For example, if a student is moving, ideally, the application recommends learning objects that do not involve lessons in which the student has to read text on the screen. Therefore, the application will recommend learning objects

that include videos and audios. The application also allows students to evaluate learning objects, considering how useful it was for them to understand the studied topic (see Fig. 4).

Fig. 4. Recommended learning objects by students [18]

3.1.4. Learning Monitoring and Customization Services

The main learning monitoring services provided to the professor by the platform are

a) Building mobile learning objects based on the student's learning styles and context;
b) Monitoring the learning objects utilized by the students, as well as the total time invested and the times the student consulted a learning object; and
c) Suggest new learning objects by the professor to reinforce a particular subject difficult for the student.

On the other hand, the main learning customization services offered by the platform are:

a) Identify student learning styles by applying the Honey – Alonso questionnaire (CHAEA) [31];
b) Obtain the context of the student based on the data collected by the sensors of the mobile device to determine the activity that the student is doing, which can be at rest or in motion;
c) Recommendation of learning objects based on the learning styles and physical activity of the students, and
d) Recommendation of learning objects evaluated by other students with the same learning styles.

3.2 Laboratory Studies

In this work the laboratory studies were designed to evaluate the usability of SiGOAM, MLOR, and MoApp from the point of view of their respective end-users. Usability evaluations based on laboratory studies have the following advantages:

- Identification of problems to improve the design of the software.
- Confirm or question assumptions made during the design process.
- Collection of quantitative data, for example, how long it takes users to complete a task and the number, type and severity of errors they make.
- Feedback from target users.
- Detection of usability problems before the launch of a software product.
- Time and cost savings when dealing with concerns.
- Obtain information on user satisfaction regarding the software before its general launch.

- Validate usability requirements.
- Impartial evaluation of the software.

In our case, we carry out a laboratory study to evaluate the usability of the SiGOAM and MLOR services, and another laboratory study to evaluate the usability of the MoApp services. Details are described in the following subsections.

3.2.1 Laboratory Study of the Usability Evaluation Applied to SiGOAM and MLOR

This section describes the scenario of the laboratory study that frames the usability evaluations of SiGAOM and MLOR. In the SiGOAM authoring tool, the teacher user builds an MLO using the services provided by SiGOAM in four phases: analysis, design, development, and product-oriented testing. Afterward, the teacher user publishes and distributes in the MLOR the MLOs generated from SiGOAM.

3.2.1.1 Case Study Definition

This section addresses the main aspects of the case study designed to evaluate the SiGOAM and MLOR.

The object of study: The objects of study were the SiGOAM and MLOR. SiGOAM can be accessed through the link: http://sigoam.lania.mx/login and MLOR can be accessed through the link: http://roa.lania.mx. With this, the accessibility and availability of the platform were remotely guaranteed.

Purpose: To evaluate the characteristics of effectiveness, efficiency, and level of satisfaction proposed in the ISO / IEC 25010 [19] and ISO / IEC 25022 [20] standards, through a laboratory study with a teacher profile.

Quality approach: The aspects evaluated were effectiveness, efficiency, and level of the user satisfaction of the SiGOAM and MLOR.

Perspective: Obtain users' point of view with a teacher profile through comments and suggestions

Context: The experiment was carried out virtually through individual video calls with each participant and with the support of a total of nine teachers: seven from the Teaching Center of the National Laboratory of Advanced Informatics (LANIA), located in the city of Xalapa, Veracruz - Mexico, and two from the faculty of the Polytechnic University of Puebla, located in the city of Puebla, Puebla-Mexico.

The objective of the laboratory study definition was to analyze the flow that the construction, publication, and distribution of the MLOs entails to evaluate the usability in terms of effectiveness, efficiency, and user satisfaction with respect to the SiGOAM and MLOR from the point of view of the professor users from LANIA and the Polytechnic University of Puebla.

3.2.1.2 Planning

This section details the activities carried out for applying the usability evaluation.

Selection of subjects: we select seven professors from the LANIA Teaching Center, representing 70% of the staff, with intermediate and advanced knowledge in using computers and mobile devices. In addition, two professors from the Polytechnic University of Puebla participated. Two participants expressed being MLO users with a teacher profile. The participants were selected to gather their perspectives regarding SiGOAM and MLOR to identify and address elements for improvement. Table 2 presents the main characteristics of the participants in the usability evaluation with a teacher profile. Four participants are male (44.44%), and the remaining five are female (55.56%). Most of the participants (55.56%) are between 36 and 45 years old. Two participants have master's degrees (22.22%), and seven with doctoral studies (77.78%).

Table 2. Characteristics of the participants with a teacher profile

Characteristics	Number of participants
Gender	
Man	4
Woman	5
Age	
26 – 35	1
36 – 45	5
46 – 55	3
Level of studies	
Master's degree	2
Doctorate	7

3.2.1.3 Design of the Experiment

The general elements regarding the design of the experiment are presented below.

Randomization: The tasks to be carried out by the teachers were not assigned randomly. For the construction of the MLO during the evaluation test, it was necessary to ask each participant to have on hand a topic of their choice with an introduction, examples, exercises, and a brief evaluation. For the distribution of the MLOs, the same ones generated in the SiGOAM usability evaluation were used. For the evaluation, it was necessary to previously register test students in the MLOR so that the teachers could carry out the requested tasks. The details of the tasks performed by the teachers are those described in instrument I section.

3.2.2 Laboratory Study of the Usability Evaluation Applied to MoApp

This section describes the scenario of the laboratory study under which the usability evaluation of the mobile application (MoApp) was carried out with student-profile users using MLOs generated according to their context, i.e., if the student is at rest or in motion, and their predominant learning styles.

3.2.2.1 Case Study Definition

This section addresses the main aspects of the case study designed to evaluate the MoApp.

The object of study: The object of study is the MoApp developed on Android. Each user was asked to install the application on their mobile device.

Purpose: To evaluate the characteristics of effectiveness, efficiency, and level of satisfaction proposed in the ISO / IEC 25010 [19] and ISO / IEC 25022 [20] standards, through a laboratory study with real users playing the role of students.

Quality approach: The evaluated aspects are effectiveness, efficiency, and level of user satisfaction concerning the MoApp.

Perspective: Obtain the users' point of view with the student role through comments and suggestions.

Context: The experiment was carried out virtually through individual video calls with each participant. It supported ten students with a master's degree in Applied Computing from the Teaching Center of the National Laboratory for Advanced Informatics, located in Xalapa Veracruz-Mexico.

Therefore, the definition of the laboratory study is to analyze the use of MLOs by students to evaluate usability in terms of effectiveness, efficiency, and satisfaction.

3.2.2.2 Planning

This section details the activities carried out to apply the usability evaluation.

Selection of subjects: For the evaluation of the mobile application (MoApp), an open invitation was launched using an email to which ten students of the master's in applied computing of the Teaching Center of the National Laboratory of Advanced Informatics responded. The only requirement to participate was to have a mobile device with an Android operating system. Their participation was voluntary and was not conditioned on any type of benefit for their subjects. Table 3 shows the main characteristics of the participants in the usability evaluation with the student profile. Nine men (90%) and one woman (10%) participated in the evaluation. Most of them are between 18 and 25 years old (60%). They all have a bachelor's degree.

Regarding the Android versions of the mobile devices, they used for the usability test; two were Android 9 (20%), five were Android 10 (50%), and three with Android 11 (30%).

Table 3. Characteristics of the participants with a student profile

Characteristics	**Number of participants**
Gender	
Man	9
Woman	1
Age	
18 – 25	6
26 – 35	3
36 – 45	1
Level of studies	
Bachelor's degree	10
Android version	
9	2
10	5
11	3

3.2.2.3 Design of the Experiment

The general elements regarding the design of the experiment are presented below.

Randomization: The tasks that the students had to perform for the usability evaluation were not assigned randomly. The tasks that the students carried out are described in instrument III. The OAMs used by the students were the same that the teachers constructed and published in the laboratory study described for SiGOAM and MLOR.

3.2.2.4 Operation

Teachers and students were not informed about the usability characteristics to be evaluated, they were only announced that the purpose of the study was to analyze and assess the quality in use of the mobile learning platform. Afterward, teachers and students signed an informed consent document. Therefore, no platform training was carried out for the execution of the usability test.

3.3 Evaluation Instruments

Based on the characteristics and metrics proposed in the ISO / IEC 25010 [19] and ISO / IEC 25022 [20] standards, three evaluation instruments were made. The instruments generated to carry out the usability evaluation can be classified into:

- Instrument to collect data to evaluate the effectiveness and efficiency of SiGOAM and MLOR from the teachers' perspective, hereinafter Instrument I.

- Instrument to collect data to evaluate the effectiveness and efficiency of MoApp of the mobile learning platform from the perspective of the students, hereinafter Instrument II.
- Instrument to collect data that allows obtaining the degree of satisfaction of the users of the mobile learning platform and identifying the characteristics to be improved on the platform, hereinafter Instrument III.

3.3.1 Instrument I Description

This instrument was used to measure the effectiveness and efficiency of users with a teacher profile to the Mobile Learning Object Generator System and the Learning Object Repository, specifically evaluating the learning monitoring and personalization services. Table 4 shows the breakdown of the services included in the instrument I grouped by tasks.

Table 4. The instrument I task breakdown

Task	**Subtask**
Analysis module	Create a new MLO
	Register the student's profile
	Add content
	Add examples
	Add exercises
	Record metadata
Design module	Design content
	Design examples
	Design exercises
	Design evaluation
Development module	Build MLO
	Publish MLO
Product-oriented assessment module	Usability assessment
	Assessment of technological and pedagogical aspects
Publish MLO	Complete MLO publication
Manage group	Register new group
	Add category to group
	Add student to group
	Add MLO to category
	Consult studied MLO
Reinforcement	Suggest MLO for reinforcement

3.3.2 Instrument II Description

This instrument was used to evaluate the effectiveness and efficiency of the students concerning the platform's mobile application (MoApp). In addition, learning monitoring and personalization services were specifically evaluated. Table 5 shows the breakdown of the services included in instrument II, grouped by tasks.

Table 5. The instrument II task breakdown

Task	**Subtask**
Set learning style	Answer the CHAEA questionnaire
	Consult learning style
	Set preferences
Consult MLO	Consult added MLO
	Consult reinforcement MLO
	Consult recommended MLO

Interact with the MLO	View detail of MLO
	View of MLO
	Evaluate MLO
	Consult best evaluated MLO
Support	Consult teachers
	Send mail to teacher

3.3.3 Instrument III Description

This instrument was used to evaluate the degree of satisfaction of users with the teacher and student profile. An adaptation of the user satisfaction questionnaire (QUIS) was carried out in version 7.0 proposed in [32]. This consists of thirty-two questions grouped into six categories, which are:

- Global reaction to the system. It presents the user with questions about his perception regarding utility, flexibility, ease, and other general aspects of the system.
- General reactions on the screen. It collects information to evaluate screen characteristics such as typography, design, distribution, and sequence between windows.
- System information and terminology. It contains questions that seek to evaluate the concepts used to determine if they are useful for the user to complete the tasks within the system.
- Learning capacity. Question the user regarding the ease of learning to use the software.
- System capabilities. It collects information that allows knowing the performance and recovery between errors made by the user.
- Ease of use and user interface. Question the user about general aspects of the software interface design.

3.4 Execution of the Evaluation

This section details the activities to carry out the usability evaluation.

3.4.1 Execution of Usability Evaluation with Teachers

For the evaluation with teachers, the experiment lasted approximately an hour and a half, framed in a video call, where the teachers carried out the tasks described in instruments I and II. For each task carried out, a note was made of the start and end time of the task or abandonment in case of not completing it. After the usability test, the user satisfaction questionnaire was applied (QUIS 7.0) [32]. Participants were asked to comment out loud on the problems they encountered, their impressions, and what they were trying to do on the platform, to obtain a video recording of each of their comments regarding their experience with the platform.

3.4.2 Execution of Usability Evaluation with Students

For the usability evaluation from the students' perspective, the experiment lasted forty-five minutes through a video call, where the students performed the tasks described in instrument II. The start time of each task was controlled, as well as the completion time of the task or abandonment in case of not completing it. After the usability test, the user satisfaction questionnaire (QUIS 7.0) proposed in [32] was applied. Previously, each participant was asked to install MoApp of the learning platform on their mobile device with the Android operating system. The test was videotaped to analyze each of the comments that the participants were making at the time of carrying out the requested tasks.

4. Analisys of Results

This section describes the results obtained after conducting the usability evaluation in terms of effectiveness, efficiency, and satisfaction.

4.1. Results of Instrument I

The instrument I was applied to users with a teacher profile, who were asked to perform seven tasks distributed between the SiGOAM and MLOR applications. Each participant was asked to fill out this instrument to determine their effectiveness and efficiency for the platform. The results obtained indicate that users present 87.11% average effectiveness and 84.08% average efficiency for SiGOAM and MLOR, respectively. Here is a detailed description of the results.

4.1.1 Effectiveness Results

The instrument I allowed to obtain the effectiveness value of the users for SiGOAM and MLOR. Therefore, the results obtained for each one are described in detail below.

4.1.1.1 Results of SiGOAM

In Table 6 it is indicated with a 1 in case the user has completed the task successfully and with a 0 otherwise. To obtain the efficiency value per user, the tasks that were completed were counted and the value obtained was divided by the total number of attempted tasks. In SiGOAM, monitoring and customization services were evaluated in tasks grouped by analysis, design, development, and product-oriented testing modules. Only five of the nine participants (55.56%) completed all the tasks successfully, while the others failed one of the tasks (44.44%).

Table 6. Results of completion of task in SiGOAM of users with a teacher profile

Task	**Task completion**								
	U1	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**	**U9**
Analysis module	0	1	0	1	1	1	1	1	1
Design module	1	1	1	1	1	1	1	1	0
Development module	1	0	1	1	1	1	1	1	1
Product-oriented assessment module	1	1	1	1	1	1	1	1	1
Efficiency value	0.75	0.75	0.75	1.00	1.00	1.00	1.00	1.00	0.75

Considering the task completion values in Table 6 and substituting these values in the effectiveness formula, we have:

$$SiGOAM\ Effectiveness = \left(\frac{0.75 + 0.75 + 0.75 + 1.00 + 1.00 + 1.00 + 1.00 + 1.00 + 0.75}{9} \right) \times 100,$$

$$SiGOAM\ Effectiveness = \left(\frac{8}{9}\right) \times 100 = 88.89.$$

4.1.1.2 Results of MLOR

Table 7 presents the tasks corresponding to the MLOR and indicates with 1 those that users could complete and with 0 those that users did not complete successfully. In this regard, the effectiveness value is obtained by dividing the number of completed tasks by the total number of attempted tasks. The activities evaluated in the MLOR were grouped into the tasks: publish MLO, group management, and reinforcement. Five of the participants managed to complete all these tasks without problems (55.56%), while the others had difficulty completing any of the tasks (44.44%).

Table 7. Result of completion of task MLOR of users with a teacher profile

Task	**Task completion**								
	U1	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**	**U9**
Publish MLO	1	1	1	1	1	1	1	1	1

Manage group	0	0	1	0	1	0	1	1	1
Reinforce-ment	1	1	1	1	1	1	1	1	1
Efficiency value	0.67	0.67	1.00	0.67	1.00	0.67	1.00	1.00	1.00

Considering the task completion values in Table 7, and substituting these values in the effectiveness formula, we have:

$$MLOR\ Effectiveness = \left(\frac{\frac{0.67+0.67+1.00+0.67+1.00}{+0.67+1.00+1.00+1.00}}{9}\right) \times 100,$$

$$MLOR\ Effectiveness\ = \left(\frac{7.68}{9}\right) \times 100 = 85.33.$$

4.1.1.3 Average Effectiveness Result

We obtained the final effectiveness result through the average of the effectiveness values in both SiGOAM and MLOR. The value obtained indicates that users with a teacher profile managed to complete the tasks in SiGOAM and MLOR with 87.11% effectiveness, which is considered a satisfactory value according to the ranges of acceptability for effectiveness reported in [33].

4.1.2 Efficiency Results

The instrument I allowed obtaining the efficiency value of users regarding SiGOAM and MLOR. Therefore, the results obtained for each one is described in detail below.

4.1.2.1 Results of SiGOAM

Table 8 shows the times in minutes that each user with a teacher profile took to complete each task of the SiGOAM described in instrument I. For the case of users who could not complete the task, the time in which they abandoned the task is indicated.

Table 8. Results of times of users with teacher profiles in SiGOAM

Task	**Total time in minutes**								
	U1	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**	**U9**
Analysis module	29	29	45	60	39	31	34	47	22
Design module	9	11	6	6	6	6	6	15	6
Development module	3	9	2	2	2	3	2	3	2
Product-oriented assessment module	4	2	2	6	2	3	2	4	3

The user efficiency value for SiGOAM considers the task completion rate, in addition to the completion times shown in Table 8. For this, the efficiency value per task was calculated. In this concern, we averaged the efficiency values of the tasks: T1, T2, T3, and T4. Obtaining that user with a teacher profile completed the SiGOAM tasks with 84.34% efficiency. Data for the calculation are the next:

$$Efficiency\ T1\text{: } 77.98$$

$$Efficiency\ T2\text{: } 91.55$$

$$Efficiency\ T3\text{: } 67.86$$

$$Efficiency\ T4\text{: } 100.00$$

$$SiGAOM\ Efficiency = \frac{77.98 + 91.55 + 67.86 + 100}{4},$$

$$SiGOAM\ Efficiency = \frac{434.42}{4} = 84.34\ \%$$

4.1.2.2 Results of MLOR

Table 9 shows the times, expressed in minutes, that each user with a teacher profile took to perform the MLOR tasks described in instrument I. For the case of users who could not complete the task, the time in which they abandoned the task is indicated.

Table 9. Results of users with teacher profiles in MLOR

Task	**Total time in minutes**								
	U1	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**	**U9**
Publish MLO	4	3	3	4	2	3	1	6	3
Manage group	8	10	5	8	8	8	5	9	9
Reinforce-ment	4	2	1	7	2	5	4	10	4

To obtain the efficiency value of the users concerning the MLOR, in addition to considering the task completion rate, the completion times are shown in Table 9. For this, the efficiency value per task was calculated. The average of the efficiency values of the tasks was obtained: T5, T6, and T7. Obtaining those users with a teacher profile completed the MLOR tasks with 83.81% efficiency. Data for the calculation are the following:

$$Efficiency\ T5\colon 100.00$$
$$Efficiency\ T6\colon 51.42$$
$$Efficiency\ T7\colon 100.00$$
$$MLOR\ Efficiency = \frac{100 + 51.42 + 100}{3},$$
$$MLOR\ Efficiency = \frac{251.42}{3} = 83.81\ \%.$$

4.1.2.3 Final Efficiency Result

Finally, to obtain the efficiency value of the users for the SiGOAM and MLOR applications, the average of the efficiency values in SiGOAM and efficiency in MLOR was obtained. The value obtained indicates that the users with a teacher profile completed the tasks in SiGOAM and MLOR with 84.08% of average efficiency, which is considered satisfactory according to the acceptability ranges for efficiency presented in [33].

4.2. Results of Instrument II

Instrument II was applied to users with a student profile, who were asked to perform four tasks in MoApp of the mobile learning platform focused on learning on monitoring and customization services. The students filled out instrument II to collect information to determine the effectiveness and efficiency of MoApp. The results obtained indicate that the users present 80.00% effectiveness and 77.30% efficiency for the evaluated mobile application.

4.2.1 Effectiveness Results

Table 10 indicates with a 1 if the user completed the task and with 0 otherwise. To obtain the efficiency value per user, the tasks that were completed were counted, and the value obtained was divided by the total number of attempted tasks. Only five users completed (50%) the four tasks in instrument I, while the remaining five users had problems completing at least one task (50%).

Table 10. Task completion results for users with a student profile

Task	**Task completion**									
	U1	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**	**U9**	**U10**
Set learning style	1	1	1	1	1	1	1	1	1	1
Consult MLO	1	1	1	1	1	0	0	0	0	1
Interact with the MLO	1	1	0	1	1	1	0	0	0	1
Support	1	1	1	1	1	1	1	1	1	1
Effectiveness value	1.00	1.00	0.75	1.00	1.00	0.75	0.50	0.50	0.50	1.00

Considering the task completion values in Table 10, and substituting these values in the effectiveness formula, we have:

$$Effectiveness = \left(\frac{1.00 + 1.00 + 0.75 + 1.00 + 1.00}{+0.75 + 0.50 + 0.50 + 0.50 + 1.00} \right) \times 100,$$

$$Effectiveness = \left(\frac{8.00}{10} \right) \times 100 = 80.00.$$

The result obtained indicates that users with a student profile were able to complete the tasks in the mobile application with an effectiveness of 80.00%, which is considered satisfactory according to the acceptability ranges for effectiveness shown in [33].

4.2.2 Efficiency Results

Table 11 shows the times in minutes that each user with a student profile took to complete each task described in instrument II. For the case of users who could not complete the task, the time in which they abandoned the task is indicated.

Table 11. Results of times of users with student's profile

Task	**Total time in minutes**									
	U1	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**	**U9**	**U10**
Set learning style	12	11	12	11	8	12	18	12	13	14
Consult MLO	9	4	6	4	4	3	11	5	7	5
Interact with the MLO	12	8	12	8	4	5	13	6	9	10
Support	2	1	2	2	1	2	2	2	2	3

The efficiency value of the users for MoApp considers the task completion rate indicated in Table 10 and the completion times in Table 11. For this, the efficiency value per task was calculated. The formula used to obtain the efficiency per task is described in [33].

In this concern, the efficiency values per task were averaged. Data for the calculation are the following:

$$Efficiency\ T1: 100.00$$
$$Efficiency\ T2: 55.17$$
$$Efficiency\ T3: 54.02$$
$$Efficiency\ T4: 100.00$$

To obtain the result for efficiency, we have the following:

$$Efficiency = \frac{100.00 + 55.17 + 54.02 + 100.00}{4},$$

$$Efficiency = \frac{309.19}{4} = 77.30\ \%.$$

The efficiency result obtained is 77.30%, which is considered satisfactory according to the acceptability ranges for efficiency shown in [32].

4.3. Results of Instrument III

Instrument III, which is based on the QUIS 7.0 proposed in [32], allowed us to determine the degree of satisfaction of teachers and students. It was obtained that the teachers achieved satisfaction of 7.53, while the student's satisfaction of 7.37. The degree of satisfaction is given from 0 to 9 as proposed in [32]. According to the acceptability ranges presented in [32], both scores are considered satisfactory.

The degree of satisfaction of the teachers for the SiGOAM and MLOR resulted from calculating the final average of instrument III from the perspective of the teacher user. The final average was obtained by adding each of the averages obtained by category and dividing the result among the total of categories. Data for the calculation are the following:

$Overall\ reaction\ to\ the\ software$: 7.41

$Screen$: 7.75

$Terminology\ and\ system\ information$: 7.44

$Learning$: 7.24

$System\ capabilities$: 7.78

$Usability\ and\ UI$: 7.56

To obtain the result for satisfaction, we have the following:

$$Satisfaction = \frac{7.41 + 7.75 + 7.44 + 7.24 + 7.78 + 7.56}{6} = 7.53.$$

The category with the highest score was the one that has to do with aspects of system capabilities. In contrast, the one with the lowest score is the category of learning capabilities.

We calculated the final average of instrument III to obtain the degree of satisfaction of the students for MoApp. The averages for each category were summed and divided by the total categories. Data for the calculation are the following:

$Overall\ reaction\ to\ the\ software$: 6.97

$Screen$: 7.35

$Terminology\ and\ system\ information$: 7.83

$Learning$: 6.95

$System\ capabilities$: 7.32

$Usability\ and\ UI$: 7.80

To obtain the result for satisfaction, we have the following:

$$Satisfaction = \frac{6.97 + 7.35 + 7.83 + 6.95 + 7.32 + 7.80}{6} = 7.37.$$

The category with the highest score is that of technology and information of the system, and the one that obtained the lowest score is the one that has to do with aspects of learning capacity.

5. Conclusions and Future

In this work, the usability evaluation of the mobile learning platform focused on learning monitoring and customization proposed in [18] was carried out based on a laboratory study to determine and improve its quality in use. Learning monitoring and customization services were specifically evaluated. The platform was evaluated under the characteristics and metrics proposed in the ISO/IEC 25010 and ISO/IEC 25022 standards. The evaluated characteristics were effectiveness, efficiency, and satisfaction from the users' perspective (teacher and student). For the evaluation, a laboratory study was designed for each platform element: SiGOAM, MLOR, and MoApp. Nine teachers and ten students participated in the laboratory study, with their comments and suggestions allowing us to identify usability issues that were corrected in their entirety. Based on the results obtained, it was determined that users with a teacher profile presented 87.11% effectiveness, 80.08% efficiency, and 7.53 satisfaction concerning SiGOAM and MLOR. These three results are considered satisfactory.

On the other hand, users with a student profile presented 80.00% effectiveness, 77.30% efficiency, and 7.37 satisfaction regarding the mobile application (MoApp). Therefore, the efficiency and satisfaction scores are classified as satisfactory, and the efficiency as acceptable.

In future work, we propose the integration of an adaptability engine [34] in terms of content, format, route, feedback, and evaluations.

References / Список литературы

[1] UNESCO, Personalized Learning, 2017. Available at: https://unesdoc.unesco.org/ark:/48223/pf0000250057, accessed: Nov. 21, 2021.

[2] Y. Hernández-Velázquez, C. Mezura-Godoy, V.Y. Rosales-Morales. M-Learning and Student-Centered Design: A Systematic Review of the Literature. Advances in Intelligent Systems and Computing, vol. 1297, 2020, pp. 349-363.

[3] M. Abech, C.A. da Costa et al. A Model for Learning Objects Adaptation in Light of Mobile and Context-Aware Computing. Personal and Ubiquitous Computing, vol. 20, no. 2, 2016, pp. 167-184.

[4] S.S. Oyelere, J. Suhonen et al. Design, development, and evaluation of a mobile learning application for computing education. Education and Information Technologies, vol. 23, no. 1, 2018, pp. 467-495.

[5] S. Benhamdi, A. Babouri, and R. Chiky. Personalized recommender system for e-Learning environment. Education and Information Technologies, vol. 22, no. 4, 2017, pp. 1455-1477.

[6] H. Xie, D. Zou et al. Personalized word learning for university students: a profile-based method for e-learning systems. Journal of Computing in Higher Education, vol. 31, no. 2, 2019, pp. 273-289.

[7] D. Cáliz, J. Gomez et al. Evaluation of a usability testing guide for mobile applications focused on people with down syndrome (USATESTDOWN). Lecture Notes in Computer Science, vol. 10069, 2016, pp. 497-502.

[8] S. S. Rani, S. Krishnanunni. Educational App for Android-Specific Users—EA-ASU. Advances in Intelligent Systems and Computing, vol. 1097, 2020, pp. 325-335.

[9] H. Imran, M. Belghis-Zadeh. PLORS: a personalized learning object recommender system. Vietnam Journal of Computer Science, vol. 3, no. 1, 2016, pp. 3-13.

[10] S. Saryar, S.V. Kolekar et al. Mobile learning recommender system based on learning styles. Advances in Intelligent Systems and Computing, vol. 900, 2019, pp. 299-312.

[11] K. Meenakshi, R. Sunder et al. An intelligent smart tutor system based on emotion analysis and recommendation engine. In Proc. of the 2017 International Conference on IoT and Application (ICIOT), 2017, pp. 1-4.

[12] L. G. Martínez, S. Marrufo et al. Using a Mobile Platform for Teaching and Learning Object Oriented Programming. IEEE Latin America Transactions, vol. 16, no. 6, 2018, pp. 1825-1830.

[13] Y. Li, L. Wang. Using iPad-based mobile learning to teach creative engineering within a problem-based learning pedagogy. Education and Information Technologies, vol. 23, no. 1, 2018, pp. 555-568.

[14] A. Sharma. A proposed e-learning system facilitating recommendation using content tagging and student learning styles. In Proc. of the 2017 5th National Conference on E-Learning &ELearning Technologies (ELELTECH), 2017, pp. 1-6.

[15] R.A.W. Tortorella and S. Graf. Considering learning styles and context-awareness for mobile adaptive learning. Education and Information Technologies, vol. 22, no. 1, 2017, pp. 297-315.
[16] T. Jagušt, I. Botički. Mobile learning system for enabling collaborative and adaptive pedagogies with modular digital learning contents. Journal of Computers in Education, vol. 6, 2019, pp. 335-362.
[17] C. B. Yao. Constructing a User-Friendly and Smart Ubiquitous Personalized Learning Environment by Using a Context-Aware Mechanism. IEEE Transactions on Learning Technologies, vol. 10, no. 1, 2017, pp. 104-114.
[18] C. Huerta-Guerrero, E. Lopez-Dominguez et al. Kaanbal: A Mobile Learning Platform Focused on Monitoring and Customization of Learning. International Journal of Emerging Technologies in Learning, vol. 16, no. 1, 2020, pp. 18-43.
[19] ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 2011.
[20] ISO/IEC 25022:2016 Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use, 2016.
[21] A. A. Arain, Z. Hussain et al. Evaluating usability of M-learning application in the context of higher education institute. Lecture Notes in Computer Science, vol. 9753, 2016, pp. 259-268.
[22] B. A. Kumar, P. Mohite. Usability Study of Mobile Learning Application in Higher Education Context: An Example from Fiji National University. In Mobile Learning in Higher Education in the Asia-Pacific Region, Springer Singapore, 2017, pp. 607-622.
[23] A. Pensabe-Rodriguez, E. Lopez-Dominguez et al. Context-aware mobile learning system: Usability assessment based on a field study. Telematics and Informatics, vol. 48, issue C, 2020, article no. 101346.
[24] D. Hariyanto, M. B. Triyono, and T. Köhler. Usability evaluation of personalized adaptive e-learning system using USE questionnaire. Knowledge Management & E-Learning: An International Journal, vol. 12, no. 1, 2020, pp. 85-105.
[25] B. A. Kumar and P. Mohite. Usability guideline for mobile learning apps: An empirical study. International Journal of Mobile Learning and Organisation, vol. 10, no. 4, 2016, pp. 223-237.
[26] S. Yağmur and M. P. Çakır. Usability evaluation of a dynamic geometry software mobile interface through eye tracking. Lecture Notes in Computer Science, vol. 9753, 2016, pp. 391-402.
[27] M. Asghar, I.S. Bajwa et al. A Genetic Algorithm-Based Support Vector Machine Approach for Intelligent Usability Assessment of m-Learning Applications. Mobile Information Systems, vol. 2022, 2022, Article ID 1609757, 20 p.
[28] B. A. Kumar, P. Mohite. Usability of mobile learning applications: a systematic literature review. Journal of Computers in Education, vol. 5, no. 3, 2018, pp. 1-17.
[29] T. S. Tullis and J. N. Stetson. A Comparison of Questionnaires for Assessing Website Usability. In Proc. of the UPA 2004 Conference, 2004, pp. 1-12.
[30] N. Jacob. Usability Heuristics for User Interface Design. URL: https://www.nngroup.com/articles/ten-usability-heuristics/, accessed: Nov. 21, 2021.
[31] C. M. Alonso, D. J. Domingo, and P. Honey. Los estilos de aprendizaje, Procedimientos de diagnóstico y mejora. Séptima edición, Bilbao: Editorial Mensajero, 2007 (in Spanish).
[32] K. L. Chin, J.P., Diehl, V.A., Norman. Questionnaire for User Interface Satisfaction. Development of an Instrument Measuring User Satisfaction of the Human-Computer Interface. In Proc. of the SIGCHI Conference on Human Factors in Computing Systems, 1988, pp. 213-218.
[33] ISO, Ergonomics of human-system interaction - Part 11: Usability: Definitions and concepts. Iso/Np 9241-11, 2018.
[34] E.N. Chujkova, A.R. Aidinyan, O.L. Tsvetkova. Adaptation Algorithm for Application Menus. Programming and Computer Software, vol. 46, no. 6, 2020, pp. 397–405 / Е.Н. Чуйкова, А.Р. Айдинян, О.Л. Цветкова. Алгоритм адаптивного изменения меню программного приложения. Программирование, том 46, no. 6, 2020 г., стр. 30-40.

Информация об авторах / Information about authors

Herminio ДЕЛЬ ÁNGEL-FLORES is an associate professor in the Department of Computer Science at Laboratorio Nacional de Informática Avanzada (LANIA), in Veracruz, Mexico. He completed her MSc degree at the National Laboratory of Applied Informatics (LANIA), Mexico in 2021. His areas of interest include mobile learning systems and user experience design.

Эрминио ДЕЛЬ АНХЕЛЬ-ФЛОРЕС – доцент кафедры компьютерных наук. В 2021 году он получил степень магистра в Национальной лаборатории прикладной информатики (LANIA) в Мексике. В сферу его интересов входят мобильные обучающие системы и дизайн пользовательского интерфейса.

Eduardo LÓPEZ-DOMÍNGUEZ received the M.Sc. and Ph.D. degrees from the National Institute of Astrophysics, Optics and Electronics (INAOE), Puebla, Mexico. He is a researcher in the Department of Computer Science at Center for Research and Advanced Studies of the National Polytechnic Institute (CINVESTAV-IPN), CDMX, México. Ph.D. López Domínguez is a member of the Researchers National System Level 1 (SNI). His research interests include mobile distributed systems, partial order algorithms and multimedia synchronization.

Эдуардо ЛОПЕС-ДОМИНГЕС получил степени магистра и доктор философии в Национальном институте астрофизики, оптики и электроники (INAOE), Пуэбла, Мексика. Он является научным сотрудником отдела компьютерных наук. Является членом Национальной системы исследователей уровня 1 (SNI). Его исследовательские интересы включают мобильные распределенные системы, алгоритмы частичного порядка и синхронизацию мультимедиа.

Yesenia HERNÁNDEZ-VELÁZQUEZ is a professor-researcher in the Department of Computer Science at Laboratorio Nacional de Informática Avanzada (LANIA), in Veracruz, Mexico. She completed her MSc Degree at the Benemérita Universidad Autónoma de Puebla (BUAP), Mexico in 2011. Her areas of interest include mobile learning systems and user experience design.

Есения ЭРНАНДЕС-ВЕЛАСКЕС – профессор-исследователь кафедры компьютерных наук. В 2011 году она получила степень магистра в Автономном университете Бенемерита в Пуэбле (BUAP), Мексика. Сферы ее интересов включают системы мобильного обучения и проектирование пользовательского интерфейса.

Saúl DOMÍNGUEZ-ISIDRO received the Ph.D. degree from the Universidad Veracruzana (UV), Mexico. He has been a full-time Professor-researcher with the Department of Computer Science at Laboratorio Nacional de Informática Avanzada (LANIA), in Veracruz, Mexico, since 2018. Ph.D. Domínguez-Isidro is a member of the Researchers National System Level 1 (SNI). His research interests include Combinatorial Optimization, Evolutionary Algorithms, Bio-Inspired Algorithms Optimization Algorithms, and Computational Intelligence.

Сауль ДОМИНГЕС-ИСИДРО получил докторскую степень в Университете Веракрусана (УФ), Мексика. С 2018 года он является штатным профессором-исследователем Департамента компьютерных наук. Домингес-Исидро является членом Национальной системы исследователей уровня 1 (SNI). Его исследовательские интересы включают комбинаторную оптимизацию, эволюционные алгоритмы, биоинспирированные алгоритмы, алгоритмы оптимизации и вычислительный интеллект.

María Auxilio MEDINA-NIETO is a researcher in the Postgraduate Department at the Polytechnic University of Puebla (UPPuebla), Puebla, Mexico. She completed her M.Sc. and Ph.D. Degree from the Universidad de las Americas Puebla, Mexico in 2008. She is a member of the Researchers National System (SNI) Level 1 since 2019 and she has a higher profile recognition (PRODEP). Her areas of interest include ontologies, semantic web, and open educational repositories.

Мария Ауксилио МЕДИНА-НИЭТО – научный сотрудник отдела последипломного образования. Она получила степени магистра наук и доктора философии в университете лас-Американас, Пуэбла, Мексика. Она является членом Национальной системы исследователей (SNI) уровня 1 с 2019 года. В сферу ее интересов входят онтологии, семантическая сеть и открытые образовательные репозитории.

Jorge DE LA CALLEJA received the M.Sc. and Ph.D. degrees from the National Institute of Astrophysics, Optics and Electronics (INAOE), Mexico. He has been a full-time Professor with the Computer Science Department, Polytechnic University of Puebla (UPPuebla), Mexico, since 2008. Ph.D. De la Calleja is a member of the Researchers National System Level 1 (SNI), his research interests include machine learning, computer vision, and data mining with applications in medicine, education, and astronomy.

Хорхе ДЕ ЛА КАЛЬЕХА получил степени магистра и доктора философии в Национальном институте астрофизики, оптики и электроники (INAOE), Мексика. С 2008 года он является штатным профессором факультета компьютерных наук Политехнического университета Пуэблы. Де ла Каллеха является членом Национальной системы исследователей уровня 1 (SNI). Его исследовательские интересы включают машинное обучение, компьютерное зрение и интеллектуальный анализ данных с приложениями в медицине, образовании и астрономии.

DOI: 10.15514/ISPRAS-2022-34(3)-8

Group Cohesion for a Coaching System in Co-located Collaborative Environments

A. Reyes-Flores, ORCID: 0000-0003-0733-8453 <itreyes@uv.mx>
C. Mezura-Godoy, ORCID: 0000-0002-5386-107X <cmezura@uv.mx>
E. Benítez-Guerrero, ORCID: 0000-0001-5844-4198 <edbenitez@uv.mx>
L.G. Montané-Jiménez, ORCID: 0000-0003-2732-5430 <lmontane@uv.mx>
Faculty of Statistics and Informatics of the University of Veracruz,
Xalapa, Veracruz, 91020, Mexico

Abstract. Technologies that support co-located collaboration must not only provide a shared workspace, but also support collaboration. From an observational study, some collaboration problems were identified in groups of people working in a system with a Tangible User Interface. Some of these problems could be identified and prevented with the support of Coaching System. This system encourages interactions between group members through Social Interventions. To develop a Coaching System, it is necessary to know the cohesion between the members of the group, in order to decide the appropriate Social Interventions. In this paper, a model is proposed to represent the social interactions that occur in a group of people when performing a task. Interactions can be analyzed to determine the degree of cohesiveness of a group and support the collaboration.

Keywords: Co-located Collaboration; Group Coaching; Technologies supporting collaboration

For citation: Reyes-Flores A., Mezura-Godoy C., Benítez-Guerrero E., Montané-Jiménez L.G. Group Cohesion for a Coaching System in Co-located Collaborative Environments. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 111-126. DOI: 10.15514/ISPRAS–2022–34(3)–8

Сплоченность группы для системы коучинга в совместной среде

А. Рейес-Флорес, ORCID: 0000-0003-0733-8453 <itreyes@uv.mx>
К. Мезура-Годой, ORCID: 0000-0002-5386-107X <cmezura@uv.mx>
Э. Бенитес-Герреро, ORCID: 0000-0001-5844-4198 <edbenitez@uv.mx>
Л.Х. Монтане-Хименес, ORCID: 0000-0003-2732-5430 <lmontane@uv.mx>
Факультет статистики и информатики Университета Веракрус,
91020, Мексика, Веракрус, Халапа

Аннотация. Технологии, поддерживающие совместную работу, должны не только предоставлять общее рабочее пространство, но и поддерживать коллаборацию. В ходе наблюдательного исследования были выявлены некоторые проблемы совместной работы в группах людей, работающих в системе с материальным пользовательским интерфейсом. Некоторые из этих проблем можно выявить и предотвратить при поддержке Coaching System. Эта система поощряет взаимодействие между членами группы посредством социальных вмешательств. Чтобы разработать систему коучинга, необходимо знать сплоченность членов группы, чтобы принять решение о соответствующих социальных вмешательствах. В данной работе предлагается модель для представления социальных взаимодействий, происходящих в группе людей при выполнении задачи. Взаимодействия можно анализировать, чтобы определить степень сплоченности группы и поддержать сотрудничество.

Ключевые слова: совместная работа; групповой коучинг; технологии, поддерживающие совместную работу

Для цитирования: Рейес-Флорес А., Мезура-Годой К., Бенитес-Герреро Э., Монтане-Хименес Л.Х. Сплоченность группы для системы коучинга в совместной среде. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 111-126. DOI: 10.15514/ISPRAS–2022–34(3)–8.

1. Introduction

Collaboration means that groups of two or more people work together to complete a task or achieve a goal [1]. In order to support working groups to achieve effective collaboration, technological tools have been built under the approach of Computer-Supported Collaborative Work (CSCW). Considering the taxonomy of Ellis et al. [2] and its two dimensions of place and time, collaboration can take several forms: 1) Co-located (same time/same place), 2) Distributed (same time/different place), 3) Asynchronous (different time/same place) and 4) Asynchronous distributed (different time/different place).

One problem that has been identified is that CSCW developments that support collaboration have been mostly focused on remote collaboration and few works aim to support collocated collaboration [3]. However, in this collaborative scenario, some conflicts may also arise in the group that affect the results of the collaborative activity. For example, there are failures in communication, poorly coordinated tasks are executed, some people collaborate less than others, and so on.

Our goal is to support groups of people in collaborative activities in collocated environments. To do this, we have executed an observational study in which we identify some possible problems that could arise in a collocated collaborative activity and how it could be computationally supported.

Based on the work of Van Leeuwen, et. al. [4], we believe that a possible way to support groups in this sense is with the support of an expert facilitator [5] who encourage collaboration, as happens in a classroom through a teacher. The teacher analyzes the behavior of the groups of students and if he detects any problem in their collaboration, he executes a Social Intervention, that is, an expression that the teacher makes to students to encourage collaborative learning. Thus, our goal is to have a computational system that encourage collaboration in groups, just as a teacher would.

Proposals have been made in the literature to support collaboration by providing a shared physical space; facilitate collaboration, considering usability issues in the design of these tools; and invite collaboration, showing users information about some collaboration indicators, such as the number of participations per user or graphical representations of the group's progress. However, there are still few works that contribute to encouraging collaboration.

According to Olson et al. [3], to encourage collaboration, it is necessary to motivate or persuade people to start interacting or maintain continuous interaction. The techniques that have been used to promote collaboration have been through Tutoring Systems, Orchestration Tools or through Guide System interventions with personal agents such as a Coach [6]. In particular, the Coaching Systems observe the interaction of the users and provide them with suggestions, help and/or comments that help improve collaboration while they try to complete their activity [7]. Coaching mechanisms monitor group activity and make recommendations if they detect anomalies in collaboration.

We consider that a Coaching System could support collocated collaborative activities, since these systems have supported remote collaboration [7]. To test this hypothesis, a Coaching System centered on a case study will be developed, and then studies will be conducted to compare the outcomes of coached and unsupported colocalized collaboration. The case study will be collocated collaborative activities in systems with Tangible User Interfaces (TUIs) because these interfaces have enabled collaboration in domains of learning, project planning, information visualization, programming, and entertainment [8], but do not yet have computational support to encourage collaboration.

Building a Coaching System to encourage collocated collaboration requires modeling the group it will support. The observational study that we executed also helped us to identify the elements of the group that the coaching system should consider to determining the social interventions that encourage collaboration.

In this paper, we propose a model based on a sociogram to define the degree of Group Cohesion in a collaborative activity according to the number of interactions performed by each member. Six groups of students working in a collocated collaborative activity in a TUI were modeled, based on the results of a study where the interactions of each group were observed. Considering the results obtained, we believe that this model could be used so that in the future a coaching system analyzes the cohesion of a group and decides on the appropriate social interventions to have better outcomes of the collaboration.

The paper is structured as follows, in Section Two the important terms and the related work are described; in Section Three the method of observational study is explained; in Section Four the results are described; in Section Five the proposed model is presented; in Section Six the discussion; and finally, the conclusions and future work are shown in Section Seven.

2. Background

2.1 Collocated Collaborative Learning

Collaborative learning is the educational approach whose objective is to improve learning through joint activities. Groups of two or more students work together to solve problems, complete tasks, or learn new concepts. When students are doing the activity in the same place at the same time, it is collocated collaborative learning.

The most relevant element in a collaborative activity is the Interaction, actions carried out by each participant in a shared workspace. According to Van Leeuwen et al. [4], in a learning activity there are Interactions that support the achievement of the task properly, for example, writing a note, putting an object, drawing, talking, transferring objects, among others; and other interactions whose goal is to guide collaboration.

In a common classroom, to support a collocated collaborative learning activity, Teacher must pay attention to Interactions. For example, a student place a new object, two students converse to agree, one student transfers a paper object to another student and so on. Then, teacher must analyze if there are any problems or facts that interfere with the student to act efficiently. For example: Students do not follow assigned roles or tasks, students interfere with the actions of others, students' participation is unbalanced etc. After the teacher is aware of the problem, his next task is to execute a Social Intervention, an expression that the teacher makes to students to guide collaborative learning. These can be expressed as different types of media, for example, interrogation-type expressions, diagnoses, indications, suggestions, explanations, instructions, or feedback [4].

2.2 Co-located Collaboration CSCW support

Table. 1 Role of technology in supporting co-located collaboration

Role Technology	**Social design objective**	**Related Works**
Enable	Enabling or allowing interaction to take place.	Schneider et al., 2012 [9]; Hamidi et al., 2012 [10]; Sylla, 2013 [11]; Antle et al., 201 [12]; Leversund et al., 2014 [13]; Waje et al., 2016 [14]; Baranauskas and Posada, 2017 [15]; Wallbaum et al., 2017 [16]; Melcer and Isbister, 2018 [17].
Facilitate	Support users' communication and coordination, relieving tension and minimizing	Nacenta et al., 2010 [18] ; Doeweling et al., 2013 [19]; Wise et al., 2015 [20]; Xambó et al., 2013 [21]; Cherek et al., 2018 [22]; Huber et al., 2019 [23]; Koushik et al., 2019 [24]

	negative experiences, and generally helping to make the best of a social situation.	
Invite	Providing additional information to users about their current social situation, so that they freely decide whether to act based on the additional information provided and social cues or not.	Morris et al., [25]; ter Beek et al., 2005 [26]; Kim et al., 2008 [27]; Cherek et al., 2018 [22]; Cepero et al., 2021 [28].
Encourage	Motivating or persuading people to start interacting or maintain an ongoing interaction.	Martinez-Maldonado et al., 2019 [2]; Praharaj, 2019 [29].

For Olsson et al. [5], computer systems can assume four roles in co-located collaboration: enabling, facilitating, inviting, and encouraging interaction. Enabling interaction refers to the role of a technological tool in enabling or allowing interaction to take place. Facilitating interaction refers to facilitating users' communication and coordination, relieving tension, and minimizing negative experiences, and generally helping to make the best of a social situation. Inviting interaction refers to providing additional information to users about their current social situation, so that they freely decide whether to act based on the additional information provided and social cues or not. And finally, encouraging interaction consists of motivating or persuading people to start interacting or maintain an ongoing interaction; For this, computational features must be used that stimulate people to take action, for example, technology could make an intervention when a person does not dare to say something to another person, or it could encourage two strangers to collaborate on something in which they seem to have a common interest.

In the literature, works have been found on how co-located collaboration has been supported and they have been classified according to the roles of technology proposed by Olsson et al. [5], as shown in Table 1. In the case of the role of enabling interaction, a wide range of examples of interactive systems such as TUIs have been found, and all these systems allow co-located interaction for learning activities, however, they do not fulfill the rest of the functions to support interaction.

Although the related works have made important contributions in supporting co-located collaboration, we believe that support for collaboration could be broader. The works where collaboration is invited through group awareness tools have helped to show users their current situation in the activity so that they decide how to act accordingly, however, our goal is that decisions about the collaboration does not fall only on students who sometimes do not know how to collaborate effectively, and instead users receive well-planned recommendations through interventions that suggest a change in their behavior, as a teacher would in a classroom.

2.3 Coaching Systems

The techniques that have been used to encourage collaboration have been through Tutoring Systems, Orchestration Tools or through Guide System interventions with personal agents such as a Coach [6]. Guiding systems perform all the phases in the collaboration management process and propose remedial actions to help the learners. The desired model of interaction and the system's assessment of the current state are typically hidden from the students. The system uses this information to make decisions about how to moderate the group's interaction [30].

In particular, the Coaching Systems observe the interaction of the users and provide them with suggestions, help and/or comments that help improve collaboration while they try to complete their

activity [7]. Coaching mechanisms monitor group activity and make some interventions if it detects any anomalies in the collaboration.

Coaching is a technique in which the instructor observes students and provides hints, help and feedback while they try to complete a task. Since students often miss learning opportunities and get stuck on a certain level of proficiency, a coach can make students aware of further possibilities, provide unobtrusive assistance and create potential learning experiences that will improve individual's development. The coach's goal is to promote group-learning interactions and maintain balanced participation. The design of the coach was based on socio-cognitive and cognitive dissonance theories [7].

An example of a remote collaboration coaching system is COLLER which helps students collaborate while solving entity relationship modeling problems in a computer-mediated learning environment. This paper evaluates a new approach to supporting collaboration that identifies learning opportunities based on differences in problem solving and tracking engagement levels [7]. And an example of a coaching system in a collocated activity is described in the work of Praharaj, 2019 [28], in which they propose an automated feedback system in real time using audio signals to analyze the group and facilitate collaboration.

3. Method

To identify possible problems that could arise in a collocated collaborative activity and how it could be computationally supported, groups of students working in a co-located collaborative activity in a system with TUI were observed.

3.1 Participants

For the study it was necessary to recruit students from the User Interface Design Class. In total there were 16 students between 20 and 26 years old. Six groups were observed. Two groups with 2 students and four groups with 3 students.

All students had prior knowledge of web interface design. The students knew each other previously and worked on group projects prior to this study; however, the design activity for this observational study was previously unknown for all of them. All students signed a consent form to participate in the study and record their group interaction with a video camera.

Four observers also participated. They observed and recorded the interactions of the students during the activity. Two of the observers have knowledge and experience in collaborative systems design and software evaluation, and the other two have knowledge and experience in systems design. And in addition, an assistant participated, explaining the conditions and instructions of the study and attend to doubts and technical complications that the students had.

3.2 PaperTUI

The system used for this study was PaperTUI [31]. It is a paper-based system to design web interfaces (see Fig. 1). It has an interactive surface with a screen placed horizontally on a table and a video camera placed on top of the screen that is responsible for capturing user interactions.

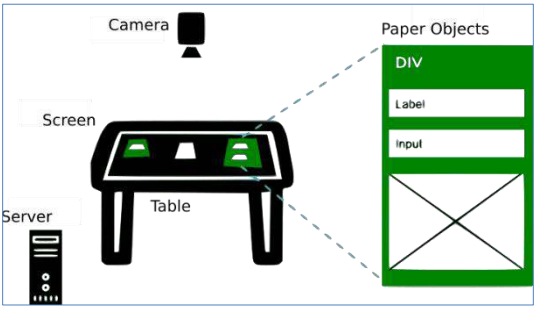


Fig. 1. PaperTUI outline (Source: [31])

Data entry in PaperTUI are paper tangible objects. These objects represent elements of a graphical web interface, such as images, video, text areas, maps, combo boxes, input fields, search fields, cancel or accept buttons, and some icons. The system must recognize the objects on the surface-screen and store the information in order to generate a HTML digital abstract web interface. Users receive feedback through video projection on the screen. Some of the feedback given to the user are projected circles below the recognized widgets and the HTML representation of the generated web interface.

PaperTUI was programmed in Python and uses Computer Vision libraries such as OpenCV and Tesseract. Its architecture is presented on Fig. 2, it is composed of a recognizer that detects objects through the detection of colors, images and text; and a generator model that creates an XML file that digitally represents the identified papers and then parses the XML to an HTML file that functions as a preview.

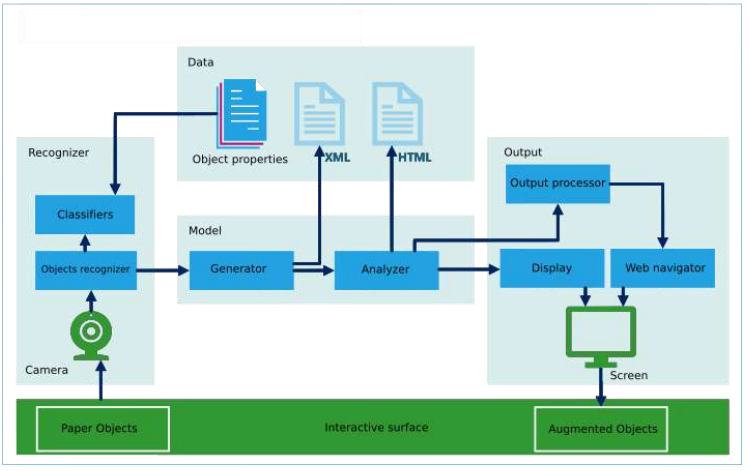


Fig. 2. PaperTUI architecture (Source: [31])

3.3 Process

Each group of people worked on a different session recorded with a video camera. In each session, the study objective was explained to the participants and they were given a consent form to record their interactions with a video camera. The task defined for the study was: “Design a user interface for a web page of a software development company”. No subtasks were specified. The students had 5 minutes to do the task on PaperTUI. Participants stood around the PaperTUI table and had 15 minutes to design the interface. On one side of the table, the video camera was mounted to record all interactions for later analysis.

During the activity, an assistant participated to answer technical questions. Observers take note of all the interactions they identified. It should be noted that in the observational studies, the students worked without the definition of instructions that would regulate their social activity. They were simply asked to create a web page for a computer equipment sales company and the system was explained to them. Fig. 3 shows two groups executing the collaborative activity in PaperTUI. After the sessions, an observer played the videos, and analyzed the participant's interactions.

Fig. 3. Group of students interacting with PaperTUI

4. Results

It was observed 19 types of interactions: 1) Put object, 2) Take object, 3) Move object, 4) Point object, 5) Show object, 6) Look object, 7) Find object, 8) Look Workspace, 9) Take Workspace, 10) Have object e, 11) Leave object, 12) Conversation, 13) Diffusion, 14) Deictic references, 15) Transfer object, 16) Request Validation, 17) Show object, 18) Get Resource and 19) Reserve object.

The interactions that each participant executed in each group were counted. From this data, the dispersion of the number of interactions per group was determined. This was calculated from the standard deviation of the set of numbers of interactions executed by each participant. The type of interactions that were considered in the count were: Put object, Take object, Move object, Point object, Conversation, Diffusion, Transfer Object and Request Validation. Since these were the interactions that had the most repetitions.

Table. 2. Group Collaboration results

Group	**Interaction Dispersion**	**# Objects**	**Time (**minutes)
1	7.39	20	04:58
2	12.88	27	07:53
3	7.97	12	09:40
4	6.13	12	07:16
5	10.61	24	09:57
6	2.12	19	06:50

In addition, the collaboration of each group was analyzed considering the number of objects placed in the web prototype and the activity time. These data are shown in Table 2.

From the count of the interactions and the observation of the behavior of the groups, communication and cohesion problems were identified:

Communication problems. It was observed that the participants asked questions and requested approvals, but sometimes the other participants did not respond. There were also long episodes in which the participants did not speak to each other; even in some groups, despite having three participants, their number of conversations and diffusion interactions was low.

Cohesion problems. In some groups there was never a transfer of figures, this could mean that the team was not as integrated or that they worked alone. Likewise, in table 2 it can be seen that in most of the teams the dispersion of the number of interactions is high (group 1, 2, 3 and 5), this means that some students participated more than others.

5. Model for Group Coaching

Building a Coaching System to encourage co-located collaboration requires modeling the group it will support. The observational study that we executed also helped us to identify the elements of the group that the coaching system should consider to define the social interventions that can be launched to encourage collaboration.

5.1 Proposal of a model based on Group Cohesion

In order to support collaboration through interventions in Coach Systems, it is necessary to understand the group and its behavior. Each group is different, they have different sizes and forms, and researchers who have tried to define groups have focused on different fundamental properties of groups, such as similarity, interdependence, entitativity, social identity, leadership, among others [32, 33]. To understand groups from so many different perspectives is a challenge for those trying to analyze it.

Cohesion is a measure that allows to analyze the meaning of groups. Specifically, cohesion is the measure to which members of a group bond with each other in trying to achieve a goal [34]. Groups can have different levels of cohesiveness among members over time. Group cohesion influences the effectiveness of collaboration.

A Cohesive Group means that you have more frequent, less inhibited, and enhanced interactions [35]. To increase cohesion, a necessary element is to motivate people to participate [36]. A participation means the execution of an interaction. Modeling the group in terms of cohesion means representing the links between the members of the group. For us, these links are the interactions executed by members of the group. One way to represent these links is through a sociogram. Sociograms are diagrams that allow graphically explaining the position that each individual occupies within the group, as well as all the interrelationships established between the various individuals [37, 38].

The proposed group model considers the following facts. A Group is a set of Actors trying to achieve a specific goal through the execution of Interactions. An Actor is a member of the group, which can be a person or software agent that executes Interactions to handle or produce objects in a shared workspace. An Interaction represents a running action in the shared workspace. Several elements of MARs Model [39] were considering (e.g. Actor, Interaction, Object, etc.).

The representation of Cohesion of the Group is through a sociogram. In the diagram of a sociogram, the cycle point represents an Actor. When an Actor executes an Interaction, an arrow will be added on the line, and it represents the direction of the interaction (another Actor). The number on the line is used for indicating the number of Interactions that the Actor has executed. A simple graphical representation of this sociogram is shown in Fig. 4. That a group is cohesive means that the numbers of the Interactions executed by all the actors is balanced.

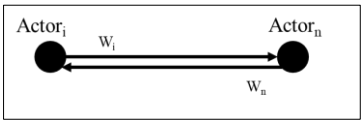


Fig. 4. Simple graphical representation of Group Cohesion Model

Formally, this sociogram can be represented as a weighted directed graph, where each vertex represents an Actor, and each edge represents the link between the actors that has been produced through Interactions. Each vertex has an associated weight based on the number of Interactions that the Actor has executed and that have linked it to the rest of the Actors.

This graph can be defined as an ordered trio

$$G\pm = (A, L, W)$$

where

$$A = \{a_1, \dots, a_n\}$$

is a set of vertices that are the actors,

$$L = \{l_1, \dots, l_n\}$$

is a set of edges that represent the link between two actors, and

$$W = \{w_1, \dots, w_n\}.$$

W is a set of weights associated with each edge that represents the number of Interactions that each Actor executes. Likewise, W can also be represented as a weight assignment function $w: L \rightarrow \mathbb{R}$ so that for any vertex $l \in L$, its weight is $w(L) \rightarrow \mathbb{R}$. Since each weight is restricted to a subset of the natural numbers.

5.2 Instances of the proposed model

The 6 observed groups were modeled with the proposed group model. For this, in each group each actor, the link of each actor with the rest of the actors, and the weight of the link were modeled. The latter, calculated from the number of interactions that each actor executed towards another actor. The type of interactions that were considered in the count were: Put object, Take object, Move object, Point object, Conversation, Diffusion, Transfer Object and Request Validation. Since these were the interactions that had the most repetitions.

In the following subsections, the modeling of each observed group will be described.

5.2.1 Group One

The cohesion model of group one is presented in Fig. 5.

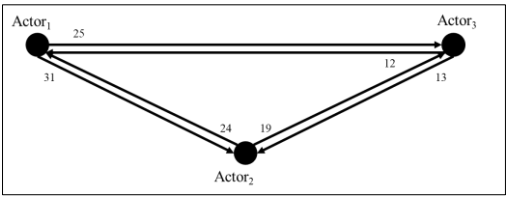


Fig. 5. Cohesion model within group one

The group one is a set of 3 student actors. The most active actor was Actor$_1$, who executed 31 interactions towards Actor$_2$ and 25 interactions towards Actor$_3$; followed by Actor$_2$, who executed 24 interactions towards Actor$_1$ and 19 interactions towards Actor$_3$; finally, Actor$_3$ executed 12

interactions towards $Actor_1$ and 13 interactions towards $Actor_2$. The left side view of the final prototype of group one is shown in Fig. 6. The number of objects placed for the prototype was 20, and the time of the activity was 04:58 minutes.

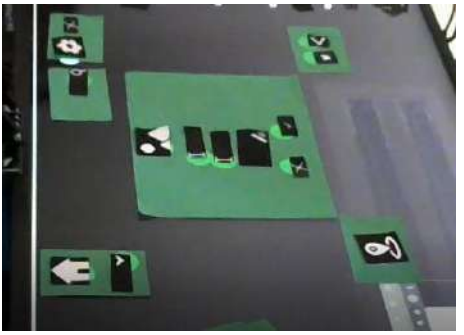


Fig. 6. Final prototype on PaperTUI of group one

5.2.2 Group Two

The cohesion model of group two is presented in Fig. 7.

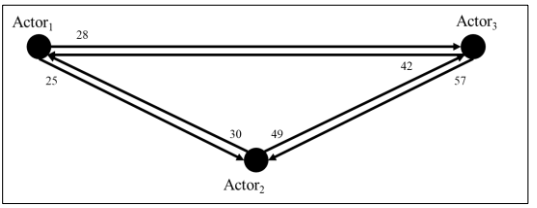


Fig. 7. Cohesion model within group two

The group two is a set of 3 student actors. The most active actor was $Actor_3$, who executed 42 interactions towards $Actor_1$ and 57 interactions towards $Actor_2$; followed by $Actor_2$, who executed 30 interactions towards $Actor_1$ and 49 interactions towards $Actor_3$; finally, $Actor_1$ executed 25 interactions towards $Actor_2$ and 28 interactions towards $Actor_3$. The left side view of the final prototype of group two is shown in Fig. 8. The number of objects placed for the prototype was 27, and the time of the activity was 07:53 minutes.

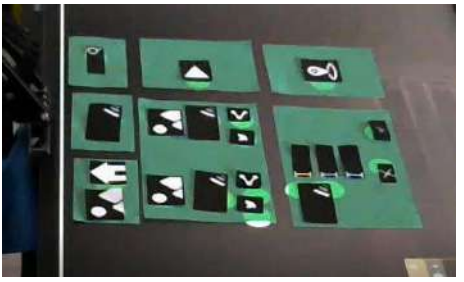


Fig. 8. Final prototype on PaperTUI of group two

5.2.3 Group Three

The cohesion model of group three is presented in Fig. 9.

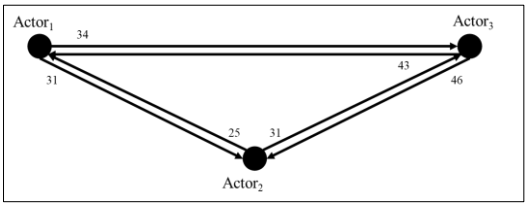


Fig. 9. Cohesion model within group three

The group three is a set of 3 student actors. The most active actor was $Actor_3$, who executed 43 interactions towards $Actor_1$ and 46 interactions towards $Actor_2$; followed by $Actor_1$, who executed 31 interactions towards $Actor_2$ and 34 interactions towards $Actor_3$; finally, $Actor_2$ executed 25 interactions towards $Actor_1$ and 31 interactions towards $Actor_3$. The left side view of the final prototype of group three is shown in Fig. 10. The number of objects placed for the prototype was 12, and the time of the activity was 09:40 minutes.

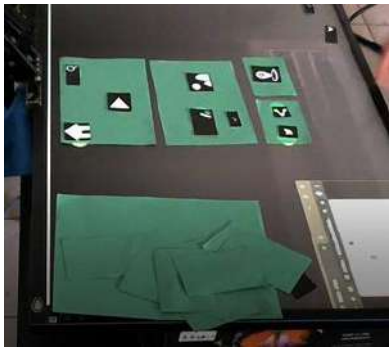


Fig. 10. Final prototype on PaperTUI of group three

5.2.4 Group Four

The cohesion model of group four is presented in Fig. 11.

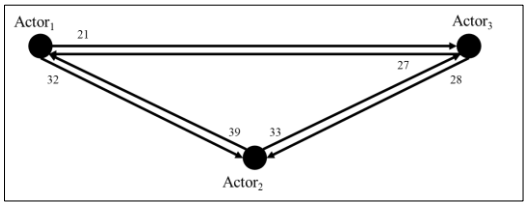


Fig. 11. Cohesion model within group four

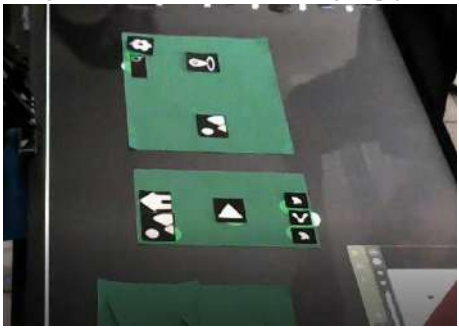


Fig. 12. Final prototype on PaperTUI of group four

The group four is a set of 3 student actors. The most active actor was $Actor_2$, who executed 39 interactions towards $Actor_1$ and 33 interactions towards $Actor_3$; followed by $Actor_1$, who executed 32 interactions towards $Actor_2$ and 21 interactions towards $Actor_3$; finally, $Actor_3$ executed 27 interactions towards $Actor_1$ and 28 interactions towards $Actor_2$. The left side view of the final prototype of group four is shown in Fig. 12. The number of objects placed for the prototype was 12, and the time of the activity was 07:16 minutes.

5.2.5 Group Five

The cohesion model of group five is presented in Fig. 13.

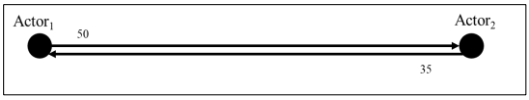


Fig. 13. Cohesion model within group five

The group five is a set of 2 student actors. The most active actor was $Actor_1$, who executed 50 interactions towards $Actor_2$; and $Actor_2$ executed 35 interactions towards $Actor_1$. The left side view of the final prototype of group five is shown in Fig. 14. The number of objects placed for the prototype was 24, and the time of the activity was 09:57 minutes.

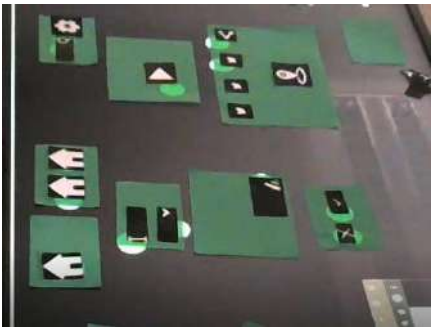


Fig. 14. Final prototype on PaperTUI of group five

5.2.6 Group Six

The cohesion model of group six is presented in Fig. 15.

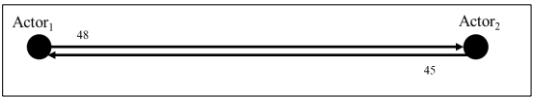


Fig. 15. Cohesion model within group six

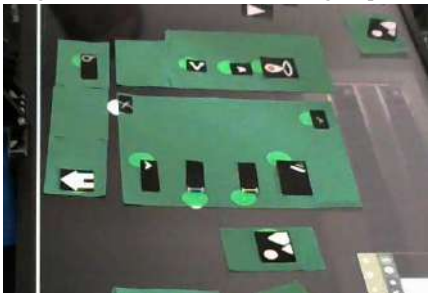


Fig. 16 Final prototype on PaperTUI of group six

The group six is a set of 2 student actors. The most active actor was $Actor_1$, who executed 48 interactions towards $Actor_2$; and $Actor_2$ executed 45 interactions towards $Actor_1$.

The left side view of the final prototype of group six is shown in Fig. 16. The number of objects placed for the prototype was 19, and the time of the activity was 06:50 minutes.

Table 2 shows the value of the instances of each group. Column 1 shows the group number, column 2 shows the number of actors, column 3 shows the number of links between the actors, and columns 4-9 show the value of the weight of each link.

Table 3. Group Cohesion model instance

Group	# Actors	# Links	Link weight					
			W_1	W_2	W_3	W_4	W_5	W_6
1	3	6	31	25	24	19	12	13
2	3	6	25	28	30	49	42	57
3	3	6	31	34	25	31	43	46
4	3	6	21	32	39	33	27	28
5	2	2	50	35	-	-	-	-

6. Conclusions and Future Work

We are interested in developing systems that support collaboration on co-located technologies. We executed an observational study to analyze the problems that might arise in groups collaborating on a co-located activity in a system with a Tangible User Interface. From this study, we identified some problems of communication and Group Cohesion.

Computational support for collocated collaboration can take various forms, in particular we commit to encouraging interaction between group members through a coaching system that implements social interventions. For this, it is necessary to model the group that the system will support and decide on the appropriate interventions to have an effective collaboration.

This paper proposed a model based on a sociogram to define the degree of cohesion of groups in a collaborative activity according to the number of interactions executed by each member of the group.

The goal is to have a Coaching System that automatically supports collaborative student learning as a teacher would regularly do in a classroom. For this, the system must identify the interactions of the students who carry out the activity, through the recognition of images by camera; then analyze interactions and take action to support collaboration. To analyze the interaction, the Coaching System would rely on the proposed group model, which helps determine the level of group cohesion. If the Coaching System identifies that the level of group cohesion is low, then it would launch a social intervention to remind the group to collaborate equally.

The design and implementation of this Coaching System is a future work. Likewise, carry out other observational studies with a larger number of groups and in other contexts.

References

[1] Schuman Sandy, ed. Creating a culture of collaboration: The International Association of Facilitators handbook. John Wiley & Sons, 2006, 536 p.

[2] Ellis C.A., Gibbs S.J., & Rein G. Groupware: some issues and experiences. Communications of the ACM, vol. 34, issue 1, 1991, pp. 39-58.

[3] Olsson T., Jarusriboonchai P. et al. Technologies for enhancing co-located social interaction: review of design solutions and approaches. Computer Supported Cooperative Work (CSCW), vol. 29, issue 1, 2020, pp. 29-83.

[4] Van Leeuwen A., Janssen J. et al. Teacher interventions in a synchronous, co-located CSCL setting: Analyzing focus, means, and temporality. Computers in Human Behavior, vol. 29, issue 4, 2013, pp. 1377-1386.

[5] Martinez-Maldonado R., Kay J. et al. Co-located collaboration analytics: Principles and dilemmas for mining multimodal interaction data. Human–Computer Interaction, vol. 34, issue 1, 2019, pp. 1-50.
[6] Jermann P., Soller A., & Muehlenbrock M. From mirroring to guiding: A review of the state of art technology for supporting collaborative learning. International Journal of Artificial Intelligence in Education, vol. 15, issue 4, 2005, pp 261-290.
[7] Constantino-Gonzales M.A., & Suthers D.D. Coaching Collaboration by Comparing Solutions and Tracking Participation. In Proc. of the First European Conference on Computer-Supported Collaborative Learning, 2001, pp. 173-180.
[8] Shaer O., & Hornecker E. Tangible user interfaces: past, present, and future directions. Foundations and Trends® in Human–Computer Interaction, vol. 3, no. 1-2, 2010, pp 4-137.
[9] Schneider B., Blikstein P., and Mackay W. Combinatorix: a tangible user interface that supports collaborative learning of probabilities. In Proc. of the 2012 ACM International Conference on Interactive Tabletops and Surfaces, 2012, pp. 129-132.
[10] Foad H., Baljko M. et al. Synchrum: a tangible interface for rhythmic collaboration. In Adjunct Proc. of the 25th Annual ACM Symposium on User Interface Software and Technology. 2012, pp. 63-64.
[11] Sylla C. Designing a tangible interface for collaborative storytelling to access 'embodiment' and meaning making. In Proc. of the 12th International Conference on Interaction Design and Children. 2013, pp. 651-654.
[12] Antle A.N., Wise A.F. et al. Youtopia: a collaborative, tangible, multi-touch, sustainability learning activity. In Proc. of the 12th International Conference on Interaction Design and Children. 2013, pp. 655-658.
[13] Leversund A.H., Krzywinski A., and Chen W. Children's collaborative storytelling on a tangible multitouch tabletop. Lecture Notes in Computer Science, vol. 8530, 2014, pp. 142-153.
[14] Waje A., Tearo K. et al. Grab this, swipe that: Combining tangible and gestural interaction in multiple display collaborative gameplay. In Proc. of the 2016 ACM International Conference on Interactive Surfaces and Spaces. 2016, pp. 433-438.
[15] Baranauskas M.C.C., and Gutiérrez Posada J.E. Tangible and shared storytelling: Searching for the social dimension of constructionism. In Proc. of the 2017 Conference on Interaction Design and Children. 2017, pp. 193-203.
[16] Wallbaum T., Ananthanarayan S. et al. Towards a tangible storytelling kit for exploring emotions with children. In Proc. of the on Thematic Workshops of ACM Multimedia 2017, pp. 10-16.
[17] Melcer E.F. and Isbister K. Bots & (Main) frames: exploring the impact of tangible blocks and collaborative play in an educational programming game. In Proc. of the 2018 CHI Conference on Human Factors in Computing Systems. 2018, pp. 1-14.
[18] Nacenta M.A., Pinelle D. et al. Individual and group support in tabletop interaction techniques. In Tabletops – Horizontal Interactive Displays. Human-Computer Interaction Series. Springer, London, 2010, pp. 303-333.
[19] Doeweling S., Tahiri T. et al. Support for collaborative situation analysis and planning in crisis management teams using interactive tabletops. In Proc. of the 2013 ACM International Conference on Interactive Tabletops and Surfaces, 2013, pp. 273-282.
[20] Wise A.F., Antle A.N. et al. What kind of world do you want to live in? Positive interdependence and collaborative processes in the tangible tabletop land-use planning game Youtopia. In Proc. of the Computer Supported Collaborative Learning Conference (CSCL 2015), 2015, pp. 236-243.
[21] Xambó A., Hornecker E. et al. Let's jam the reactable: Peer learning during musical improvisation with a tabletop tangible interface. ACM Transactions on Computer-Human Interaction (TOCHI), vol. 20, issue 6, 2013, article no. 36, pp. 1-34.
[22] Cherek C., Brocker A. et al. Tangible Awareness: How Tangibles on Tabletops Influence Awareness of Each Other's Actions. In Proc. of the 2018 CHI Conference on Human Factors in Computing Systems. 2018, paper no. 298, pp. 1-7.
[23] Huber S., Berner R. et al. Tangible objects for reminiscing in dementia care. In Proc. of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction. 2019, pp. 15-24.
[24] Koushik V., Guinness D., and Kane S.K. Storyblocks: A tangible programming game to create accessible audio stories. In Proc. of the 2019 CHI Conference on Human Factors in Computing Systems. 2019, paper no. 492, pp. 1-12.
[25] Morris M.R., Cassanego A. et al. Mediating group dynamics through tabletop interface design. IEEE computer graphics and applications, vol. 26, issue 5, 2006, pp. 65-73.

[26] Ter Beek M.H., Massink M. et al. A case study on the automated verification of groupware protocols. In Proc. of the 27th International Conference on Software Engineering, 2005, pp. 596-603.
[27] Kim T., Chang A. et al. Meeting mediator: enhancing group collaboration using sociometric feedback. Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work, 2008, pp. 3183-3188.
[28] Cepero T., Montané-Jiménez L.G., and Toledo-Toledo G. Visualization Technologies to Support Decision-Making in City Management. Programming and Computer Software, vol. 47, issue 8, 2021, pp. 803-816.
[29] Praharaj S., Scheffel M. et al. Group Coach for Co-located Collaboration. European Conference on Technology Enhanced Learning. Lecture Notes in Computer Science, vol. 11722, 2019, pp. 732-736.
[30] Soller A., Martínez A. et al. From mirroring to guiding: A review of state of the art technology for supporting collaborative learning. International Journal of Artificial Intelligence in Education, vol. 15, issue 4, 2005, pp. 261-290.
[31] Paredes Rendón P. A. PaperTUI: Sistema tangible para el prototipado de interfaces gráficas de usuario. Doctoral dissertation. Universidad Veracruzana. Facultad de Estadística e Informática, 2020, 225 p. (in Spanish).
[32] Stangor C. Social groups in action and interaction. Routledge, 2015, 454 p.
[33] Fjermestad J., Hiltz S.R. Group support systems: A descriptive evaluation of case and field studies. Journal of Management Information Systems, vol. 17, issue 3, 2000, pp. 115-159.
[34] Carron A.V. Cohesiveness in sport groups: Interpretations and considerations. Journal of Sport Psychology, vol. 4, issue 2, 1982, pp. 123-138.
[35] Yoo Y., and Alavi M. Media and group cohesion: Relative influences on social presence, task participation, and group consensus. MIS Quarterly, vol. 25, no. 3, 2001, pp. 371-390.
[36] Rosas P. and Cristina C. Indicadores de cohesión grupal a considerar para su diagnóstico: estudio de un caso. Acta Odontológica Venezolana, vol. 59, no. 2, 2021, pp. 4-9 (in Spanish).
[37] Bautista E., Casas E. et al. Utilidad del sociograma como herramienta para el análisis de las interacciones grupales. Psicologia para América Latina, no. 18, 2009, 6 p. (in Spanish).
[38] Crespo P.T. and Antunes C. Predicting teamwork results from social network analysis. Expert Systems, vol. 32, issue 2, 2015, pp. 312-325.
[39] Mezura-Godoy C., Riveill M., and Talbot S. Mars: Modelling arenas to regulate collaborative spaces. Lecture Notes in Computer Science, vol. 2806, 2003, pp. 10-25.

Information about authors / Информация об авторах

Alessandra REYES-FLORES – Master in User-Centered Interactive Systems from the University of Veracruz in Mexico. Professor at the Faculty of Statistics and Informatics of the University of Veracruz. Main research interest: Human Computer Interaction, User eXperience (UX), Computer Support Collaborative Work (CSCW), Software Development, Mobile Development.

Алессандра РЕЙЕС-ФЛОРЕС – профессор факультета статистики и информатики. Основные научные интересы: взаимодействие человека с компьютером, пользовательский опыт, компьютерная поддержка совместной работы, разработка программного обеспечения, разработка мобильных приложений.

Carmen MEZURA-GODOY – PhD in Computer Science from the University of Savoie in France. Professor at the Faculty of Statistics and Informatics of the University of Veracruz in Mexico. Main research interests: Human-Computer Interaction, User eXperience-UX, Computer Support Collaborative Work, Visualization and Multiagent Systems.

Кармен МЕЗУРА-ГОДОЙ получила степень доктора компьютерных наук в Университете Савойи во Франции. Профессор факультета статистики и информатики Университета Веракруса в Мексике. Основные исследовательские интересы: взаимодействие человека и компьютера, компьютерная поддержка совместной работы, визуализация и мультиагентные системы.

Edgard BENÍTEZ-GUERRERO – PhD in Computer Science from the University of Grenoble in France. Professor at the Faculty of Statistics and Informatics of the University of Veracruz in

Mexico. Research interests: Human Computer Interaction, Artificial Intelligence, Collaborative Computing, Data Management and Visualization.

Эдгар БЕНИТЕС-ГЕРРЕРО – доктор компьютерных наук Гренобльского университета во Франции. Профессор факультета статистики и информатики Университета Веракруса в Мексике. Научные интересы: взаимодействие человека с компьютером, искусственный интеллект, совместные вычисления, управление данными и визуализация.

Luis Gerardo MONTANÉ-JIMÉNEZ – PhD in Computer Science graduated from the University of Veracruz in Mexico. Professor at the Faculty of Statistics and Informatics of the University of Veracruz. Main research interests: Computer-Supported Cooperative Work (CSCW), Visualization, Human-Computer Interaction, Context-Aware Computing and Videogame Development.

Луис Херардо МОНТАНЕ-ХИМЕНЕС – доктор компьютерных наук, профессор факультета статистики и информатики. Основные исследовательские интересы: совместная работа с компьютерной поддержкой, визуализация, взаимодействие человека и компьютера, контекстно-зависимые вычисления и разработка видеоигр.

DOI: 10.15514/ISPRAS-2022-34(3)-9

Model-Driven in Serios Games and Serious Games with User-Centered Design in the Last Decade: A Review

[1] *P.O. Silva-Vásquez, ORCID: 0000-0001-5785-745X <zS19019681@estudiantes.uv.mx>*
[1,2] *V.Y. Rosales-Morales, ORCID: 0000-0003-2890-3343 <vivrosales@uv.mx>*
[1] *E. Benítez-Guerrero, ORCID: 0000-0001-5844-4198 <edbenitez@uv.mx>*

[1] *Facultad de Estadística e Informática, Universidad Veracruzana,*
Av. Xalapa s/n, Obrero Campesina, 91020 Xalapa-Enríquez, México
[2] *Cátedras CONACYT,*
México

Abstract. This paper reviews the literature on automatic code generation of user-centered serious games. We decided to break the study in two parts: one study about serious games with model driven engineering, and another study about user-centered serious games. This paper presents an extension of a paper presented at CONISOFT 20 where a systematic review of 5 years old at the time of writing was presented exclusively. The systematic literature review conducted in this paper covers a decade of information from January 2012 to June 2022. The main objective is to know the literature that helps to mitigate the costs and time of software development in serious games. The overall conclusion is that there is still work to be done to combine serious user-centered games and automatic generation. This paper is a systematic review that identifies relevant publications and provides an overview of research areas and publication venues. In addition, Research perspectives were classified according to common objectives, techniques, and approaches. Finally, is presented point out challenges and opportunities for future research and development.

Keywords: user-centered design; automatic code generation; serious games; systematic literature review

For citation: Silva-Vásquez P.O., Rosales-Morales V.Y. and Benítez-Guerrero E. Model-Driven in Serios Games and Serious Games with User-Centered Design in the Last Decade: A Review. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 127-144. DOI: 10.15514/ISPRAS–2022–34(3)–9

Acknowledgements. This work was partially developed under the support of the National Council of Science and Technology (CONACYT-Mexico) in the scope of the project “Infraestructura para Agilizar el Desarrollo de Sistemas Centrados en el Usuario” (Cátedras, Ref. 3053). In addition, the authors thank CONACYT for the doctoral scholarship (number 395377) granted to the first author. We also thank the Universidad Veracruzana for the support in the development of this research.

Модельно-ориентированная разработка серьезных игр и серьезные игры с ориентированным на пользователя дизайном в последнее десятилетие: обзор

[1] *П.О. Сильва-Васкес, ORCID: 0000-0001-5785-745X <zS19019681@estudiantes.uv.mx>*
[1,2] *В.Я. Росалес-Моралес, ORCID: 0000-0003-2890-3343 <vivrosales@uv.mx>*
[1] *Э. Бенитес-Герреро, ORCID: 0000-0001-5844-4198 <edbenitez@uv.mx>*
[1] *Факультет статистики и информатики Университета Веракрус, Веракрус, Халапа, 91020, Мексика*
[2] *Национальный совет по науке и технологиям, Мехико, Мексика*

Аннотация. В этой статье представлен обзор литературы по автоматической генерации кода для серьезных игр, ориентированных на пользователя. Мы решили разделить исследование на две части: одна часть посвящена серьезным играм с модельно-ориентированным проектированием, а другая – серьезным играм, ориентированным на пользователя. Статья представляет собой продолжение статьи, представленной на CONISOFT 20, в которой был представлен систематический обзор литературы 5-летней давности к моменту написания. Систематический обзор литературы, проведенный в данной статье, охватывает десятилетие с января 2012 года по июнь 2022 года. Основная цель – выявить результаты, которые помогает снизить затраты и время на разработку программного обеспечения для серьезных игр. Общий вывод заключается в том, что еще предстоит проделать работу, чтобы объединить серьезные игры, ориентированные на пользователя, и автоматическую генерацию. В статье указаны соответствующие публикации, а также представлен обзор областей исследований и мест публикации. Кроме того, исследования были классифицированы в соответствии с общими целями, методами и подходами. Наконец, представлены проблемы и возможности для будущих исследований и разработок.

Ключевые слова: ориентированный на пользователя дизайн; автоматическая генерация кода; серьезные игры; систематический обзор литературы

Для цитирования: Сильва-Васкес П.О., Росалес-Моралес В.Я., Бенитес-Герреро Э. Модельно-ориентированная разработка серьезных игр и серьезные игры с ориентированным на пользователя дизайном в последнее десятилетие: обзор. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 127-144. DOI: 10.15514/ISPRAS–2022–34(3)–9.

Благодарности. Эта работа была частично выполнена при поддержке Национального совета по науке и технологиям (CONACYT-Mexico) в рамках проекта «Инфраструктура для оптимизации разработки систем, ориентированных на пользователя» (Cátedras, Ref. 3053). Кроме того, авторы благодарят CONACYT за докторскую стипендию (номер 395377), предоставленную первому автору. Мы также благодарим Университет Веракрус за поддержку этого исследования.

1. Introduction

Serious Games (SG) can be defined as digital games with educational objectives, and can be considered as an alternative and effective way to convey new knowledge to people [1]. SGs mix, with pedagogical principles, the engaging and motivational characteristics of video games (history, character design, game rules, to name a few [2]), and touch on a wide range of subjects such as science, healthcare, business practices, and history [3].

It is important to notice that a SG is a software where a student/player must do a learning task with ease of use and high levels of playability. This can be accomplished if User-Centered Design (UCD) is used. UCD is based on the needs and interests of the user so that the resulting products are useful, usable, and subsequently desirable [4]. It has been argued [4], [5], however, that most educational video games have been developed with greater emphasis on the educational aspect, losing the

effectiveness, playability, and immersion that can be achieved with UCD. The study of user centeredness in SGs is then important.

The increasing complexity of systems development creates the need for tools to improve productivity in terms of time, cost, and quality [5]. One approach is automatic code generation (ACG) from models, as in model-driven engineering (MDE). MDE introduces a paradigm shift as models become the basis for software development, maintenance, and evolution [6]. By focusing on models that specify systems rather than code, a higher level of abstraction is achieved, and automation of the development process is possible. MDE and ACG have been applied in several domains.

Considering the principles of MDE presented in the study [7], which is a paradigm that encompasses the set of methods, techniques and technologies aimed at building software faster and easier, through the development and transformation of models, software development is accelerated. In research study [6] the authors direct model-driven engineering, called as Model-Driven Game Development (MDGD), towards video games.

Our general work is then interested in (semi) automatic generation of user-centric serious games. It was necessary to analyze the works previous, so we conducted a preliminary systematic literature review conducted in this paper covers a decade of information from January 2012 to June 2022.

In this paper presented the analysis and classification of the state of the art, a compilation is presented that contemplates the parts of model-driven engineering and integrates the UCD for the development of SG from the various proposals found in the systematic literature search. Articles about SG, ACG and UCD were previously analyzed; some of them are mentioned below.

The document is structured as follows: Section 2 introduces the research method that has been followed. Section 3 presents the results of the first search string. Section 4 presents the results of the second search string. Section 5 discusses our findings. Section 6 presents challenges and opportunities for future works. Finally, section 7 concludes this paper.

2. Research Method

The systematic review of the literature is based on the guidelines advanced [8] and inspired by proposals such as that of [9].

Accordingly, the main objectives of this review were to answer the following questions:

- How many papers related to serious games that have used Model-driven for serious game development were proposed from January 2012 to June 2022?
- How many papers related to serious games that have used user-centered design were proposed from January 2012 to June 2022?
- What are the shortcomings of the models found for the development of serious games?

As stated before, in a first effort, a single search string was considered, but given its limited results, the decision was made to consider two search strings, which would have better results and would allow us to analyze, describe and classify the results. As stated above, the literature review was divided into two different search strings. The first one ["Serious Games" AND "Model-Driven" AND ("Engineering" OR "Development" OR "Architecture" OR "Code Generation")] was composed in this way as there may already be a wide range of definitions related to model-driven and we were concerned about discarding some of these. The second ["Serious Game" AND "User-Centered Design"] is simpler because only two areas were considered. This paper presents these two parts, in which a total of 170 papers were analyzed.

Inclusion and Exclusion criteria were used to select articles to be reviewed. These criteria helped us limit the search and meet the objectives of this research. If the documents did not meet the selection criteria, then they were excluded. A selection criterion was applied to narrow the search:

- Paper analyzes the articles published from January 2012 to June 2022.
- Paper published on scientific international journals.
- Papers in English.
- Paper focuses on serious games with user-centered design
- Paper focuses on the serious games that were developed using Model-Driven

With these selection criteria and with the search strings defined to obtain the most accurate results, the sequence of the information search is the presented below.

It was examined whether the article had specified its objective, problem, and solution. Finally, the results presented in the articles must be related to our search strings. Once the search was performed, the title, keywords and abstract of the papers were checked to ensure that they met the inclusion and exclusion criteria. They also had to be related to the search strings.

3. First Search String

3.1. Application of the method

Concerning the first question, as mentioned above, a detailed search in a journal database was conducted to obtain a complete bibliography, as mentioned in one of the criteria. In the first search string which is ["Serious Games" AND "Model-Driven" AND ("Engineering" OR "Development" OR "Architecture" OR "Code Generation")], which amounts to a total of 397 papers.

The criteria of the time (January 2012-june 2022) After excluding the results regarding the fact that they are only from journals, the total found was 75 of the 397.

To finalize the selection of the articles that we considered for this research, a quick search was carried out in titles, abstracts, and related key words, considering "Model-Driven", and most of all, "serious games" to discard the research that was not related to serious games. Leaving us only with 35 works directly related.

3.2. Quantitative Analysis

In this article presents 35 papers related to the search string ["Serious Games" and "Model-Driven" and ("Engineering" or "Development" or "Architecture" or "Code Generation")] nine online databases to search the articles were used. The results were as follows: a) Science Direct: 11, b) Springer Link: 13, c) Emerald: 1, d) Wiley Online Library: 2, e) ACM Digital Library: 3, f) SAGEPUB: 1, g) MDPI: 3 and h) Hindawi: 2. Based on these results, IEEE Xplore were discarded because they did not present results.

The journal articles were published in 19 journals related to the first search string. The journal "Multimedia Tools and Applications" published most of the articles related to our search string; that is four papers. "Entertainment Computing" contributed with three papers. While "International Journal of Computer Games Technology" and “Procedia Computer Science” contributed only with two papers. The remaining 24 journals also published papers related to the search string.

3.3. Qualitative analysis

The works found were classified into 3 categories, which are explained below. Although they fall withing the criteria of the search string, we are aware that they have a different approach to the problems they want to solve.

3.3.1. Model-Driven Engineering of Serious Games

Serious or playful games need a well-defined framework to develop them. That is why the alternative to implement the phases of conceptualization, application, and monitoring in the generated applications was taken. This helped them to focus only on what is crucial for the success of the learning strategies. In the classification phase, the guide of the model-driven engineering was made use. Among these, those using the Model-Driven Engineering were presented because specifically, the tool in these documents provides a detailed description of the proposed component-based model and it also presents a validation of the requirements obtained through the use of game activity, [10]–[15]. Likewise, some papers define themselves as using the Model-Driven Architecture since they argue that most educational games are not supported by specific architectures because the existing ones do not include fundamental aspects such as collaboration, adaptation, or playability, or their conceptual language is difficult to understand for the educational team. To fill this gap, the architectures for designing, executing, monitoring, and adapting the learning processes supported by video games are described, considering the design and customization aspects, [16]-[26].

Among the works reviewed, some considered following a Model-Driven Development, where the novelty is based on the complexity of the design of the games, seeking to facilitate the design of the final user. This model does not impose the cognitive overload of learning a new design language to describe game designs that can be exported to XML files, and a game engine capable of interpreting those files and automatically generating a serious game, [27] & [28].

Finally, In [29] and [30] the authors used the Model-Driven Framework, since it allows geolocation based games to be edited and deployed in many places quickly. The core models and represents the structure of the game and its multimedia content (e.g. video, 3D objects), while [31] and [32] present a Model-Driven Game, which serves to adapt the game design to the players' personality type. This improved the effectiveness of the games. The intention is to change the behavior and self-efficacy by changing the context concerning the player. Besides, it shows that the benefits of customizing the game improve the player's experience.

3.3.2. Application Domains

For this classification, works that have developed a serious game that was guided in the model-driven engineering are presented. In this classification, only the serious game with its characteristics is presented but details of the development are not described.

Educational. In [33] introduced the importance of making this type of software as an educational alternative for students, since some applications present it as a support for distance education that can raise the quality of education and student satisfaction. In the case of some, they present a tool that allows monitoring students and tracks their improvement while using the video game [34]–[36].

Rehabilitation. In [37] and [38] the system that has been developed with the main objective of improving the physical and cognitive skills of students with special needs is presented. The different activities are configurable, and the tutor can modify the settings according to the needs of the student. The activities are game oriented to attract the students' attention and motivate them to learn. It is highly interactive and encourages students to be active learners. The results showed that students will be able to use the computer while improving their digital competence and their cognitive and physical skills.

3.3.3. Evaluate Gameplay

The objective of this category is to involve the final user in the discussion of the use of the serious game. For this, the category investigations, such as [39] and [40], use diverse tools that can inform us of the observations of the user. For their evaluation, these two studies were introduced in the digital games, as they played with the application. Whatever their presentation in mobile, console

or pc, they discussed the motivations that the game offers them and the obstacles for the current game.

The increasing familiarity and age ranges play an important role in this type of classification. In most of the works it is concluded that the creation of a safe, comfortable and accessible space for learning must be considered for serious games, as a valuable tool for learning [35], [41]-[43]

Most of the research presented a model-driven engineering that helps serious game developers with tools that reduce development times and abstraction of concepts for serious games. Considering that a video game developer does not know the concepts that a serious game must have incorporated so that the users have an experience with playability.

4. Second Search String

4.1. Application of the method

Concerning the second question, as mentioned above, a detailed search to obtain a complete bibliography was conducted. In the second search string, which is ["serious games" AND "User-Centered Design"]. In total 616 articles were found.

The criterion of the time (January 2012 to June 2022) After eliminating those articles that were not published in journals, the total was reduced to 214 out of the 569.

To finalize the selection of the articles that were considered for this research, a quick search was conducted in titles, abstracts, and related words considering "User-Centered Design" and mainly "Serious Games", with the aim of discarding research that was not related to serious games. Leaving us only with 135 pieces of directly related to the topic.

4.2. Quantitative analysis

In this article presents 135 papers related to the search string ["Serious Game" and "User-Centered Design"]. nine online databases to search the articles were used. The results were as follows: a) Science Direct: 50, b) Springer Link: 55, c) Wiley Online Library:15, d) Emerald: 4, e) ACM Digital Library: 12, f) IEEE: 4, g) SAGEPUB: 1, h) MDPI: 2 and i) Hindawi: 1.

The journal articles are divided into 78 journals that published articles related to the second search string. The journal “Entertainment Computing” published most of the articles related to our search string, with eleven papers. “British Journal of Educational Technology”, "Journal of Ambient Intelligence and Humanized Computing", “Procedia Computer Science” comprises seven papers, “International Journal of Human-Computer Studies” comprises six papers each one while “Multimedia Tools and Applications” supplied four papers. There were 72 other journals that also published papers related to the search string.

4.3. Qualitative analysis

In the search 135 papers have a direct relationship with the topic. The works found were classified into four categories, which are explained below. Even though the criteria for the search string was observed, it can be ascertained that they have a different approach to the problems they want to solve.

4.3.1. Application Model or Framework

The papers in this classification presented a guide for the development of projects in serious games. In these investigations, the authors give their views on how a serious game should be developed. They describe the process of generating the project from the point of view of software engineering.

For the model, they make a graphic presentation of what they consider should compose the application, but they are not specific in the points. They leave the generation of the analysis of requirements and the development of the project, as well as the evaluation, it to the criterion of the developer. The papers that work it this way are [23], [44]-[53]

Those that present a framework provide a more specific guide of how it should be implemented, giving details of what the best practices are to determine whether the application will have the success that the developer seeks. In [54]-[72] the author explained the framework.

4.3.2. User Model

For the user model classification, these are investigations that consider that users have a particularity that does not allow them to use a serious game as engineers would design it for the public to which it is addressed.

It is interesting to talk about the users because most of them for whom the user models were generated are people who have a disease or people who want to know if they can be diagnosed with this disease, as such in [12], [37], [73]-[77] Likewise, there is a project [78] where the user model is oriented to a general aspect; for example, they take a single user "child" between "5-7" and with "kindergarten schooling", for the model of these users. Although the model is focused on the user, it does not provide specific details of the user. But this classification helps the developers to know the particularities and pay attention if they plan to generate an application where their target audience is children with these characteristics.

Similarly, we found these papers [79]-[133], that present an application that undergoes an evaluation with the target audience. They present a list of adjustments to achieve better usability. After this, they provide details of the model of the user and develop changes in the application to reevaluate and contrast the new changes with the list of requests, ending up with the requirements of the user, as the final part of the project.

4.3.3. Application Domains

In this classification are the papers that aim to use the tool to achieve an objective in a specific area.

Learning: They are all those that have the objective of being before a specific user so that the person who uses it obtains information that can later be considered acquired knowledge. This is true of [74]-[85], [90], [92], [94], [98], [100], [101], [105], [108], [109], [111], [112], [115], [117], [119], [120], [129], [130], [134]-[144], where that is the principal

In some cases, they implemented an improvement of the serious game, and they re-submitted it to evaluation, to know if everything they considered adding was enough or if they had omitted some requirements again. As an example of this, we found [122]-[126].

For other cases, they evaluated some applications that lacked the consideration of user-centered design and advised that the list of requirements include improvements to the applications. These would improve the projects developed so that the user would not feel frustrated and would stop using the serious game as a tool for his benefit [145]-[152].

Rehabilitation: Works such as [132], [153], & [154] include projects that can be augmented with hardware to help people with a physical condition or that present a totally specific interaction for a user with a mental illness.

Diagnostic: Here all the applications such as [155]-[159] had the purpose of finding signs in the user. Whoever uses them presents particularities that can help him to know that he is a person who is suffering a condition or is prone to suffer it in a short period time.

Selecting: These applications or tools are useful to know if the user, who uses the application, has characteristics that the person who implements the tool is looking for. This is a way to evaluate the

knowledge of a person in a particular case, e.g. for a job or a subject [86], [91], [103], [110], [113], [114], [116], [118] & [120].

User-centered design: This tool gives us particularities of user-centered design. It teaches us to know if our project has user-centered design, as well as it shows us how it should be implement [70], [72] & [160]-[165].

Similarly, we can realize that some considerable overlapping still exists in the classification, as in some tools, before designing them, they modeled a user and advanced and used a model for the development of the application.

4.3.4. Evaluation of User-Centered Design

Several works decided to evaluate an application to know its user-centered design. For this, they conducted interviews with the public the application was directed to in order to know their point of view and how they considered their interaction, usability, and user experience. When they realized that they did not consider several things that the user required and that they had suggested to improve the usability, the application was modified.

5. Discussion

The combination of sound, art, control systems and artificial intelligence (AI) for a video game makes it totally different from traditional software development. However, software engineering techniques help game development achieve less effort and cost, and better design. The purpose of this study was to assess the state of research on software engineering processes for serious game development using an automatic code generation strategy, such as model-driven engineering. They have been submitted to user-centered design scrutiny, as well as highlighting areas that need further consideration by researchers.

In the first search string, several model-driven that are used for the development of applications were presented, but even though they can be a development tool that helps at the time of the creation of video games, they are still not so popular, and everyone chooses to start a serious game development from scratch, learning the programming language and designing the whole application. Game engines present a great development tool; however, the most sophisticated ones still require a knowledge of programming that can represent a challenge for a conventional user. It is considered that the tools that use model-driven and try to be useful do not have user-centered design patterns, making their use complicated.

Serious games are an alternative for educators. Research suggests that they provide knowledge to their end users, but tools are needed to facilitate their development. Therefore, it is considered necessary to explore new agile development tools that follow a model of automatic generation of serious games, but also consider the minimum requirements for the serious games that have a design focused on the user. In the analysis of user-centered serious games, these games remain specific to a particular user and context, which limits their contribution to new serious games that do not have those same features.

6. Recommendations for Future Research

Interesting areas of opportunity were discussed regarding agile SG development looking for a solution to the challenges for developing SGs: 1) developing better SGs, with 2) less budget and resources, and 3) with time [166]. This is where MDE and ACG can propose a solution, along with UCD.

Therefore, we present below the challenges, opportunities for research and the development of future work. Proposing to follow the bases of the MDE and the UCD. The deficiencies and research areas

where models need to be developed and a successful ACG that meets most of the requirements requested by the user needs to be developed.

Model-Driven Game Development (MDGD) [167] presents a structure of two essential parts. The platform independent models (PIM), the platform specific model (PSM), moving to the final part in the generation of the final source code, to perform the first evaluation of a prototype.

Consider as an example the structure proposed by [169] & [170] using UML to present the Model-driven. Presenting a structure that can be interpreted by the developer.

Model for functional and non-functional requirements of serious games. Some studies presented user models to solve end-user problems with positive conclusions from them. However, the problem with these studies presenting a user model is that they lack a guide to streamline the creation process from a requirements analysis. Functional and non-functional requirements that help us to describe the behavior of the SG for its functionality, and focus the change on the design or implementation, the functional requirements give priority to Learning Analytics, motivation, and types of games, while in the non-functional may be the Interoperability of learning tools, connectivity and scalability. To mention a few, work should be done to propose a guide that presents a greater diversity and that are related to the SG.

Task Model for Serious Games. The user task model helps to understand the activities performed by the user in the SG. This type of modeling is separated into four types of existing tasks: user tasks, application tasks, interaction tasks and abstract tasks. This model can be represented in a Concur Task Tree (CTT). But this model has not been integrated into the model proposals for SG development, which would provide developers with time-saving work.

User or player model for serious games. The problem that exists is that no proposal considers the integration of the UCD. The user model should describe the particularities of the target user, with an abstract representation of the information that we can know about the end user.

The player model is also considered where the role that the player has, the tasks to be performed and how to achieve them, the goals and the different states in which these goals of the SG are found. Therefore, we work on a user model that incorporates several techniques to know the characteristics of the final player and provide a UCD.

Model for game mechanics. The game mechanics would determine the rules of the video game, the definition of the rewards, the storytelling, the feedback, the debriefing, the objectives and mechanics of the game where the mobile and static actors (Player and enemies), the environment and design (scenarios), definition of the scoring rules, and hardware definitions are presented. In several works it is taken for granted that developers know the elements that a SG must have.

Model of pedagogical/playful objectives. It is necessary to work on what the teacher wishes to present in the SG and how it can be presented, since this aspect is not considered in the MDGD proposal. The teacher must determine the competencies to be achieved by the player. An example to define the model step 1 identification and description of the activity, step 2 representation of the game sequence, step 3 identification of the actions, tools and objectives of the game, description of the objective to be fulfilled with the exercise, to relate the mechanics of the game with the learning objective.

Model for scenarios and assets. A SG needs a model to define the levels and from these the scenes with a brief description, the characteristics of the playable or non-playable objects of the scene and the identification of the educational challenges. For the scenes to present their graphic part, with the graphic design sketches.

Stringent evaluations. The Evaluations they use in their research mention the success of serious games with end users with product acceptance and functionality, but do not subject the result of the serious game to strict evaluation by experts who might conclude that the serious game has playability. Some others conduct ad-hoc interviews with users to get their opinion. Consider in

evaluating the presentation of your developed product on various electronic devices (mobile, tablet, pc or game console) to determine which device is the most suitable for your SG. Therefore, it is necessary to work on an evaluation of the final product that can be provided in the MDGD. Therefore, it should not be left to drift to maintain the balance in the middle point of learning and entertainment so that a serious game can be had that provides the necessary elements for the student to continue using it and the teacher to see reflected the reinforcement of knowledge that he wants his students to obtain. The following are the final conclusions of the work presented in this academic article.

7. *Conclusion*

The research works reported above focus on streamlining the development of serious games, but they ignore an important part: the user-centered design, being the design the first thing that the user visualizes and should be more important. Because of this, even if the serious game meets all the functions and pedagogical requirements requested by the user, if the interaction does not have an appropriate design (user-centered), it becomes a poorly rated aspect by the end user. UCD is an approach to the design and development of interactive systems that seeks to make them more usable. UCD helps to determine whether the application is useful, usable and, subsequently, desirable [4]. However, in research study [168], they change the concept of usability to playability, which they define as the set of properties that describe the player's experience with a given game system whose main objective is to amuse and entertain in a satisfactory way It proposes a series of attributes, which are very similar to the aspects of usability.

The research compiles some of the information found shows the integration of various multidisciplinary points. Section 6 helps to clearly identify the user requirements and needs of user-centered design with the work found in the systematic review, as well as the integration of model-driven orientation, which provides a software engineering foundation that can serve as a guide.

We are working on user-centered educational applications to validate if the proposed methodology contributes to streamline the process to obtain applications that meet the criteria of usability and playability.

References / Список литературы

[1] Catalano C.E., Luccini A.M., Mortara M. Guidelines for an effective design of serious games. International Journal of Serious Games, vol. 1, issue 1, 2014, 13 p.

[2] Frasca G. Juego, videojuego y creación de sentido. Una introducción. Comunicación: revista Internacional de Comunicación Audiovisual, Publicidad y Estudios Culturales, vol. 1, no. 7, 2009, pp. 37-44 (in Spanish).

[3] Hanes L, Stone R. A model of heritage content to support the design and analysis of video games for history education. Journal of Computers in Education, vol. 6, issue 4, 2019, pp. 587-612. doi:10.1007/s40692-018-0120-2

[4] Nielsen J. Usability 101: Introduction to Usability. Nielsen Norman Group. Published 2012. URL: https://www.nngroup.com/articles/usability-101-introduction-to-usability/, accessed April 7, 2020.

[5] Blow J. Game Development: Harder Than You Think. Queue, vol. 1, no. 10, 2004, pp. 28-37.

[6] Reyno E.M., Á Carsí Cubel J. Automatic prototyping in model-driven game development. Computers in Entertainment, vol. 7, issue 2, 2009, pp. 1-9.

[7] Ruiz-Rube I., Dodero J.M., Ruiz M. Ingeniería Dirigida por Modelos como soporte a la gestión de procesos software. Trabajo incluido en las actas de las II Jornadas Predoctorales de la Escuela Superior de Ingenieria. 2010. 4 p.

[8] Kitchenham B., Pearl Brereton O. et al. Systematic literature reviews in software engineering - A systematic literature review. Information and Software Technology, vol. 51, issue 1, 2009, pp. 7-15.

[9] Hong J., Suh E., Kim S.J. Context-aware systems: A literature review and classification. Expert Systems with Applications, vol. 36, issue 4, 2009, pp. 8509–8522.

[10] Calderón A., Boubeta-Puig J., Ruiz M. MEdit4CEP-Gam: A model-driven approach for user-friendly gamification design, monitoring and code generation in CEP-based systems. Information and Software Technology, vol. 95, 2018, pp. 238-264.
[11] Pérez-Berenguer D., García-Molina J. A standard-based architecture to support learning interoperability: A practical experience in gamification. Software: Practice and Experience, vol. 48, issue 6, 2018, pp. 1238-1268.
[12] Teipel S., König A. et al. Use of nonintrusive sensor-based information and communication technology for real-world evidence for clinical trials in dementia. Alzheimer's & Dementia, vol. 14, issue 9, 2018, pp. 1216-1231.
[13] P. de Lope R., Medina-Medina N. et al. A novel UML-based methodology for modeling adventure-based educational games. Entertainment Computing, vol. 38, article no. 100429, pp. 1-19.
[14] Antunes A., Madeira R.N. PLAY - Model-based Platform to Support Therapeutic Serious Games Design. Procedia Computer Science, vol. 198, 2022, pp. 211-218.
[15] Pérez F., Lapeña R. et al. Topic modeling for feature location in software models: Studying both code generation and interpreted models. Information and Software Technology, 140, 2021, article no. 106676.
[16] Martínez-Pernía D., Núñez-Huasaf J. et al. Using game authoring platforms to develop screen-based simulated functional assessments in persons with executive dysfunction following traumatic brain injury. Journal of Biomedical Informatics, vol. 74, 2017, pp. 71-84.
[17] Oberdörfer S., Latoschik M.E. Predicting learning effects of computer games using the Gamified Knowledge Encoding Model. Entertainment Computing, vol. 32, article no. 100315, 39 p.
[18] Padilla-Zea N., Medina-Medina N. et al. PLAGER-VG: platform for managing educational multiplayer video games. Multimedia Tools and Applications, vol. 77, issue 2, 2018, pp. 2115-2152.
[19] Thomas A., Menassa C.C., Kamat V.R. Lightweight and adaptive building simulation (LABS) framework for integrated building energy and thermal comfort analysis. Building Simulation, vol. 10, issue 6, 2017, pp. 1023-1044.
[20] Reinkensmeyer D.J., Burdet E. et al. Computational neurorehabilitation: Modeling plasticity and learning to predict recovery. Journal of NeuroEngineering and Rehabilitation, vol. 13, issue 1, 2016, pp. 1-25.
[21] Torrens P.M. Intertwining agents and environments. Environmental Earth Sciences, vol. 74, issue 10, 2015, pp. 7117-7131.
[22] Muñoz J.E., Gouveia E.R. et al. Physiolab - A multivariate physiological computing toolbox for ECG, EMG and EDA signals: A case of study of cardiorespiratory fitness assessment in the elderly population. Multimedia Tools and Applications, vol. 77, issue 9, 2018, pp. 11511-11546.
[23] Kritikos K., Plexousakis D., Paternò F. Task Model-Driven Realization of Interactive Application Functionality through Services. ACM Transactions on Interactive Intelligent Systems, vol. 3, issue 4, 2014.
[24] Feron H., Lehmann A., Josse F. A generic architecture and validation considerations for tactical combat casualty care serious games. The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology, vol. 12, issue 3, 2015, pp. 319-334.
[25] Fanini B., Pagano A., Ferdani D. A novel immersive VR game model for recontextualization in virtual environments: The μVRmodel. Multimodal Technologies and Interaction, vol. 2, no. 2, 2018, article no. 20, 22 p.
[26] Mestadi W., Nafil K. et al. An Assessment of Serious Games Technology: Toward an Architecture for Serious Games Design. International Journal of Computer Games Technolog, vol. 2018, 2018, article no. 9834565, 17 p.
[27] Zarraonandia T., Diaz P., Aedo I. Using combinatorial creativity to support end-user design of digital games. Multimedia Tools and Applications, vol. 76, issue 6, 2017, pp. 9073-9098.
[28] Bozzon A., Fraternali P. et al. Modeling CrowdSourcing Scenarios in Socially-Enabled Human Computation Applications. Journal on Data Semantics, vol. 3, issue 3, 2014, pp. 169-188.
[29] Ferreira C., Maia L.F. et al. Modelling and transposition of location-based games. Entertainment Computing, vol. 30, 2018, article no. 100295.
[30] Van Rozen R. Languages of Games and Play: A Systematic Mapping Study. ACM Computing Surveys, vol. 53, issue 6, 2021, pp. 1-37.
[31] Orji R., Mandryk R.L., Vassileva J. Improving the efficacy of games for change using personalization models. ACM Transactions on Computer-Human Interaction, vol. 24, issue 5, 2017, Article no. 32, pp. 1-22.

[32] Aleem S., Capretz L.F., Ahmed F. Critical Success Factors to Improve the Game Development Process from a Developer's Perspective. Journal of Computer Science and Technology, vol. 31, issue 5, 2016, pp. 925-950.
[33] Minović M., Milovanović M. et al. Visualisation of student learning model in serious games. Computers in Human Behavior, vol. 47, 2015, pp. 98-107.
[34] Rumeser D., Emsley M. Design and evaluation of the project and program crashing games. Journal of Applied Research in Higher Education, vol. 14, issue 1, 2022, pp. 471-488.
[35] Predescu A., Arsene D. et al. A serious gaming approach for crowdsensing in urban water infrastructure with blockchain support. Applied Sciences, vol. 11, issue 4, 2021, pp. 1-32.
[36] Furtado L.S., de Souza R.F. et al. Teaching Method for Software Measurement Process Based on Gamification or Serious Games: A Systematic Review of the Literature. International Journal of Computer Games Technology, vol. 2021, 2021, article ID 8873997, 8 p.
[37] Ojeda-Castelo J.J., Piedra-Fernandez J.A. et al. KiNEEt: application for learning and rehabilitation in special educational needs. Multimedia Tools and Applications, vol. 77, issue 18, 2018, pp. 24013-24039.
[38] Yamin M.M., Katt B., Nowostawski M. Serious games as a tool to model attack and defense scenarios for cyber-security exercises. Computers & Security, vol. 110, 2021, article no. 102450.
[39] Tong T., Guana V. et al. Rapid Deployment and Evaluation of Mobile Serious Games: A Cognitive Assessment Case Study. Procedia Computer Science, vol. 69, 2015, pp. 96-103.
[40] Gibson D.C., Webb M.E. Data science in educational assessment. Education and Information Technologies, vol. 20, issue 4, 2015, pp. 697-713.
[41] Khalili-Mahani N., de Schutter B. et al. For Whom the Games Toll: A Qualitative and Intergenerational Evaluation of What is Serious in Games for Older Adults. The Computer Games Journal, vol. 9, 2020, pp. 221-244.
[42] Rieger C., Majchrzak T.A. Towards the definitive evaluation framework for cross-platform app development approaches. Journal of Systems and Software, vol. 153, 2019, pp. 175-199.
[43] Koch J., Gomse M., Schüppstuhl T. Digital game-based examination for sensor placement in context of an Industry 4.0 lecture using the Unity 3D engine - a case study. Procedia Manufacturing, vol. 55, 2021, pp. 563-570.
[44] Coghlan A., Carter L. Serious games as interpretive tools in complex natural tourist attractions. Journal of Hospitality and Tourism Management, vol. 42, 2020, pp. 258-265.
[45] Carvalho M.B., Bellotti F. et al. An activity theory-based model for serious games analysis and conceptual design. Computers & Education, vol. 87, 2015, pp. 166-181.
[46] Johnsen H.M., Fossum M. et al. Teaching clinical reasoning and decision-making skills to nursing students: Design, development, and usability evaluation of a serious game. International Journal of Medical Informatics, vol. 94, 2016, pp. 39-48.
[47] Kiili K., Lainema T. et al. Flow framework for analyzing the quality of educational games. Entertainment Computing, vol. 5, issue 4, 2014, pp. 367-377.
[48] Gray S.I., Robertson J. et al. BrainQuest: The use of motivational design theories to create a cognitive training game supporting hot executive function. International Journal of Human-Computer Studies, vol. 127, 2019, pp. 124-149.
[49] Marcucci E., Gatta V., le Pira M. Gamification design to foster stakeholder engagement and behavior change: An application to urban freight transport. Transportation Research Part A: Policy and Practice, vol. 118, 2017, pp. 119-132.
[50] Xu F., Buhalis D., Weber J. Serious games and the gamification of tourism. Tourism Management, vol. 60, 2017, pp. 244-256.
[51] Huang H., Ng K.H. et al. A card-based internet of things game ideation tool for museum context. Journal of Ambient Intelligence and Humanized Computing, vol. 12, issue 10, 2021, pp. 9229-9240.
[52] Vardaxoglou G., Baralou E. Developing a platform for serious gaming: Open innovation through closed innovation. Procedia Computer Science, vol. 15, 2012, pp. 111-121.
[53] de Freitas S., Routledge H. Designing leadership and soft skills in educational games: The e-leadership and soft skills educational games design model (ELESS). British Journal of Educational Technology, vol. 44, issue 6, 2013, pp. 951-968.
[54] van Dooren M.M.M., Siriaraya P. et al. Reflections on the design, implementation, and adoption of a gamified eHealth application in youth mental healthcare. Entertainment Computing, vol. 31, 2019, article no. 100305.

[55] Seaborn K., Fels D.I. Gamification in theory and action: A survey. International Journal of Human-Computer Studies, vol. 74, 2015, pp. 14-31.
[56] Ahmad N.B., Barakji S.A.R. et al. How to launch a successful video game: A framework. Entertainment Computing, vol. 23, 2017, pp. 1-11.
[57] Urh M., Vukovic G. et al. The Model for Introduction of Gamification into E-learning in Higher Education. Procedia - Social and Behavioral Sciences, vol. 197, 2015, pp. 388-397.
[58] Padilla-Zea N., Gutiérrez F.L. et al. Modeling storytelling to be used in educational video games. Computers in Human Behavior, vol. 31, issue 1, 2014, pp. 461-474.
[59] Fischinger D., Einramhof P. et al. Hobbit, a care robot supporting independent living at home: First prototype and lessons learned. Robotics and Autonomous Systems, vol. 75, 2016, pp. 60-78.
[60] Front A., Rieu D. et al. A participative end-user method for multi-perspective business process elicitation and improvement. Software & Systems Modeling, vol. 16, issue 3, 2017, pp. 691-714.
[61] Räisänen T., Ypsilanti A. et al. Examining the requirements for an intergenerational learning game. Education and Information Technologies, vol. 19, issue 3, 2014, pp. 531-547.
[62] Kourouthanassis P.E., Boletsis C., Lekakos G. Demystifying the design of mobile augmented reality applications. Multimedia Tools and Applications, vol. 74, issue 3, 2013, pp. 1045-1066.
[63] Shahri A., Hosseini M. et al. Engineering Digital Motivation in Businesses: A Modelling and Analysis Framework. Requirements Engineering, vol. 25, 2020, pp. 153-184.
[64] Priego-Roche L.M., Front A., Rieu D. A framework for virtual organization requirements. Requirements Engineering, vol. 21, issue 4, 2016, pp. 439-460.
[65] Hersh M., Leporini B. Editorial: Serious games, education and inclusion for disabled people. British Journal of Educational Technology, vol. 49, issue 4, 2018, pp. 587-595.
[66] Terras M.M., Boyle E.A. Integrating games as a means to develop e-learning: Insights from a psychological perspective. British Journal of Educational Technology, vol. 50. issue 3, 2019, pp. 1049-1059.
[67] Dimeff L.A., Koerner K. Fulfilling the promise of behavioral health technologies to improve public health impact and reduce public health disparities: A commentary. Clinical Psychology: Science and Practice, vol. 26, issue 1, 2019, article e12276, pp. 1-4.
[68] Peñeñory V.M., Collazos C.A. et al. APRehab: a methodology for serious games design oriented to psychomotor rehabilitation in children with hearing impairments. Universal Access in the Information Society, vol. 20, 2021, pp. 255–264.
[69] Ramos-Aguiar L.R., Alvarez-Rodriguez F.J. Teaching Emotions in Children with Autism Spectrum Disorder Through a Computer Program with Tangible Interfaces. IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, vol. 16, no. 4, 2021, pp. 365-371.
[70] Böckle M., Novak J., Bick M. Exploring gamified persuasive system design for energy saving. Journal of Enterprise Information Management, vol. 33, issue 6, 2020, pp. 1337-1356.
[71] Wang X, Goh D.H.L. et al. Understanding the determinants of human computation game acceptance: The effects of aesthetic experience and output quality. Online Information Review, Vol. 40 No. 4, 2016, pp. 481-496.
[72] Carrión-Toro M., Santorum M. et al. iPlus a user-centered methodology for serious games design. Applied Sciences, vol. 10, issue 24, pp. 1-33.
[73] Kondylakis H., Bucur A. et al. Patient empowerment for cancer patients through a novel ICT infrastructure. Journal of Biomedical Informatics, vol. 101, 2020, article no. 103342, 14 p.
[74] Cano S., Collazos C.A. et al. Towards a methodology for user experience assessment of serious games with children with cochlear implants. Telematics and Informatics, vol. 35, issue 4, 2018, pp. 993-1004.
[75] Fanfarelli J.R., McDaniel R., Crossley C. Adapting UX to the design of healthcare games and applications. Entertainment Computing, vol. 28, 2018, pp. 21-31.
[76] Fernandez-Cervantes V., Neubauer N. et al. VirtualGym: A kinect-based system for seniors exercising at home. Entertainment Computing, vol. 27, 2018, pp. 60-72.
[77] Menghi R., Papetti A., Germani M. Product Service Platform to improve care systems for elderly living at home. Health Policy and Technology, vol. 8, issue 4, 2019, pp. 393-401.
[78] Havukainen M., Laine T.H. et al. A Case Study on Co-designing Digital Games with Older Adults and Children: Game Elements, Assets, and Challenges. The Computer Games Journal, vol. 9, 2020, pp. 163–188.

[79] O'Connor S., Shuttleworth J. et al. Assessing the perceived realism of agent grouping dynamics for adaptation and simulation. Entertainment Computing, vol. 32, 2019, article no. 100323.
[80] Johnson C.M., McIlwain S. et al. Creating a sustainable collaborative consumer health application for chronic disease self-management. Journal of Biomedical Informatics, vol. 71, 2017, pp. 198-206.
[81] López S., Cervantes J.A. et al. The plausibility of using unmanned aerial vehicles as a serious game for dealing with attention deficit-hyperactivity disorder. Cognitive Systems Research, vol. 59, 2020, pp. 160-170.
[82] Quint F., Sebastian K., Gorecky D. A Mixed-reality Learning Environment. Procedia Computer Science, vol. 75, 2015, pp. 43-48.
[83] Koivisto J.M., Haavisto E. et al. Design principles for simulation games for learning clinical reasoning: A design-based research approach. Nurse Education Today, vol. 60, 2018, pp. 114-120.
[84] Gerling K.M., Linehan C. et al. Creating wheelchair-controlled video games: Challenges and opportunities when involving young people with mobility impairments and game design experts. International Journal of Human-Computer Studies, vol. 94, 2016, pp. 64-73.
[85] Cinquin P.A., Guitton P., Sauzéon H. Online e-learning and cognitive disabilities: A systematic review. Computers & Education, vol. 130, 2018, pp. 152-167.
[86] Aebli A. Tourists' motives for gamified technology use. Annals of Tourism Research, vol. 78, 2019, article no. 102753.
[87] Ingram J., Gaskell P. Searching for meaning: Co-constructing ontologies with stakeholders for smarter search engines in agriculture. Wageningen Journal of Life Sciences, issues 90-91, 2019, article no. 100300, pp. 1-13.
[88] Scott M.J., Spyridonis F., Ghinea G. Designing for designers: Towards the development of accessible ICT products and services using the VERITAS framework. Computer Standards & Interfaces, vol. 42, 2015, pp. 113-124.
[89] Tan J.L., Goh D.H.L. et al. Learning efficacy and user acceptance of a game-based social skills learning environment. International Journal of Child-Computer Interaction, vol. 9-10, 2016, pp. 1-19.
[90] Lokshina I.V., Durkin B.J. Redesigning the Healthcare Model to Address Obesity Problem Using the Integration of Processes and Mobile Technologies: Facing a Worldwide Epidemic in an Innovative Manner. Wireless Personal Communications, vol. 96, issue 4, 2017, pp. 5483-5498.
[91] Hocine N., Gouaïch A. et al. Adaptation in serious games for upper-limb rehabilitation: an approach to improve training outcomes. User Modeling and User-Adapted Interaction, vol. 25, issue 1, 2015, pp. 65-98.
[92] Vayanou M., Ioannidis Y. et al. How to Play Storytelling Games with Masterpieces: From Art Galleries to Hybrid Board Games. Journal of Computers in Education, vol. 6, 2019, pp. pages 79–116.
[93] Martinho D., Carneiro J. et al. A systematic review of gamification techniques applied to elderly care. Artificial Intelligence Review, vol. 53, 2020, pp. 4863–4901.
[94] Palumbo F., la Rosa D. et al. Reliability and human factors in Ambient Assisted Living environments: The DOREMI case study. Journal of Reliable Intelligent Environments, vol. 3, issue 3, 2017, pp. 139-157.
[95] Ivanov R. Blind-environment interaction through voice augmented objects. Journal on Multimodal User Interfaces, vol. 8, issue 4, 2014, pp. 345-365.
[96] Tuerk P.W., Schaeffer C.M. et al. Adapting Evidence-Based Treatments for Digital Technologies: a Critical Review of Functions, Tools, and the Use of Branded Solutions. Current Psychiatry Reports, vol. 21, issue 10, article no. 106, 14 p.
[97] Kosmas P., Galanakis G. et al. Enhancing accessibility in cultural heritage environments: considerations for social computing. Universal Access in the Information Society, vol. 19, issue 2, 2019, pp. 471–482.
[98] Puigdomenech E., Martin A. et al. Promoting healthy teenage behaviour across three European countries through the use of a novel smartphone technology platform, PEGASO fit for future: Study protocol of a quasi-experimental, controlled, multi-Centre trial. BMC Medical Informatics and Decision Making, vol. 19, issue 1, 2019, article no. 278, pp. 1-13.
[99] Nisiforou E.A., Zaphiris P. Let me play: unfolding the research landscape on ICT as a play-based tool for children with disabilities. Universal Access in the Information Society, vol. 19, issue 1, 2020, pp. 157-167.
[100] Stuij S.M., Labrie N.H.M. et al. Developing a digital communication training tool on information-provision in oncology: Uncovering learning needs and training preferences. BMC Medical Education, vol. 18, issue 1, 2018, article no. 220, pp. 1-12.

[101] Schließmann D., Nisser M. et al. Trainer in a pocket - Proof-of-concept of mobile, real-time, foot kinematics feedback for gait pattern normalization in individuals after stroke, incomplete spinal cord injury and elderly patients. Journal of NeuroEngineering and Rehabilitation, vol. 15, issue 1, 2018, article no. 44, pp. 1-15.

[102] Reinkensmeyer D.J., Blackstone S. et al. How a diverse research ecosystem has generated new rehabilitation technologies: Review of NIDILRR's Rehabilitation Engineering Research Centers. Journal of NeuroEngineering and Rehabilitation, vol. 14, 2017, article no. 109, pp. 1-53.

[103] Castelló V., Traver V.J. et al. Assisting therapists in assessing small animal phobias by computer analysis of video-recorded sessions. Multimedia Tools and Applications, vol. 76, issue 20, 2017, pp. 21033-21049.

[104] Santos O.C., Kravcik M., Boticario J.G. Preface to Special Issue on User Modelling to Support Personalization in Enhanced Educational Settings. International Journal of Artificial Intelligence in Education, vol. 26, issue 3, 2016, pp. 809-820.

[105] Wüller H., Behrens J. et al. A scoping review of augmented reality in nursing. BMC Nursing, vol. 18, issue 1, 2019, article no. 19, pp. 1-11.

[106] Alnusair A., Zhong C. et al. Context-aware multimodal recommendations of multimedia data in cyber situational awareness. Multimedia Tools and Applications, vol. 76, issue 21, pp. 22823-22843.

[107] Powell L., Parker J., Harpin V. What is the level of evidence for the use of currently available technologies in facilitating the self-management of difficulties associated with ADHD in children and young people? A systematic review. European Child & Adolescent Psychiatry, vol. 27, issue 11, 2018, pp. 1391-1412.

[108] Campos J.C., Abade T. et al. Don't go in there! using the APEX framework in the design of ambient assisted living systems. Journal of Ambient Intelligence and Humanized Computing, vol. 8, issue 4, 2017, pp. 551-566.

[109] Merilampi S., Koivisto A., Sirkka A. Designing serious games for special user groups—design for somebody approach. British Journal of Educational Technology, vol. 49, issue 4, pp. 646-658.

[110] Terras M.M., Boyle E.A. et al. The opportunities and challenges of serious games for people with an intellectual disability. British Journal of Educational Technology, vol. 49, issue 4, pp. 690-700.

[111] Bossavit B., Parsons S. Outcomes for design and learning when teenagers with autism codesign a serious game: A pilot study. Journal of Computer Assisted Learning, vol. 34, issue 3, 2018, pp. 293-305.

[112] Cano A.R., Fernández-Manjón B., García-Tejedor Á.J. Using game learning analytics for validating the design of a learning game for adults with intellectual disabilities. British Journal of Educational Technology, vol. 49, issue 4, 2018, pp. 659-672.

[113] Hodge P., Davis J. et al. StreetWise: A valid ecology for a serious game in a secure forensic mental health setting. Procedia Computer Science, vol. 63, 2015, pp, 252-259.

[114] Ganzeboom M., Bakker M. et al. Speech training for neurological patients using a serious game. British Journal of Educational Technology, vol. 49, issue 4, pp. 761-774.

[115] Perry D., Robinson J. et al. Game design for bioinformatics and cyberinfrastructure learning: a parallel computing case study. Concurrency and Computation: Practice and Experience, vol. 22, issue 6, 2014, pp. 685-701.

[116] Leroi I., Watanabe K. et al. "Psychogeritechnology" in Japan: Exemplars from a super-aged society. International Journal of Geriatric Psychiatry, vol. 33, issue 12, pp. 1533-1540.

[117] Park J., Mostafa N.A., Han H.J. "StoryWeb": A storytelling-based knowledge-sharing application among multiple stakeholders. Creativity and Innovation Management, vol. 29, issue 2, 2020, pp. 224-236.

[118] Sharit J., Lisigurski M. et al. The roles of health literacy, numeracy, and graph literacy on the usability of the VA's personal health record by veterans. Journal of Usability Studies, vol. 9, issue 4, pp. 173-193.

[119] Nunes F., Verdezoto N. et al. Self-care technologies in HCI: Trends, tensions, and opportunities. ACM Trans Comput Interact, vol. 22, issue 6, 2015, article no. 38, pp. 1-40.

[120] Spiel K, Frauenberger C, Keyes OS, Fitzpatrick G. Agency of autistic children in technology research - A critical literature review. ACM Transactions on Computer-Human Interaction, vol. 26, issue 6, 2019, article no. 33, pp. 1-45.

[121] Reynolds L.M., Davies J.P. et al. StreetWise: developing a serious game to support forensic mental health service users' preparation for discharge: a feasibility study. Journal of Psychiatric and Mental Health Nursing, vol. 24, issue 4, 2017, pp. 185-193.

[122] Savazzi F., Isernia S. et al. Engaged in learning neurorehabilitation: Development and validation of a serious game with user-centered design. Computers & Education, vol. 125, 2018, pp. 53-61.
[123] Robertson J., Macvean A. et al. Savouring our mistakes: Learning from the FitQuest project. International Journal of Child-Computer Interaction, vol. 16, 2018, pp. 55-67.
[124] Adams A., Hart Jet al. Co-created evaluation: Identifying how games support police learning. International Journal of Human-Computer Studies, vol. 132, 2019, pp. 34-44.
[125] Rodrigues L.F., Costa C.J., Oliveira A. Gamification: A framework for designing software in e-banking. Computers in Human Behavior, vol. 62, 2016, pp. 620-634.
[126] Françoise J., Bevilacqua F. Motion-sound mapping through interaction: An approach to user-centered design of auditory feedback using machine learning. ACM Transactions on Interactive Intelligent Systems, vol. 8, issue 2, 2018, article no. 16, pp. 1-30.
[127] Kayali F., Silbernagl M. et al. Design considerations for a serious game for children after hematopoietic stem cell transplantation. Entertainment Computing, vol. 15, 2016, pp. 57-73.
[128] Salomão R.C.S., Rebelo F., Rodríguez F.G. Defining Personas of University Students for the Development of a Digital Educational Game to Learn Portuguese as a Foreing Language. Procedia Manufacturing, vol. 3, 2015, pp. 6214-6222.
[129] Ramos-Vega M.C., Palma-Morales V.M. et al. Stimulating children's engagement with an educational serious videogame using Lean UX co-design. Entertainment Computing, vol. 38, 2020, pp. 6214-6222.
[130] van der Lubbe L.M., Gerritsen C. et al. Empowering vulnerable target groups with serious games and gamification. Entertainment Computing, vol. 38, 2021, article no. 100402, pp. 1-27.
[131] Bennani S., Maalel A., Ben Ghezala H. Age-learn: Ontology-based representation of personalized gamification in e-learning. Procedia Computer Science, vol. 176, 2020, pp. 1005-1014.
[132] Stamm O., Dahms R., Müller-Werdan U. Virtual Reality in Pain Therapy: A Requirements Analysis for Older Adults with Chronic Back Pain. Journal of NeuroEngineering and Rehabilitation, vol. 17, 2020, article no. 129, pp. 1-12.
[133] Spil T.A.M., Romijnders V. et al. Are serious games too serious? Diffusion of wearable technologies and the creation of a diffusion of serious games model. International Journal of Information Management, vol. 58, 2021, article no. 102202, pp. 1-9.
[134] Teruel M.A., Navarro E. et al. Applying thematic analysis to define an awareness interpretation for collaborative computer games. Information and Software Technology, vol. 74, 2016, pp. 17-44.
[135] Bruno F., Barbieri L. et al. Virtual dives into the underwater archaeological treasures of South Italy. Virtual Reality, vol. 22, issue 2, 2018, pp. 91-102.
[136] Koutsabasis P., Vosinakis S. Kinesthetic interactions in museums: conveying cultural heritage by making use of ancient tools and (re-) constructing artworks. Virtual Reality, vol. 22, issue 2, 2018, pp. 103-118.
[137] Speake H., Copeland R.J. et al. Embedding Physical Activity in the Heart of the NHS: The Need for a Whole-System Approach. Sports Medicine, vol. 46, issue 7, 2016, pp. 939-946.
[138] Sigala M. The application and impact of gamification funware on trip planning and experiences: the case of TripAdvisor's funware. Electronic Markets, vol. 25, issue 3, 2015, pp. 189-209.
[139] Bonet N., von Barnekow A. et al. Three-Dimensional Game-Based Cardiopulmonary Bypass Training. Clinical Simulation in Nursing, vol. 50, 2021, pp. 81-91.
[140] Zhang-Kennedy L., Chiasson S. A Systematic Review of Multimedia Tools for Cybersecurity Awareness and Education. ACM Computing Surveys, vol. 54, issue 1, 2021, article no. 12, pp. 1-39.
[141] Schulz R., Smaradottir B. et al. User-Centered Design of a Scenario-Based Serious Game: Game-Based Teaching of Future Healthcare. IEEE Transactions on Games, vol. 12, issue 4, 2020, pp. 376-385.
[142] Agbo F.J., Oyelere S.S. et al. Co-design of mini games for learning computational. Education and Information Technologies, vol. 26, issue 5, 2021, pp.5815--5849.
[143] Woolford K., Dunn S. Experimental archaeology and games: Challenges of inhabiting virtual heritage. Journal on Computing and Cultural Heritage, vol. 6, issue 4, 2013, article no. 16, pp. 1-15.
[144] Gilbert S.B., Jang W. et al. Re-solution-Katrina edition: Moving a face-to-face game online. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 51, issue 1, 2017, pp. 356-360.
[145] Wasil A.R., Venturo-Conerly K.E. et al. A review of popular smartphone apps for depression and anxiety: Assessing the inclusion of evidence-based content. Behaviour Research and Therapy, vol. 123, 2019, article no. 103498, pp. 1-9.

[146] Li Q. Enactivism and teacher instructional game building: an inquiry of theory adoption and design consideration. Educational Technology Research and Development, vol. 66, issue 6, 2018, pp. 1339-1358.
[147] Lorenz T., Weiss A., Hirche S. Synchrony and Reciprocity: Key Mechanisms for Social Companion Robots in Therapy and Care. International Journal of Social Robotics, vol. 8, issue 1, 2016, pp. 125-143.
[148] Money A., Coughlan J. Team-taught versus individually taught undergraduate education: a qualitative study of student experiences and preferences. Higher Education, vol. 72, issue 6, 2016, pp. 797-811.
[149] Corrêa Souza A.C., Nunes F.L.S., Delamaro M.E. An automated functional testing approach for virtual reality applications. Software Testing, Verification and Reliability, vol. 28, issue 8, 2018, article no. e1690, pp. 1-31.
[150] Pyae A., Liukkonen T. et al. When Japanese elderly people play a Finnish physical exercise game: a usability study. Journal of Usability Studies, vol. 11, issue 4, 2016, pp. 131-152.
[151] Konstantakis M., Caridakis G. Adding culture to UX: UX research methodologies and applications in cultural heritage. Journal on Computing and Cultural Heritage, vol. 13, issue 1, 2020, article no. 4, pp. 1-17.
[152] Tao G., Garrett B. et al. Immersive virtual reality health games: a narrative review of game design. Journal of NeuroEngineering and Rehabilitation, vol. 18, issue 1, 2021, article no. 31, pp. 1-21.
[153] Pedraza-Hueso M., Martín-Calzón S. et al. Rehabilitation Using Kinect-based Games and Virtual Reality. Procedia Computer Science, vol. 75, 2015, pp. 161-168.
[154] Fonseca D., García-Peñalvo F.J. Interactive and collaborative technological ecosystems for improving academic motivation and engagement. Universal Access in the Information Society, vol. 18, issue 3, 2019, pp. 423-430.
[155] Tong T., Chignell M., Sieminowski T. Case Study: A Serious Game for Neurorehabilitation Assessment. Procedia Computer Science, vol. 69, 2015, pp. 125-131.
[156] Valladares-Rodriguez S., Fernández-Iglesias M.J. et al. Touchscreen games to detect cognitive impairment in senior adults. A user-interaction pilot study. International Journal of Medical Informatics, vol. 127, 2019, pp. 52-62.
[157] Luz S., Masoodian M. et al. Using a serious game to promote community-based awareness and prevention of neglected tropical diseases. Entertainment Computing, vol. 15, 2016, pp. 43-55.
[158] Hidalgo-Mazzei D., Reinares M. et al. OpenSIMPLe: A real-world implementation feasibility study of a smartphone-based psychoeducation programme for bipolar disorder. Journal of Affective Disorders, vol. 241, 2018, pp. 436-445.
[159] Martínez-González C.L, Camargo-Fajardo M.C.C. et al. Therapeutic Patient Education with Learning Objects Improves Asthma Control in Mexican Children. Journal of Medical Systems, vol. 44, issue 4, 2020, article no. 79, pp. 1-9.
[160] Ghanbari H., Similä J., Markkula J. Utilizing online serious games to facilitate distributed requirements elicitation. Journal of Systems and Software, vol. 109, 2015, pp. 32-49.
[161] de Troyer O., Janssens E. Supporting the requirement analysis phase for the development of serious games for children. International Journal of Child-Computer Interaction, vol. 2, issue 2, 2014, pp. 76-84.
[162] Sobrino-Duque R., Martínez-Rojo N. et al. Evaluating a gamification proposal for learning usability heuristics: Heureka. International Journal of Human-Computer Studies, vol. 161, 2022, article no. 102774, pp. 1-15.
[163] Howes S.C., Charles D. et al. User-centred design of an active computer gaming system for strength and balance exercises for older adults. Journal of Enabling Technologies, vol. 13, issue 2, 2019, pp. 101-111.
[164] Harrington M.C.R. The Virtual Trillium Trail and the empirical effects of Freedom and Fidelity on discovery-based learning. Virtual Reality, vol. 16, issue 2, 2012, pp. 105-120.
[165] Bontchev B., Antonova A. et al. "Let Us Save Venice"—An Educational Online Maze Game for Climate Resilience. Sustainability, vol. 14, issue 1, 2022, article no. 7, pp. 1-23.
[166] Ali Z., Usman M. A framework for game engine selection for gamification and serious games. In Proc. of the 2016 Future Technologies Conference (FTC), 2016, pp. 1199-1207.
[167] Zhu M., Wang A.I. Model-driven game development: A literature review. ACM Computing Surveys, vol. 52, issue 6, 2020, article no. 123, pp 1–32.
[168] González Sánchez J.L., Padilla Zea N. et al. De la Usabilidad a la Jugabilidad: Diseño de Videojuegos Centrado en el Jugador. In Proc. of the IX Congreso Internacional Interacción, 2008, pp. 99-108 (in Spanish).

[169] Kuznetsov M. B. UML model transformation and its application to MDA technology. Programming and Computer Software., vol. 33, issue 1, 2007, pp. 44-53, 2007 / Кузнецов М.Б. Трансформация UML-моделей и ее использование в технологии MDA. Программирование, том 33, вып. 1, 2007 г., стр. 65-79.

[170] Gorshkova E.A., Novikov B.A. et al. A UML-based modeling of web application controller. Programming and Computer Software., vol. 31, issue 1, 2005, pp. 29–33 / Горшкова Е.А., Новиков Б.А. Моделирование контроллера web-приложений с использованием UML. Программирование, том 31, вып. 1, 2005 г., стр. 44-51.

Information about authors / Информация об авторах

Pedro Omar SILVA-VÁSQUEZ – Currently a PhD student in computer science since 2019. in the Universidad Veracruzana, with a master's degree in User-Centered Interactive System in 2016 and bachelor's in administrative computer systems graduated from Universidad Veracruzana in 2013. His areas of interested are software development research, user-centered design, user experience, human-computer interaction and video game development.

Педро Омар СИЛЬВА-ВАСКЕС – в настоящее время аспирант компьютерных наук с 2019 г. в университете Веракрусаны, где получил и степени бакалавра и магистра. Области его интересов – исследования в области разработки программного обеспечения, дизайн, ориентированный на пользователя, пользовательский опыт, взаимодействие человека с компьютером и разработка видеоигр.

Viviana Yarel ROSALES-MORALES – received the BS degree in Computer Systems and MSc degree in Computer Systems in 2009 and 2011, respectively. And in June 2017, she got a PhD in Engineering Sciences from the Technological Institute of Orizaba, Veracruz, Mexico. She has involved in some Mexican research projects and joined the Faculty of Statistics and Informatics of the Universidad Veracruzana through the Cátedras CONACyT program in 2019. Her research interests include: Human-Computer Interaction, User Experience, Serious Games and eHealth Applications, to name a few.

Вивиана Ярель РОЗАЛЕС-МОРАЛЕС в июне 2017 года получила степень доктора технических наук в Технологическом институте Орисаба, Веракрус, Мексика. Она участвовала в некоторых мексиканских исследовательских проектах и поступила на факультет статистики и информатики университета Веракрусана в рамках программы Cátedras CONACYT в 2019 году. Ее исследовательские интересы среди прочего включают взаимодействие человека с компьютером, пользовательский опыт, серьезные игры и приложения для электронного здравоохранения.

Edgard BENÍTEZ-GUERRERO – Ph. D. in Computer Science from the University of Grenoble in France. Professor at the Faculty of Statistics and Informatics of the University of Veracruz in Mexico. Research interests: Human Computer Interaction, Artificial Intelligence, Collaborative Computing, Data Management and Visualization.

Эдгар БЕНИТЕС-ГЕРРЕРО – доктор компьютерных наук Гренобльского университета во Франции. Профессор факультета статистики и информатики Университета Веракруса в Мексике. Научные интересы: взаимодействие человека с компьютером, искусственный интеллект, совместные вычисления, управление данными и визуализация.

DOI: 10.15514/ISPRAS-2022-34(3)-10

Usability Evaluation of Brain Computer Interfaces: Analysis of State of Art

Y.N. Ortega-Gijón, ORCID: 0000-0002-5396-0676 <zS18016065@estudiantes.uv.mx>
C. Mezura-Godoy, ORCID: 0000-0002-5386-107X <cmezura@uv.mx>
Facultad de Estadística e Informática, Universidad Veracruzana,
Xalapa-Enríquez, Veracruz, México 91020

Abstract. Brain Computer Interfaces – BCI allow users to communicate with the software system through cognitive functions measurable by brain signals, identified as Electroencephalography – EEG. User tests have been the most used method for usability evaluation of BCI software applications. In user tests, the data collected comes from the opinions of users through questionnaires, these tests require a lot of time, since they include not only performing interaction task and the application of the questionnaires, but also include placing and calibrating the EEG device. All this makes the evaluation process a very heavy task for the participants of the test and can mean that the data collected is not entirely reliable. That is why we are interested in including EEG signals in the usability evaluation process of applications with BCI software applications. Therefore, we present in this paper the result of the analysis of state of art in order to identify the relevant works in the area and future lines of research.

Keywords: usability evaluation; brain-computer interfaces; EEG signals

For citation: Ortega Y.N., Mezura-Godoy C. Usability Evaluation of Brain Computer Interfaces: Analysis of State of Art. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 145-158. DOI: 10.15514/ISPRAS–2022–34(3)–10

Оценка пригодности к использованию нейрокомпьютерных интерфейсов: анализ состояния дел

Й.Н. Ортега-Хихон, ORCID: 0000-0002-5396-0676 <zS18016065@estudiantes.uv.mx>
К. Мезура-Годой, ORCID: 0000-0002-5386-107X <cmezura@uv.mx>
Факультет статистики и информатики, Университет Веракрусана,
91020 Мексика, Веракрус, Халапа-Энрикес

Abstract. Нейрокомпьютерные интерфейсы (Brain Computer Interfaces, BCI) позволяют пользователям общаться с программной системой посредством когнитивных функций, измеряемых сигналами мозга, которые опознаются с помощью. электроэнцефалографии — ЭЭГ. Наиболее часто используемым методом оценки удобства использования программных приложений BCI являются пользовательские тесты. В пользовательских тестах данные собираются на основе мнений пользователей, получаемых путем анкетирования. Такая оценка требуют много времени, поскольку требуются не только выполнение задания на взаимодействие и заполнение анкет, но также и размещение и калибровку устройства ЭЭГ. Все это делает процесс оценки очень тяжелой задачей для участников теста и может означать, что собранные данные не совсем надежны. Вот почему нас интересует включении сигналов ЭЭГ в процесс оценки удобства пригодности к использованию приложений BCI. Поэтому мы представляем в этой статье результат анализа состояния дел, чтобы определить значимые работы в этой области и будущие направления исследований.

Ключевые слова: пригодность к использованию; нейрокомпьютерные интерфейсы; сигналы ЭЭГ

Для цитирования: Ортега-Хихон Й.Н., Мезура-Годой К. Оценка пригодности к использованию нейрокомпьютерных интерфейсов: анализ состояния дел. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 145-158.. DOI: 10.15514/ISPRAS–2022–34(3)–10

1. Introduction

Usability is defined as: «The range in which a product can be used by specific users to achieve certain specified goals with effectiveness, efficiency and satisfaction in a specified context of use» (ISO 9241-11). The usability evaluation can be carried out following different paradigms: Quik and dirty, usability tests, field and predictive studies, which through different techniques such as: user tests, thinking aloud, interviews, questionnaires, heuristics, etc. collect data for analysis. The data collected can be quantitative and qualitative, which can also be recovered from physiological measures such as: cardiac rhythm, blood pressure, temperature, etc.

Our interest is the evaluation of the usability of BCI software applications. In these applications, the interaction between users and the computer system takes place through electrical activity of the human brain and the device to be controlled Gentiletti [17]. In BCI software applications, usability is traditionally evaluated by user testing. User tests can be Qualitative usability testing and Quantitative usability testing.

These tests comprise 3 phases: 1) Interaction with the User, 2) Application of the questionnaires (post-task or post-test, and 3) Collection and analysis of data. The interaction is the time allocated to the task, the application of questionnaires, the opinion about the application is obtained, then the data of the questionnaires are collected and analyzed by statistical means (mean, average, mode). Through the questionnaires, subjective measures are retrieved that depend on the opinion or state of mind of the user, which may affect the results of the evaluation. On the other hand, the application of the questionnaire can be given only at specific times, it is usually done after completing the test, however, the questionnaires can also be applied after completing specific tasks, which can increase the time of the test. evaluation, this can cause fatigue in the participant when performing the usability test, for example at work [35], users express that the questionnaires are confusing, long, tedious and presented a high degree of fatigue and workload throughout the evaluation process.

Particularly for the evaluation of BCI software applications through usability testing, the calibration phase is added to the process at the beginning of the evaluation process. We call it Phase «0» calibration refers to ensuring that all channels respond equally, looking at the quality of the signal. This phase can make the test longer and can contribute to participant fatigue.

On the other hand, it is important to note that EEG signals have been used in «Neuromarketing»[52] and «Clinical studies» [16] where are they used to assess cognitive states of patients. Therefore, the EEG signals have been linked to workload, concentration, emotions, etc. [2,3,14].

The paper by Rhiu [44] shows a review of the BCI evaluation works, classifying by dimension of usability and showing the measures, but obtained only through questionnaires and describing the most used. The objective of this paper is to present the results of the analysis of the literature on the usability evaluation of applications with ICC, that is, to know what are the methods and techniques used in the evaluation, highlighting the analysis of the use of EEG signals in the obtaining usability measures, the data that ensured the EEG signals and exploring the possibility of being used in the evaluation of usability, classifying the EEG signals by measure and dimension.

The paper is organized as follows: In section 2 it presents the description of the BCI software applications, in section 3 the techniques, dimensions, measures, and the usability evaluation process are shown, in section 4 the process that was carried out for the search, from the selection of the database to the analysis of the results, the classification and the analysis carried out of the information (papers) obtained, and in section 5 the discussion of the results (evidence) and finally the conclusions.

2. Brain-Computer Interface – BCI

The term interface is used to name the functional connection that exists between two software systems, devices, which provides communication at various levels, making an exchange of information possible. In Brain-Computer Interfaces-BCI, this exchange of information takes place between the electrical activity of the brain and the device to be controlled. BCIs provide their users with communication and control channels that are not dependent on normal output channels [17]. BCI software applications can be developed using a variety of different types of neurological signals, such as functional near-infrared spectroscopy (fNIRS), magnetic encephalography (MEG), or functional magnetic resonance imaging (fMRI). However, one of the most widely used methods to measure neurological activity used in BCI is the electroencephalogram (EEG).

The architecture of a BCI can be divided into 3 important components. The first component is the human through cognitive functions that are executed when the human being receives, interprets, and stores information, at the moment of sensory perception, then an action is executed according to the previous perception. The second component is the interaction, when the action performed by the user is sent to an input interface, the recognition of the action is performed, the representation and ending with the sending to the output interface, to start the cycle again with the perception. The third component is the recognition of the actions by the computer and the subsequent representation, the computer is in charge of interpreting the cognitive functions of the user and executing the actions thus giving feedback to the user to continue with the cycle.

There are different types of devices to measure brain activity, from the complete medical EEG with 32 channels, to headbands or caps that contain 32 to 2 channels. One of the most widely used headbands is the Emotiv Epoc [47,49]. Other hardware used are NIRSport 2 [28], the IMEC EEG [39], or they decide to create their own device [39].

Neural activity during user interaction is recovered with EEG signals. In the signal analysis, three stages are defined: 1) the acquisition of the signal, 2) the processing of the signal, and finally, 3) the interaction with the control interface and the device driver. The signal processing stage can be divided into 2 actions: characteristic extraction and classification. In this phase, specialized techniques and algorithms are used. For the extraction of characteristics, the most used algorithms are: ICA [49], LDA [51], PCA, etc., dedicated to obtaining the characteristics, important or predominant patterns in the EEG signal. Subsequently, for the classification of characteristics are: the linear discriminant analysis that uses Bayes' theorem, the vector support machine [10], the artificial neural networks (multilayer perceptron) [53], the model classifiers of hidden Markov, the fast Fourier transform (FFT) [7,16], among others.

3. Usability in Bci Software Applications

Usability is part of the broader term «User Experience-Ux» and refers to the ease of access and/or use of a product or website [33]. A design is not usable or unusable; it depends on its characteristics, the user context (what the user wants to do with it and the user's environment), all this determines its level of usability.

There are 4 paradigms for usability evaluation: Quick and dirty, usability testing, field studies and predictive or heuristics. The first 3 paradigms require user participation and in the last paradigm, the evaluation is done by usability experts, using heuristics or interaction models [18,57]. These techniques require the participation of a representative sample of end users. These evaluations are usually carried out during the later stages of development. Representative techniques are: 1) ***Thought aloud protocol***: When users are tested, while they are in the interaction phase, users are asked to verbally express what they are thinking and what they do not understand, to express their opinions about the system, product, software, etc. [34,38]. 2) ***Eye-tracking***: Allows documenting the system points that the user has been always looking [29]. 3) ***Card Sorting:*** This technique helps to discover or validate how users understand the relationship between different elements. It consists of giving the participants a series of «cards» to organize items under predetermined categories [46].

4) ***Test A/B***: It consists of comparing two versions of the same system, interface, or application to check which of the two versions is more efficient. These variations, called A and B, are randomly shown to different users [13]. 5) ***Questionnaires***: These instruments allow the software evaluator to retrieve data during the task or after the test. Some of these questionnaires are described below. NASA Task Load Index (NASA TLX) [25] is a technique for assessing mental workload. Derive a general workload based on six subscales: mental demand, physical demand, time demand, performance, effort, and frustration. Visual Analogue Scale (VAS) [11] is a questionnaire to evaluate a "feeling", generally it is carried out to evaluate the satisfaction of a system in the BCI usability studies. Questionnaire for User Interface Satisfaction (QUIS) [9] is a Satisfaction Questionnaire that elicits user feedback and assesses user acceptance of a computer interface. System Usability Scale (SUS) [25] and Utility, Satisfaction and Ease of Use (USE) questionnaire are simple but effective tools to evaluate the usability of various products. Also, the IBM IT [26,44] usability satisfaction questionnaires also measure user satisfaction with usability in a computer system.

In another hand, there are three principles or dimensions have been defined for the measurement of usability, they are efficiency, effectiveness, and satisfaction [ISO/IEC TR 9126-4]. Effectiveness refers to the precision and completeness with which certain users achieved specific objectives in a particular environment. Efficiency corresponds to the fact that the system must be efficient to use so that once the user has learned the system, and satisfaction refers to how pleasant it is to use the product. It has also been considered that usability can also be measurable in terms of: «Ease of use», «Learning ability», «Consistency», «Frustration», «Task speed», «Accuracy», etc. [19,22,37,44].

To obtain the dimension of effectiveness, it is obtained through objective measures, which correspond to «How effective and efficient is a system / product», the questions raised are perfectly delimited, the results are quantitative and admit a single solution, taking as an example of measurements the accuracy of the classification, the error rate, the task completion rate, etc. In the case of efficiency, it is achieved through objective and subjective measures, the latter refer to the personal opinions of the user, as an example of the measures on the part of efficiency is mental demand, frustration, effort, and on the other hand satisfaction, measures of ease of use, learn-ability, usefulness, reliability, consistency, etc.

4. Method of Search Process

There are different types of research like quick review, scope review, etc. However, it was decided to conduct a systematic review. Systematic review is a research method and process for identifying and evaluating relevant research, as well as collecting and analyzing data from such research. The goal of a systematic review is to identify all the empirical evidence that meets the inclusion criteria to answer a given research question [45]. In the process of searching and selecting the papers, the Kitchman proposal for systematic reviews was followed [58]. Kitchman's method includes the following phases: 1) Selected database, 2) keywords for the search, 3) Inclusion and exclusion criteria, 4) Selected papers and finally 5) Quality assessment, then the analysis task was carried out (See Fig.1).

For the search and analysis of the related works, the following research questions were defined: 1) How is the evaluation of the BCI software applications carried out?, Which would allow to know the methods and techniques used for the evaluation in the BCI, to understand the phases, stages and instruments used, 2)What have the EEG signals been used for? and what information can be obtained from the signals?, with the objective of know what data or information the EEG signals can provide, in what area and for which the EEG signals have been used, 3) Do the EEG signals provide enough elements (data) to assess usability?; with the purpose of analyze whether the information provided by the signals is sufficient and influence the usability evaluation. The research method for the search for related papers began with the selection of the most important and well-known databases, subsequently, the keywords related to the subject of usability evaluation in the BCI software

applications were defined. Once the search was carried out, those works that were not directly related to the topic of interest were excluded, and the resulting papers were classified by the evaluation.

Fig. 1. Process for search and selection of related papers

In the **Selected Database** for the search, the information sources IEEE, Springer, Elsevier, ACM, Taylor and Francis, among others, were defined. These bases are the best known, important, and complete, mainly considering the research area of this work.

For the **Keywords in the search** the papers were searched in the electronic databases, with the following search string: («EEG» and «HCI») OR ((«Brain Computer Interface» OR «BCI») AND («Usability» OR «User Experience» OR» UX «)), using the boolean operators «AND» and «OR» that are used in the formation of the search string in the database, this string was used in the search of each of the databases selected. Those works that were not indexed and published, also those prior to 2005, were discarded.

In order to focus only on papers relevant to the research, it was necessary to define **inclusion and exclusion criteria**. Which are described below: ***Inclusion criteria*:** 1) The paper is related to some dimension or measure of usability or UX and EEG. 2) The paper presents an experimental study on usability or the use of EEGs or on obtaining usability measures. 3)Papers published since 2005-Present. ***Exclusion criteria***: 1) The paper includes the BCI software applications but has no relation to usability. 2) The paper is related to EEG but does not perform the analysis or evaluation of any measure of usability. 3) If you do not present significant evidence or a conclusion.

From the search in the different databases, 139 papers were found. Subsequently, those papers that were found to be duplicates were excluded. In case of doubt, the full text versions of the citations were consulted. Resulting in 96 full-text papers evaluated for eligibility.

Continuing with the filtering by each inclusion and exclusion criteria, 18 papers that were not related to usability or UX were excluded, 21 papers that were related to the topic of electroencephalography but that did not carry out the analysis or evaluation of any usability measure, and finally 18 papers that do not present significant evidence for the study. Culminating with 41 primary papers.

The **Selected papers in** Table 1 shows the total number of selected jobs. Column 1 shows the database consulted, column 2 the total number of papers excluded per database, column 3 the number of primary papers and finally column 4 the percentage of primary papers per database. A total of 41 papers were identified. The databases with the highest results were IEEE, Springer and ACM. Numerous context-aware papers have grown considerably since 2010. The number of papers in 2018 has become 7 times the number of papers in 2007.

Table 1. Classification of papers based on the database

Editor	**#Disc.**	**#Exc**	**#Prim.**	**%**
Elsevier	9	6	3	7.32%
IEEE	26	9	17	41.5%
Inderscience Enterprises	6	5	1	2.44%
Springer	17	10	7	17%
Public Library of Science	4	3	1	2.44%
M D P I	5	3	2	4.88%
ACM	8	3	5	12.2%
Others	64	59	5	12.2%
Total Items	139	98	41	100%

In the **Quality assessment**, each SLR was evaluated using the York University, Centre for Reviews and Dissemination (CDR) A quality assessment tool for diagnostic accuracy studies (QUADAS)[3], using the following criteria: 1) Are the user's representative of the users who will receive the test in practice?, 2) Is the reference standard likely to correctly classify the target condition?, 3) Was the execution of the experimental study described in sufficient detail to allow replication?, 4) Were un-interpretable/ intermediate test results reported?, and 5) Are the data with which the test results were interpreted available?. these criteria consist of 4 key domains that cover the selection of patients, the reference standard, the execution of the test (description and replication), and the interpretation of results.

5. Analysis of Results

In the works found, it was identified that the usability of the ICC was obtained through usability tests, applying techniques such as questionnaires or EEG signals for data collection.

In the **Data collection by Questionnaires**, some questionnaires that were used in BCI usability studies are as follows: NASA-TLX [25], VAS [11], Assistive Technology Device (ATD-PA) Readiness Assessment Device form, SUS survey, QUEST 2.0 Questionnaire, IBM Computing Usability, USE Questionnaire, and QCM Questionnaire. These questionnaires were described in section 3.1. The works of García Ramírez *et al* [16], and Pasqualotto. *et al.* [38], used the SUS questionnaire to obtain ease of use. On the other hand, the works of Chowdhury [10] and Laar *et al.* [24] used the VAS questionnaire to measure the mood of users. And finally, Pasqualotto *et al*. [38] and Laar *et al*. [24] measured workload using the NASA tlx questionnaire. Finally, some studies proposed and carried out their own evaluation tools [44], to obtain measures such as comfort and mood, in a precise way, which may not be possible when using existing questionnaires. Table 2 shows works that have used questionnaires to collect data that measure different elements of efficiency, effectiveness, and satisfaction.

Table 2. Subjective measures obtained by questionnaires in BCI

Dimension	Measure	Reference
Efficiency	Workload	[24,38]
	Comfort	[55]
	Frustration	[24]
	Fatigue	[10]
Satisfaction	Mood	[10]
	Learning ability	[38]
	Easy to use	[16, 38]
	Motivation	[10]
	Presence	[24]
	Fun	[24]

On the other hand, **data collection by EEG** is presented, with respect to EEG signals and frequency bands Delta (0.1 Hz to 3.9 Hz), Theta (4.0 Hz to 7.9 Hz), Alpha (8.0 Hz to 12.9 Hz), Beta (13 Hz to 29.9 Hz) and Gamma (from 30 Hz to 100 Hz) in BCI that have been used to obtain measures such as «Concentration», «Emotions» and «Fatigue». In several works they measure the factors by obtaining the frequency bands of the signals, in most they perform the combination of the different bands. For the level of ***concentration***, in the work of Wang *et al.* [54], they get the concentration level in the entertainment area, reporting the experiments they carried out and identifying that the «O1» channel and the combination of all bands help in the detection. In relation to «***emotions***», the following works were found: In Garcia's work [16] performs emotion detection through EEG signals of a BCI software application for people with cerebral palsy is described, the main contribution of this work is the method used to obtain the emotions of the users during their interaction with BCI,

and it's identified the channels F3 and F4 with the band Alpha are the data that provides or influences the most for the detection of emotions. Another work is of the Sourina *et al.* [36] where emotion detection is performed and describes some examples where it can be used such as «Emotional Avatar», «website», «music reproductions», etc., using 14 channels and the combination of all bands. And finally, regarding "***fatigue***", the work of Arai *et al.* [4], performs the load measurement with the Alpha band.

Regarding the *use of signals*, the brain processes that reflect cognitive and attention states during human-machine interaction are studied extensively with EEG. Therefore, the signals have been occupied in obtaining measures such as: "Workload", "Comfort", "Attention", "Stress", "Mistakes" and "Emotions" mainly in the areas of Neuromarketing and clinical studies. In the field of neuromarketing, economists use EEG research to detect brain processes that drive consumer decisions, brain areas that are active, and mental states [50]. In clinical and psychiatric studies, EEG is used to assess the cognitive states of patients, determine sites of lesions, and symptoms [16].

In the workload, some examples of these works are: 1) The work of Kumar *et al.* [23] carried out the measurement of the workload of cognitive tasks, obtaining with this study which are the channels that most influence to make a correct classification, being AF3, AF4, T7, and T8. 2) The works of Appriou *et al.* [3] occupying 28 active electrodes in the 10/20 system. And 3) Frey *et al.* [14] use 32 channels, and Antonenko obtains the workload with the channels «F7 and P3». For "comfort", in the work by Frey *et al.* [15] they use the 32 active channels. Regarding "attention", in the work of Putze *et al* [40], they detected that the channels «P8, CP6, and O2» are the ones that contribute the most or influence. In the "stress", the work of Hosseini *et al.* [19], occupying the channels FP1, FP2, T3, T4, and Pz provides enough information for stress measurement. And finally, regarding "emotions", the work of Ansari *et al*. [1] given that there are already several studies for the detection of emotions, its main attribution is the selection of the channels that most influence their classification/EEG detection, making the processing of count is lower, selecting the channels F3, F4, CP5, CP6 those that influence the most. The works that carry out the obtaining of measurements by means of EEG are presented in table 3, classifying the works by signal, in this the usability module that is being evaluated is presented, what measure are they obtaining, what are the signals that they are occupying.

Table 3. Measures obtained by EEG signals

Measures	**Signals**	**Reference**
Workload	AF3, AF4, T7 and T8	[23]
	28 channels	[3]
	32 channels	[14]
	F7, P3	[2]
Comfort	32 C- Pz	[15]
Attention	O1	[54]
	P8, CP6 y O2	[40]
	32 channels	[14]
Stress	FP1, FP2, T3, T4, Pz	[19]
Mistakes	64 channels	[37]
Emotions	14 channels	[8, 22, 36, 41]
	F3, F4	[16]
	AF3, F4 and FC6	[27]
	32 channels	[20, 56]
	AFz, F3, F4, CP5, CP6	[1]
	63 channels	[32]
	AF3, AF4, F3, F4	[42]

	AF3, AF4, F3, F4, T7, T8	[21]
	FP1, F3, P3, O1	[6]
	P3, P6, P7 and PO8	[30]

On the other hand, the frequency bands have been used to obtain the measurements: «workload, fatigue, comfort, attention, stress, and emotions». To obtain the workload, Antonenko *et al*. [2] occupied the Theta and Alpha bands, on the other hand, Appriou *et al*. [3] only occupied the alpha band, and Frey *et al*. [14] when performing a combination of the 5 bands. Regarding fatigue, Arai *et al*. [4] and Mardiyanto [5] determined that the alpha band is decisive. About comfort, for obtaining Frey *et al*. [15] occupies the Theta, Alpha and Beta bands. In attention, the Alpha [40], Delta [23] have been dealt with, apart from the works of Frey [14] and Wang [54]. And finally for emotions, theta, alpha, beta, and gamma bands have been used, there are also works such as the one by Kortelainen et al.[20] and Liu et al. [27] that use all 5 frequency bands. The table 4 shows the classification of the works by frequency bands. This table presents the usability module that you are evaluating, what measures are you obtaining, what are the occupied bands and the job (s).

Table 4. Measures obtained by bands

Measures	**Bands**	**Reference**
Workload	Theta	[2, 23]
	Alpha	[2, 3, 23]
	Beta	[22, 23]
	Gamma	[22]
Comfort	All	[14]
Fatigue Comfort	Alpha	[4, 5]
	Theta	[15]
	Alpha	[15]
Stress	Beta	[15]
Attention	All	[14, 54]
Emotions Stress Emotions **Measures** Workload	Alpha	[40]
	Delta	[23]
	All	[19]
	Theta	[6, 22]
	Alpha	[6, 16, 21, 22, 31, 32, 42]
	Beta	[6, 21, 22, 31, 42]
	Gamma	[31]
	All	[20, 27, 30, 36, 41, 56]
	Bands	**Reference**
	Theta	[2, 23]

6. Discussion

According to the analysis of the literature, it is observed that evaluation of BCI software applications, are carried out through usability tests and mainly are used questionnaires for data collection. Usability tests are carried out in three stages: 1) carrying out the task, 2) data collection through questionnaires and 3) data analysis. This evaluation method allows the collection of subjective data that comes from the opinions of users, this can cause a certain bias. It is also observed that some works incorporate EEG signals to complement the evaluation by measuring emotions.

Studies in other areas such as marketing and medicine use EEG signals and frequency bands to measure workload, fatigue, and emotions. Delta bands allow to detect retention and concentration. Several studies analyze Alpha, Beta, Theta bands in combination for the detection of emotions, comfort, and workload level. These works have allowed us to identify how EEG signals and frequency bands can be linked to usability measures in the field of efficiency and satisfaction, since they have been used in other fields with favorable results. Table 5 shows a summary of the signals and bands, linking them to the measurements and therefore to the respective usability dimension. classifying by dimension, measure, signals, and bands.

Table 5. EEG signals and frequency bands linked to usability measures

Dimension	Measures	Signals	Bands
Efficiency	Workload	F7, P3	Theta
		28 channels	Alpha
		32 channels	Beta
		AF3, AF4, T7 and T8	Gamma
			All
	Fatigue	14 channels	Alpha
	Comfort	32 C- Pz	Theta
			Alpha
			Beta
	Attention	O1	All
		P8, CP6 y O2	Alpha
		32 channels	Delta
	Stress	FP1, FP2, T3, T4, Pz	All
Satisfaction	Emotions	14 channels	Theta
		F3, F4	
		AF3, F4 and FC6	Alpha
		32 channels	
		AFz, F3, F4, CP5 and CP6	Beta
		63 channels	
		AF3, AF4, F3, F4	Gamma
		AF3, AF4, F3, F4, T7, T8	
		FP1, F3, P3, O1	All
		P3, P6, P7 and PO8	

7. Conclusions

A systematic and exhaustive review was carried out [58], defining the research questions, keywords, inclusion, and exclusion criteria, and subsequently the analysis of the results. This research allowed to know how the evaluation process is carried out in the BCI software applications, to identify the use of EEG signals, what information can be obtained from them and if they provide enough elements (data) to be able to occupy them in the evaluation process. Usability evaluations of BCI software applications are carried out through usability test, using questionnaires mainly for data collection. However, the main problem we observe is that the questionnaires give subjective answers, without the certainty of precision and based on the user's opinion. When comparing both techniques for data collection: questionnaires and EEG Signals, it is appreciated that the main advantage of using EEG signals objective data are collected, to measure workload, emotions, and concentration. On the other hand, both in the BCI software applications and in other domains like

Neuromarketing and clinical studies, the EEG signals have been used mainly, but EEG signals have not been applied for usability evaluation.

Our analysis reveals that the EEG signals have been used to measure usability, because studies have used the EGG signals, in order to measure workload, fatigue, attention, comfort and emotions, all these human factors that have been considered elements of efficiency and effectiveness.

Given the results of this work, the research lines are the following: 1) to explore the possibility to measures other human factors through EEG signals like Learning, Utility, Predictability, Consistency, Reliability, Adaptability, Effort, etc., 2) to know what algorithms are used in EEG analysis to obtain measures such as SVM (Vector Support Machine), NN (Neural Network), ICA (Independent Component Analysis), FFT (Fast Fourier Transform), and 3) to apply in BCI software applications EEG signals and bands used in other domains.

References / Список литературы

[1]. Ansari-Asl K., Chanel G., & Pun T. A channel selection method for EEG classification in emotion assessment based on synchronization likelihood. In Proc. of the 2007 15th European Signal Processing Conference, 2007, pp. 1241-1245.

[2]. Antonenko P., Paas F. et al. Using electroencephalography to measure cognitive load. Educational psychology review, vol. 22, no. 4, 2010, 425-438.

[3]. Appriou A., Cichocki A., & Lotte F. Towards robust neuroadaptive HCI: exploring modern machine learning methods to estimate mental workload from EEG signals. In Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems, 2018, pp. 1-6.

[4]. Arai K. Evaluation of users' impact for using the proposed eye based HCI with moving and fixed keyboard by using eeg signals. International Journal of Research and Reviews in Computer Science, vol. 2, no. 6, 2015, 1-7.

[5]. Arai K., & Mardiyanto R. Eye based HCI with moving keyboard for reducing fatigue effects. In Proc. of the 2011 Eighth International Conference on Information Technology: New Generations, 2011, pp. 417-422.

[6]. Bhardwaj A., Gupta A. et al. Classification of human emotions from EEG signals using SVM and LDA Classifiers. In Proc. of the 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN), 2015, pp. 180-185.

[7]. Bos D.P.O., Reuderink B. et al. Human-computer interaction for BCI games: Usability and user experience. In Proc. of the 2010 International Conference on Cyberworlds, 2015, pp. 277-281.

[8]. Charisis V., Hadjidimitriou S. et al. EmoActivity-An EEG-based gamified emotion HCI for augmented artistic expression: The i-Treasures paradigm. Lecture Notes in Computer Science, vol. 9178, 2015, pp. 29-40.

[9]. Chin J.P., Diehl V.A., & Norman K. L. Development of an instrument measuring user satisfaction of the human-computer interface. In Proc. of the SIGCHI Conference on Human Factors in Computing Systems, 1988, pp. 213-218.

[10]. Chowdhury A., Meena Y.K. et al. Active physical practice followed by mental practice using BCI-driven hand exoskeleton: a pilot trial for clinical effectiveness and usability. IEEE Journal of Biomedical and Health Informatics, vol. 22, no. 6, 2018, 1786-1795.

[11]. Crichton N. Visual analogue scale (VAS). Journal of Clinical Nursing, vol. 10, no. 5, 2001, pp. 697-706.

[12]. Erkan E., & Akbaba M. A study on performance increasing in SSVEP based BCI application. Engineering Science and Technology, an International Journal, vol. 21, issue 3, 2018, 421-427.

[13]. Freidman V., & Mielke C. A field guide to Usability Testing. Smashing Magazine, 2013, 85 p.

[14]. Frey J., Daniel M. et al. Framework for electroencephalography-based evaluation of user experience. In Proc. of the 2016 CHI Conference on Human Factors in Computing Systems, 2016, pp. 2283-2294.

[15]. Frey J., Pommereau L. et al. Assessing the zone of comfort in stereoscopic displays using EEG. In CHI'14 Extended Abstracts on Human Factors in Computing Systems, 2014, pp. 2041-2046.

[16]. García Ramírez A.R., Da Silva J.F. et al. User's emotions and usability study of a brain-computer interface applied to people with cerebral palsy. Technologies, vol. 6, no. 1, 2018, article no. 28, 13 p.

[17]. Gentiletti G., Tabernig C., & Acevedo R. Interfaz cerebro-computadora: Estado del arte y desarrollo en Argentina. Revista Argentina de Bioingeniería, Revista SABI, vol. 13, no. 1, 2007, pp. 22-29 (in Spanish).

[18]. Hartson H.R., Andre T.S., & Williges R.C. Criteria for evaluating usability evaluation methods. International Journal of Human-Computer Interaction, vol. 15, no. 1, 2013, 145-181.
[19]. Hosseini S.A., & Khalilzadeh M.A. Emotional stress recognition system using EEG and psychophysiological signals: Using new labelling process of EEG signals in emotional stress state. In Proc. of the 2010 International Conference on Biomedical Engineering and Computer Science, 2010, pp. 1-6.
[20]. Kortelainen J., & Seppänen T. EEG-based recognition of video-induced emotions: selecting subject-independent feature set. In Proc. of the 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2013, pp. 4287-4290.
[21]. Kosiński J., Szklanny K. et al. An analysis of game-related emotions using Emotiv EPOC. In Proc. of the 2018 Federated Conference on Computer Science and Information Systems (FedCSIS), 2018, pp. 913-917.
[22]. Kumar, J. Affective modelling of users in HCI using EEG. Procedia Computer Science, vol. 84, 2016, pp. 107-114.
[23]. Kumar N., & Kumar J. Measurement of cognitive load in HCI systems using EEG power spectrum: an experimental study. Procedia Computer Science, vol. 84, 2016, pp. 70-78.
[24]. Laar B.V.D., Gürkök H. et al. Brain–computer interfaces and user experience evaluation. In Towards Practical Brain-Computer Interfaces, Springer, Berlin, Heidelberg, 2012, pp. 223-237.
[25]. Laubheimer P. Beyond the NPS: Measuring Perceived Usability with the SUS, NASA-TLX, and the Single Ease Question After Tasks and Usability Tests. Nielsen Norman Group, 2018. URL: https://www.nngroup.com/articles/measuring-perceived-usability/.
[26]. Lewis J.R. IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. International Journal of Human-Computer Interaction, vol. 7, no. 1, 1995, pp. 57-78.
[27]. Liu Y., Sourina O., & Nguyen M.K. Real-time EEG-based human emotion recognition and visualization. In Proc. of the 2010 International Conference on Cyberworlds, 2010, pp. 262-269.
[28]. Lührs M., & Goebel R. Turbo-Satori: a neurofeedback and brain–computer interface toolbox for real-time functional near-infrared spectroscopy. Neurophotonics, vol. 4, issue 4, 2017, article no. 041504, 11 p.
[29]. Massa S.M., De Giusti A.E., & Pesado P.M. Métodos de evaluación de usabilidad: una propuesta de aplicación en Objetos de Aprendizaje. In Proc. of the Workshop de Investigadores en Ciencias de la Computación, vol. 14, 2012, pp. 922-926 (in Spanish).
[30]. Murugappan M., Juhari M.R. et al. An Investigation on visual and audiovisual stimulus based emotion recognition using EEG. International Journal of Medical Engineering and Informatics, vol. 1, no. 3, 2009, pp. 342-356.
[31]. Murugappan M., & Murugappan S. Human emotion recognition through short time Electroencephalogram (EEG) signals using Fast Fourier Transform (FFT). In Proc. of the 2013 IEEE 9th International Colloquium on Signal Processing and its Applications, 2013, pp. 289-294.
[32]. Murugappan M., Rizon M. et al. Lifting scheme for human emotion recognition using EEG. In Proc. of the 2008 International Symposium on Information Technology, 2008, vol. 2, pp. 1-7.
[33]. Nielsen J. Usability inspection methods. In Proc. of the Conference Companion on Human Factors in Computing Systems, 1994, pp. 413-414.
[34]. Nielsen J. Usability Engineering. Morgan Kaufmann, 1993, 376 p.
[35]. Ortega-Gijón Y.N., & Mezura-Godoy C. Usability evaluation process of brain computer interfaces: an experimental study. In Proc. of the IX Latin American Conference on Human Computer Interaction, 2019, pp. 1-8.
[36]. Sourina O., Liu Y. et al. EEG-based personalized digital experience. Lecture Notes in Computer Science, vol. 6766, 2011, pp. 591-599.
[37]. Parra L.C., Spence C.D. et al. Response error correction-a demonstration of improved human-machine performance using real-time EEG monitoring. IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 11, no. 2, 2003, 173-177.
[38]. Pasqualotto E., Federici S. et al. Usability of brain computer interfaces. In Everyday Technology for Independence and Care. IOS Press, 2011, pp. 481-488.
[39]. Pradhapan P., Großekathöfer U. et al. Toward practical BCI solutions for entertainment and art performance. In Brain–Computer Interfaces Handbook: Technological and Theoretical Advances, CRC Press, 2018, pp. 65-115.
[40]. Putze F., Scherer M., & Schultz T. Starring into the void? Classifying Internal vs. External Attention from EEG. In Proc. of the 9th Nordic Conference on Human-Computer Interaction, 2016, pp. 1-4.

[41]. Puwakpitiyage C.A., Rao V.R. et al. A Proposed Web Based Real Time Brain Computer Interface (BCI) System for Usability Testing. International Journal of Online & Biomedical Engineering, vol. 15, no. 8, 2019, pp. 111-123.
[42]. Ramirez R., & Vamvakousis Z. Detecting emotion from EEG signals using the emotive epoc device. Lecture Notes in Computer Science, vol. 7670, 2012, pp. 175-184.
[43]. Nielsen Norman Group. Usability Testing 101. URL: https://www.nngroup.com/articles/usability-testing-101/.
[44]. Rhiu I., Lee Y. et al. Toward usability evaluation for brain–computer interfaces. In Brain–Computer Interfaces Handbook, CRC Press, 2018, pp. 563-584.
[45]. Snyder H. Literature review as a research methodology: An overview and guidelines. Journal of Business Research, vol. 104, 2019, pp. 333-339.
[46]. Spencer D., Warfel T. Card Sorting. Boxes and arrows: A Definitive Guide, 2014. URL: https://boxesandarrows.com/card-sorting-a-definitive-guide/.
[47]. Stein A., Yotam Y. et al. EEG-triggered dynamic difficulty adjustment for multiplayer games. Entertainment computing, vol. 25, 2018, pp. 14-25.
[48]. Spüler M. A high-speed brain-computer interface (BCI) using dry EEG electrodes. PloS one, vol. 12, no. 2, 2017, article no. e0172400, 12 p.
[49]. Taherian S., Selitskiy D. et al. Are we there yet? Evaluating commercial grade brain–computer interface for control of computer applications by individuals with cerebral palsy. Disability and Rehabilitation: Assistive Technology, vol. 12, no. 2, 2017, pp. 165-174.
[50]. Theofanos M., & Quesenbery W. Towards the design of effective formative test reports. Journal of Usability Studies, vol. 1, no. 1, 2005, pp. 27-45.
[51]. Tidoni E., Abu-Alqumsan M. et al. Local and remote cooperation with virtual and robotic agents: a P300 BCI study in healthy and people living with spinal cord injury. IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 25, no. 9, 2017, pp. 1622-1632
[52]. Valderrama C.E., & Ulloa G. Spectral analysis of physiological parameters for emotion detection. In Proc. of the 2012 XVII Symposium of Image, Signal Processing, and Artificial Vision (STSIVA), 2012, pp. 275-280.
[53]. Villegas A., Salvatierra E. et al. Reconocimiento de patrones de actividad cerebral asociados a tareas mentales mediante RNA para una interfaz cerebro computador. Revista Ingeniería UC, vol. 15, no. 1, 2008, 88-92 (in Spanish).
[54]. Wang Q., Sourina O., & Nguyen M. K. Eeg-based «serious» games design for medical applications. In Proc. of the 2010 International Conference on Cyberworlds, 2010, pp. 270-276.
[55]. Xing X., Wang Y. et al. A high-speed SSVEP-based BCI using dry EEG electrodes. Scientific reports, vol. 8, no. 1, 2018, pp. 1-10.
[56]. Zhang J., Chen M. et al. PNN for EEG-based Emotion Recognition. In Proc. of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2016, pp. 002319-002323.
[57]. Kurniawan, S. Interaction design: Beyond human–computer interaction by Preece, Sharp and Rogers. Universal Access in the Information Society, vol. 3, issue 3-4, 2001, p. 289.
[58]. Kitchenham B., Brereton O.P. et al. Systematic literature reviews in software engineering–a systematic literature review. Information and Software Technology, vol. 51, no. 1, 2009, pp. 7-15.
[59]. Tello-Rodríguez M., Ocharán-Hernández J.O. et al. A design guide for usable web APIs. Programming and Computer Software, vol. 46, issue 8, 2020, pp. 584-593.
[60]. Lukin V.N., Dzyubenko A.L., & Chechikov Y.B. Approaches to user interface development. Programming and Computer Software, vol. 46, issue 5, 2020, pp. 316-323 / Лукин В.Н., Дзюбенко А.Л., Чечиков Ю.Б. Подходы к разработке пользовательского интерфейса. Программирование, том 46, no. 5, 2020 г., стр. 16-24.

Information about authors / Информация об авторах

Yoselyn Nohemí ORTEGA-GIJÓN – PhD student in Computer Science from the Universidad Veracruzana. Master's in Computer Systems from the Instituto Tecnológico Superior de Misantla. Main research interests: Brain Computer Interaction, Usability evaluation, EEG processing signals.

Йоселин Нохеми ОРТЕГА-ХИХОН – аспирант. Степень магистра в области компьютерных систем получила в Высшем технологическом институте в Мисантле. Основные научные

интересы: взаимодействие мозга с компьютером, оценка пригодности к использованию, обработка сигналов ЭЭГ.

Carmen MEZURA-GODOY – PhD in Computer Science from the University of Savoie in France. Professor at the Faculty of Statistics and Informatics of the University of Veracruz in Mexico. Main research interests: Human-Computer Interaction, User eXperience-UX, Computer Support Collaborative Work, Visualization and Multiagent Systems.

Кармен МЕЗУРА-ГОДОЙ получила степень доктора компьютерных наук в Университете Савойи во Франции. Профессор факультета статистики и информатики Университета Веракруса в Мексике. Основные исследовательские интересы: взаимодействие человека и компьютера, компьютерная поддержка совместной работы, визуализация и мультиагентные системы.

DOI: 10.15514/ISPRAS-2022-34(3)-11

Реализация распределённых и параллельных вычислений в сети SDN

И.Б. Бурдонов, ORCID: 0000-0001-9539-7853 <igor@ispras.ru>
Н.В. Евтушенко, ORCID: 0000-0002-4006-1161 <evtushenko@ispras.ru>
А.С. Косачев, ORCID: 0000-0001-5316-3813 <kos@ispras.ru>

Институт системного программирования им. В.П. Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25

Аннотация. В статье рассматривается выполнение на плоскости данных SDN, моделируемой конечным связным неориентированным графом физических связей, программы задания, которая понимается в духе парадигмы объектно-ориентированного программирования как состоящая из объектов и сообщений, которыми объекты могут обмениваться. Объекты реализуется в хостах, причём в одном хосте может быть реализовано несколько разных объектов, а один и тот же объект может быть реализован в нескольких хостах. Сообщения между объектами, реализованными в разных хостах, помещаются в пакеты, маршрутизацию которых исполняют коммутаторы на основе идентификаторов, присвоенных пакетам и помещаемых в заголовки пакетов как набор значений некоторых параметров пакетов. В работе решаются две задачи: 1) минимизация числа идентификаторов, 2) настройка коммутаторов для реализации путей, которые должны проходить пакеты. Эти задачи решаются в двух случаях: A) пакет, предназначенный для некоторого объекта, должен попасть ровно в один хост, в котором реализован этот объект, B) пакет может попадать в несколько хостов, но в одном и только одном из них должен быть реализован нужный объект. Показано, что задача 1 в случае A эквивалентна задаче о покрытии множества, а минимальное число идентификаторов в наихудшем случае равно $min\{ n, m \}$, где n число объектов, а m число хостов, реализующих объекты. В случае B задача является специальной модификацией задачи о покрытии множества, высказывается гипотеза о том, что минимальное число идентификаторов в наихудшем случае равно $min\{ \lfloor lb(n + 1) \rfloor, m \}$. Пока получена верхняя оценка $O(min \{ ln (min \{ n, m \}) \cdot ln (n, m) \})$. Для решения задачи 2 в случаях A и B предложены алгоритмы настройки коммутаторов сложности, соответственно, $O(m)$ и $O(k m)$, где m число рёбер графа физических связей, а k результат решения задачи 1 в случае B как число требуемых идентификаторов пакетов.

Ключевые слова: распределенные и параллельные вычисления, программно-конфигурируемые сети, маршрутизация пакетов, задача о покрытии множества

Для цитирования: Бурдонов И.Б., Евтушенко Н.В., Косачев А.С. Реализация распределённых и параллельных вычислений в сети SDN. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 159-172. DOI: 10.15514/ISPRAS–2022–34(3)–11

Благодарности. Работа выполнена при поддержке Российского фонда фундаментальных исследований, проект 20-07-00338 А.

Implementation of distributed and parallel computing in the SDN network

I.B. Burdonov, ORCID: 0000-0001-9539-7853 <igor@ispras.ru>
N.V. Yevtushenko, ORCID: 0000-0002-4006-1161 <evtushenko@ispras.ru>
A.S. Kossatchev, ORCID: 0000-0001-5316-3813 <kos@ispras.ru>

Ivannikov Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia

Abstract. The paper discusses the execution of a program of tasks on the SDN data plane, modeled by a finite connected undirected graph of physical connections; the execution is understood in the sense of the object-oriented programming paradigm as consisting of objects and messages that objects can exchange. Objects are implemented in hosts. Several different objects can be implemented in one host and the same object can be implemented in several hosts. Messages between objects implemented in different hosts are transmitted in packets which are routed by switches based on identifiers assigned to packets that is on a set of values of some packet parameters in the packet header. Two problems are tackled in the work: 1) minimizing the number of identifiers, 2) setting up switches to implement the paths that packets should take place. These tasks are solved in two cases: A) a packet intended for some object must get into exactly one host in which this object is implemented, B) a packet can get into several hosts, but the desired object must be implemented in one and only one of them. It is shown that problem 1 in case A is equivalent to the set covering problem, and the minimum number of identifiers in the worst case is $min\{n, m\}$ where n is the number of objects, and m is the number of hosts implementing objects. In case B, the problem is a special modification of the set covering problem, the hypothesis is proposed that the minimum number of identifiers in the worst case is $min\{\lfloor lb(n+1)\rfloor, m\}$. So far, an upper bound is $O(min\{ln(min\{n, m\})\cdot ln(n, m)\})$. To solve problem 2 in cases A and B, algorithms for switches' setting are proposed which have the complexity $O(m)$ and $O(km)$, , respectively, where m is the number of the edges of the graph of physical connections and k is the number of the required packet identifiers.

Keywords: distributed and parallel computations, software-defined networks (SDN), packet routing, set cover problem

For citation: Burdonov I.B., Yevtushenko N.V., Kossatchev A.S. Implementation of distributed and parallel computing in the SDN network. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 159-172 (in Russian). DOI: 10.15514/ISPRAS–2022–34(3)–11

Acknowledgements. The work was supported by the Russian Foundation for Basic Research, project 20-07-00338 A.

1. Введение

Рассматривается плоскость данных SDN, моделируемая конечным связным неориентированным графом G, терминальные вершины (вершины степени 1) которого – *хосты*, а внутренние вершины (вершины степени больше 1) – *коммутаторы*. Этот граф называется графом физических связей. *Пакеты* генерируются хостами и двигаются по путям, определяемым правилами коммутаторов. Коммутаторы не меняют пакеты и только пересылают принятые ими пакеты. Пакет должен пройти путь от хоста до хоста, в котором все промежуточные вершины коммутаторы. Такой путь называется *полным*. Коммутаторы различают пакеты по их *идентификаторам*. Идентификатор моделирует набор значений параметров пакета в заголовке пакета. Коммутатор s, приняв пакет с идентификаторов d от соседа a, посылает пакет соседу b в зависимости от d и a, что определяется *правилом пакета* вида (d, a, s, b). Пакет может быть послан нескольким соседям (клонирование пакета), если в таблице коммутатора s есть несколько правил, отличающихся только принимающим узлом b.

В данной работе рассматривается выполнение на плоскости данных SDN *задания*, которое задаётся 1) *начальным хостом*, инициирующим выполнение задания, 2) *описанием*

программы, 3) *набором параметров* выполнения задания. *Программа* понимается в духе парадигмы объектно-ориентированного программирования как состоящая из *объектов* и *сообщений*, которыми объекты могут обмениваться. Объект реализуется в хосте, причём в одном хосте может быть реализовано несколько разных объектов, а один и тот же объект может быть реализован в нескольких хостах (в этом случае говорят о нескольких *экземплярах* одного объекта). Если сообщение передаётся от объекта a к объекту b, и эти объекты реализованы в одном хосте, то передача сообщения – это просто вызов объекта b из объекта a внутри этого хоста. Если же объекты a и b реализованы в разных хостах h_a и h_b, соответственно, то сообщение передаётся в пакете, направляемом по плоскости данных SDN из хоста h_a в хост h_b. Параллелизм обеспечивается передачей нескольких сообщений от одного объекта a нескольким объектам $b_1, ..., b_k$, которые могут быть реализованы в нескольких хостах. Выполнение объекта b может зависеть от результатов выполнения нескольких объектов $a_1, ..., a_n$. Это означает, что выполнение объекта b начинается после получения им сообщений от объектов $a_1, ..., a_n$, содержащих аргументы вызова объекта a.

Выполнение задания начинается в начальном хосте, который инициирует выполнение задания рассылкой сообщений тем объектам, с которых должно начинаться выполнение задания согласно описанию программы, с теми параметрами, которые определяются набором параметров задания.

Если объекты a и b реализованы в разных хостах h_a и h_b, соответственно, сообщение от a к b передаётся в пакете, который содержит 1) идентификатор пакета, помещаемый хостом h_a в заголовок пакета как значения некоторых его параметров и предназначенный для маршрутизации пакетов через коммутаторы, 2) идентификатор объекта b, которому направляется сообщение, 3) само сообщение для объекта b, 4) описание программы или части программы, которая ещё не выполнена. Хост h_b, получив пакет с сообщением для объекта b, вызывает этот объект и, при необходимости, формирует пакеты для передачи сообщений объектам, вызываемым из объекта b и реализованным в других хостах.

Мы предполагаем, что идентификатор пакета формируется тем хостом, который этот пакет отправляет, и может отличаться от идентификаторов принятых им пакетов.

Мы будем считать, что распределение объектов по хостам может быть любым, но оно задано, а начальным хостом может быть любой хост. При этом мы предполагаем, что каждый объект, который может вызываться в программе задания, реализован хотя бы в одном хосте.

Относительно пакета с сообщением для объекта b возникают вопросы о допустимости следующих ситуаций.

1) Пакет не попадает ни в один из хостов, в которых реализован объект b.
2) Пакет в результате клонирования в коммутаторах попадает в несколько хостов, в которых реализован объект b.
3) Пакет в результате клонирования в коммутаторах проходит несколько дублирующих путей, заканчивающихся в одном хосте, в котором реализован объект b.
4) Пакет попадает в хост, в котором не реализован объект b.

Мы считаем, что настройка коммутаторов и правила генерации идентификаторов пакетов должны исключать ситуации 1 (объект b не вызывается) и 2 (вызывается несколько экземпляров объекта b). В ситуации 3 хост, повторно получивший пакет, прошедший дублирующий путь, должен этот пакет игнорировать. Допустимость этой ситуации зависит от того, способен ли хост «помнить», какие пакеты он получал, чтобы иметь возможность игнорировать повторные пакеты. В данной работе мы рассматриваем случай, когда хосты не обладают такой способностью и, следовательно, ситуация 3 также должна быть запрещена. Что касается ситуации 4, то возможны разные ответы.

Сначала мы рассмотрим случай, когда ситуация 4 запрещена, т.е. пакет с сообщением для объекта b должен пройти без клонирования один и только один полный путь до хоста, в котором этот объект реализован. Это случай, когда клонирование пакетов запрещено.

Затем мы ослабим требования, разрешив пакету с сообщением для объекта b приходить в хост h, в котором этот объект не реализован. В этой ситуации предполагается, что хост h просто игнорирует такой пакет. Таким образом, пакет с сообщением для объекта b должен пройти, быть может, несколько полных путей за счёт его клонирования в коммутаторах, но среди этих путей должен быть один и только один путь, заканчивающийся в хосте, в котором реализован объект b. Это случай, когда клонирование пакетов разрешено.

В обоих случаях нас будут интересовать следующие вопросы.

1) Как минимизировать число идентификаторов пакетов, требуемых для выполнения п.1?
2) Возможна ли такая настройка коммутаторов (т.е. задание соответствующих правил в таблицах коммутаторов), чтобы любое задание, инициированное в любом начальном хосте, можно было выполнить?

Число требуемых идентификаторов пакетов – это важная характеристика сети, поскольку от неё зависит размер таблиц коммутаторов.

В данной работе нас не будут интересовать вопросы надёжности, безопасности и оптимизации нагрузки на хосты и коммутаторы, т.е. мы не будем их учитывать при рассмотрении возможности реализации требуемой настройки коммутаторов и минимизации числа идентификаторов пакетов.

Мы будем решать две задачи: 1) Минимизация числа идентификаторов пакетов, соответствующих объектам, реализованным в хостах. 2) Настройка коммутаторов для выбранного соответствия идентификаторов пакетов и объектов. Эти задачи имеют разные решения для двух случаев передачи пакетов: A) Без клонирования, когда пакет должен быть доставлен только одному хосту, в котором реализован требуемый объект. B) С клонированием, когда пакет может быть доставлен нескольким хостам, но в одном и только одном из них должен быть реализован требуемый объект.

2. Задача 1 (минимизация числа идентификаторов пакетов) в случае A (без клонирования)

Мы рассматриваем задачу минимизации числа идентификаторов пакетов с дополнительным требованием: идентификатор пакета, предназначенного для того или иного объекта, не зависит от хоста-отправителя. Решение задачи зависит только от множества объектов, множества хостов и распределения объектов по хостам, но не зависит от графа физических связей. С другой стороны, при решении задачи 2 (настройка коммутаторов) уже потребуется граф физических связей, а хост будет пониматься как терминальная вершина этого графа.

Множество всех объектов, реализованных во всех хостах, будем обозначать X. Хост будем моделировать множеством объектов, которые реализованы в этом хосте. Множество хостов есть семейство H подмножеств множества X. Соответствие идентификаторов пакетов и объектов, реализованных в хостах, задаётся функцией f, отображающей каждый объект $x \in X$ в некоторый хост $f(x)$, реализующий этот объект, т.е. $x \in f(x)$. Требуется найти функцию $f: X \to H$ с минимальной мощностью образа $|f(X)|$. Идентификаторы пакетов взаимно-однозначно соответствуют хостам из образа $f(X)$. Если пакет должен быть направлен объекту x, то идентификатор этого пакета соответствует хосту $f(x)$, в котором реализован объект x.

Теорема 1. Задача 1 для случая A эквивалентна задаче о покрытии множества: для конечного множества X и семейства его подмножеств H требуется найти подсемейство $U \subseteq H$ наименьшей мощности, объединением которого является X, т.е. $\cup U = X$.

Доказательство. Утверждение непосредственно следует из того, что для каждой функции $f: X \to H$ её образ есть подсемейство семейства H, $f(X) \subseteq H$, и для любого подсемейства $U \subseteq H$ такого, что $\cup U = X$, можно определить функцию $f: X \to H$, выбрав для каждого объекта $x \in X$ один такой хост $h \in U$, что $x \in h$.

□

В данной работе не предлагается каких-то новых решений классической задачи о покрытии множества, нам было важно только показать, что наша задача 1 в случае A эквивалентна ей. Задача о покрытии множества одна из старейших и наиболее изученных *NP*-трудных задач [1]. Существуют различные полиномиальные алгоритмы (в частности, жадный), дающие приближённые решения с той или иной точностью, а также точные полиномиальные алгоритмы для специальных классов семейств *H*. Кроме того, рассматривались различные обобщения этой задачи, например, [1]. На эту тему есть много работ в мировой литературе. Обзор некоторых алгоритмов и методов можно найти в [3][2],[3].

Задача о покрытии множеств имеет наглядную интерпретацию в терминах матриц. Пусть $X = \{ x_1, ..., x_n \}$ и $H = \{ h_1, ..., h_m \}$, где $n = |X|$ число объектов, $m = |H|$ число хостов. Тогда эту пару X и H можно задать матрицей M размером $n \times m$, в которой строки соответствуют объектам, столбцы – хостам, а для $i = 1, ..., n$ и $j = 1, ..., m$ ячейка матрицы $M(i, j) = 1$, если i-й объект реализован в j-м хосте, т.е. $x_i \in h_j$, и $M(i, j) = 0$ в противном случае. При этом каждая строка матрицы должна содержать хотя бы одну «1» (каждый объект должен быть реализован хотя бы в одном хосте). Такую матрицу будем называть *правильной*. Требуется найти минимальное (по числу столбцов) покрывающее подмножество столбцов, т.е. такое, чтобы в каждой строке хотя бы один из этих столбцов содержал «1». Эти столбцы образуют матрицу K с n строками, которую будем называть *A-производной* матрицей (для случая A). Функция f отображает объект x_i в хост, соответствующий столбцу A-производной матрицы K, содержащему «1» в i-й строке.

Для случая A (без клонирования) через $k_A(M)$ обозначим число столбцов в минимальной (по числу столбцов) матрице A-производной от заданной матрицы M. Через $k_B(n, m)$ обозначим максимум $k_A(H)$ по всем возможным матрицам размера $n \times m$. $k_A(n, m) = max\{ k_A(H) : |H| = m \ \& \ |X| = n \}$. Докажем следующее простое утверждение.

Теорема 2. Для каждых $n \geq 1$ и $m \geq 1$ имеет место $k_A(n, m) = min\{ n, m \}$.

Доказательство. Поскольку в каждой строке матрицы M имеется «1» в некотором столбце, выберем для каждой строки один такой столбец. Выбранные столбцы, очевидно, образует A-производную матрицу, в которой число столбцов не превышает $min\{ n, m \}$, т.е. $k_A(n, m) \leq min\{ n, m \}$. Эта оценка достигается в матрице M, в которой верхняя левая подматрица размером $min\{ n, m \}$ содержит «1» на главной диагонали и «0» в остальных ячейках; если $n > m$, то нижние $n - m$ строк заполнены «1», а если $m > n$, то правые $m - n$ столбцов заполнены «0». Любая A-производная матрица должна содержать первые $min\{ n, m \}$ столбцов, а минимальная A-производная матрица – только их.

3. Задача 1 (минимизация числа идентификаторов пакетов) в случае B (с клонированием)

В случае B (с клонированием) пакет, предназначенный некоторому объекту, могут получить несколько хостов, но в одном и только в одном из них должен быть реализован этот объект. Это означает, что функция имеет сигнатуру $f: X \to 2^H$ и должна удовлетворять требованию $\forall\, x \in X \ \exists!\, h \in f(x)\ x \in h$. Требуется найти такую функцию с минимальной мощностью образа $|f(X)|$. Идентификаторы пакетов взаимно-однозначно соответствуют множествам хостов из образа $f(X)$. Если пакет должен быть направлен объекту x, то идентификатор этого пакета соответствует множеству хостов $f(x)$, в одном и только одном из которых реализован объект x.

Переформулируем эту задачу в терминах матриц. *Суммой* одного или нескольких столбцов двоичной матрицы M будем называть столбец (матрицу размера $n \times 1$), в каждой i-й строке которого находится сумма чисел, находящихся в i-й строке в суммируемых столбцах (или одному числу, если суммируется один столбец) матрицы M. Матрицу K назовём *B-производной* от правильной матрицы M (для случая B), если каждый столбец матрицы K есть

сумма одного или нескольких столбцов матрицы M, а каждая строка содержит хотя бы одну «1». Требуется найти B-производную матрицу K с минимальным числом столбцов. Функция f отображает объект x_i в множество хостов, соответствующих столбцу B-производной матрицы K, содержащему «1» в i-й строке.

Эта задача эквивалентна специальной модификации задачи о покрытии множества, когда в терминах матриц столбцы рассматриваемой матрицы – это все возможные суммы столбцов исходной матрицы, в которых все числа большие 1 заменены на 0.

Для случая B (с клонированием) через $k_B(M)$ обозначим число столбцов в минимальной (по числу столбцов) матрице B-производной от заданной матрицы M. Через $k_B(n, m)$ обозначим максимум $k_B(H)$ по всем возможным матрицам размера $n \times m$. $k_B(n, m) = max\{ k_B(H) : |H| = m \ \& \ |X| = n \}$.

Теорема 3. $k_B(n, m) \le m$.

Доказательство. Любая правильная матрица M размером $n \times m$, очевидно, B-производна от самой себя и содержит m столбцов. Поэтому $k_B(M) \le m$ для любой правильной матрицы M. Тем самым, $k_B(n, m) \le m$. □

Обозначим через $M_0(m)$ матрицу размером $(2^m - 1) \times m$, строки которой являются двоичными m-разрядными представлениями чисел от 1 до $2^m - 1$.

Лемма 1. Для каждого $m \ge 1$ матрица $M_0(m)$ правильная и $k_B(M_0(m)) = m$.

Доказательство. В матрице $M_0(m)$ есть одна строка, состоящая из одних «1», поэтому в любой B-производной матрице K должен быть столбец, совпадающий с одним из столбцов матрицы $M_0(m)$. Выберем этот столбец матрицы $M_0(m)$. Далее для невыбранных столбцов матрицы $M_0(m)$ найдётся одна строка, в которой в этих столбцах одни «1», а в уже выбранном столбце «0». Поэтому в матрице K должен быть столбец, являющийся суммой одного из невыбранных столбцов матрицы $M_0(m)$ и, быть может, каких-то уже выбранных столбцов матрицы $M_0(m)$. Выберем этот столбец матрицы $M_0(m)$. И так далее. На каждом шаге для невыбранных столбцов матрицы $M_0(m)$ имеется одна строка, в которой в этих столбцах одни «1», а в уже выбранных столбцах одни «0». Поэтому в матрице K должен быть столбец, являющийся суммой одного из невыбранных столбцов матрицы $M_0(m)$ и, быть может, каких-то уже выбранных столбцов матрицы $M_0(m)$. Выберем этот столбец матрицы $M_0(m)$. В результате окажется, что в матрице K число столбцов равно m. □

Теорема 4. Для каждого $n \ge 1$ и $m \ge 1$ существует правильная матрица M размером $n \times m$, для которой $k_B(M) = min\{ \lfloor lb(n+1) \rfloor, m \}$.

Доказательство. Если $n = 2^m - 1$, то по лемме 1 Лемма 1 искомой матрицей является матрица $M_0(m)$ и $k_B(M_0(m)) = m = lb(n+1) = \lfloor lb(n+1) \rfloor = min\{ \lfloor lb(n+1) \rfloor, m \}$. Если $n > 2^m - 1$, то искомая матрица M получается из матрицы $M_0(m)$ добавлением недостающих строк, каждая из которых содержит «1». Тогда $k_B(M) = m = min\{ \lfloor lb(n+1) \rfloor, m \}$. Если $n < 2^m - 1$, то искомая матрица M получается из матрицы $M_0(\lfloor lb(n+1) \rfloor)$ добавлением недостающих столбцов, заполненных «0», и недостающих строк, каждая из которых содержит «1». Тогда $k_B(M) = \lfloor lb(n+1) \rfloor = min\{ \lfloor lb(n+1) \rfloor, m \}$. □

На основании теорем 3 и 4 можно предложить гипотезу о том, что $k_B(n, m) = min\{ \lfloor lb(n+1) \rfloor, m \}$. В настоящее время эта гипотеза не доказана и не опровергнута. Наиболее интересные результаты получены в [4]. Там задача 1 (минимизация числа идентификаторов) в случае B (с клонированием) названа задачей точного покрытия набором подмножеств, показана её связь с задачей о максимальной точной выполнимости монотонной КНФ, доказана APX-трудность[1] обоих задач и для нашей задачи найдено решение, гарантирующее размер точного покрытия не более, чем $O(min \{ ln (min \{ n, m \}) \cdot ln (n, m) \})$.

[1] Класс APX – класс оптимизационных задач, для которых существуют полиномиальные приближенные алгоритмы с мультипликативной ошибкой, не превышающей некоторой абсолютной константы.

4. Задача 2 (настройка коммутаторов) в случае A (без клонирования)

Пусть выбрано некоторое решение задачи 1 в случае A, т.е. выбрана функция $f: X \to H$ (с минимальной или близкой к минимальной мощностью образа $|f(X)|$), отображающая объект x в хост $f(x)$, и установлено взаимно-однозначное отображение d образа $f(X)$ во множество идентификаторов пакетов. Пакет, направляемый объекту x, будет иметь идентификатор $d(f(x))$.

Напомним, что при решении задачи 1 мы моделировали хост множеством реализуемых им объектов. Теперь задача 2 заключается в том, чтобы так настроить коммутаторы на плоскости данных SDN, чтобы пакет, предназначенный объекту x и имеющий идентификатор $d(f(x))$, будучи отправлен с некоторого (любого) хоста, проходил один и только один полный путь до хоста $f(x)$, в котором реализован объект x. Но здесь мы должны вспомнить, что на плоскости данных SDN может быть несколько хостов, реализующих одно и то же множество объектов $f^{-1}(f(x))$.Иными словами, пакет с идентификатором $d(f(x))$ должен проходить полный путь до одного и только одного хоста из множества U хостов, реализующих одно и то же множество объектов $f^{-1}(f(x))$. Пакеты, посылаемые из разных начальных хостов, могут проходить полные пути, заканчивающиеся в разных хостах из множества U.

Мы будем решать задачу 2, отвлекаясь от функции f. Для случая A формулировка задачи такая: Пусть дан граф физических связей G, идентификатор пакетов d и подмножество хостов U. Требуется так настроить коммутаторы, чтобы пакет с идентификатором d, отправленный с одного (любого) хоста, проходил один и только один полный путь до некоторого хоста из множества U. Граф G будем задавать множеством V его вершин (хостов и коммутаторов) и функцией N, задающей для каждой вершины v множество всех её соседей $N(v)$.

Правило коммутатора имеет вид (d, a, s, b) и означает, что коммутатор s, приняв пакет с идентификаторов d от соседа a, посылает пакет соседу b. Отсюда следует, во-первых, что маршрутизации пакетов с разными идентификаторами не зависят друг от друга, поскольку определяются непересекающимися наборами правил. Во-вторых, любое множество $P(d)$ полных путей для данного идентификатора d пакета однозначно определяет правила настройки коммутаторов для идентификатора d, которые, в свою очередь, однозначно определяют множество полных путей $P(d)\downarrow\uparrow$, которые могут проходить пакеты с идентификатором d, как замыкание по дугам множества $P(d)$ **Ошибка! Источник ссылки не найден.**, [6]. Это замыкание по дугам определяется так: если есть два полных пути $p{\cdot}e{\cdot}q$ и $p'{\cdot}e{\cdot}q'$, где p, q, p' и q' пути, а e дуга графа G, то в замыкании будут эти полные пути вместе с полными путями $p{\cdot}e{\cdot}q'$ и $p'{\cdot}e{\cdot}q$. Известно, что при замыкании по дугам полных путей может возникать зацикливание пакетов, когда пакет будет бесконечно двигаться по циклу рёбер и бесконечно клонироваться в точке разветвления цикла и постфикса пути, ведущего в конечный хост, а также дублирование, когда два разных пути имеют общий начальный хост и общий конечный хост, из-за чего конечный хост получит один и тот же пакет дважды.

Таким образом, для заданного идентификатора d и соответствующего ему непустого множества хостов U нужно так настроить коммутаторы плоскости данных SDN, т.е. определить такие правила коммутаторов для данного идентификатора d, чтобы они порождали замкнутое по дугам множество путей $P(d)$ без зацикливания и дублирования такое, чтобы из каждого хоста в этом множестве был один и только один полный путь до некоторого хоста из множества U. Для этого предлагается приведённый ниже алгоритм 1, который для каждого хоста строит кратчайший путь (один из кратчайших путей) до некоторого хоста из множества U.

Алгоритм 1 основан на обходе неориентированного графа в ширину и может рассматриваться как модификация алгоритма Дейкстра, только вместо вычисления длины кратчайшего пути выполняется настройка коммутаторов вдоль пути. Идея алгоритма заключается в следующем. Начиная с множества хостов U по графу распространяется «волна» построения путей с помощью пометок вершин. Фронт этой волны F состоит из

помеченных вершин v с одним и тем же расстоянием до ближайшего хоста из множества U, тогда как остальные помеченные вершины находятся на меньшем расстоянии от U. Для каждой вершины v из фронта волны просматриваются её соседи и для каждого соседа w, который ещё не помечен, выставляется пометка и запоминается его сосед $p(w) = v$, через которого ведёт кратчайший путь из вершины w до ближайшего хоста из множества U. Если вершина v коммутатор, в нём устанавливается правило $(d, w, v, p(v))$. Если вершина w коммутатор, она помещается в новый фронт волны F_{next}. Когда просмотрены все вершины фронта F, начинается новый шаг, на котором фронтом волны становится F_{next}. Алгоритм завершает свою работу, когда просмотрены все вершины и фронт F стал пустым.

Алгоритм_1 (V, N, R, d, U) /* Настройка коммутаторов для кратчайшего доступа (без зацикливания и дублирования) от каждого хоста до ближайшего хоста из множества хостов U */

***Input*:** множество V вершин графа физических связей, функция N, задающая для каждой вершины v множество $N(v)$ её соседей, текущая настройка коммутаторов R, задающая для каждого коммутатора v текущее множество правил вида (d', a, s, b), где $d' \neq d$, новый идентификатор пакетов d, непустое множество хостов U, соответствующее d.

***Output*:** новая настройка коммутаторов R.

$r(v)$ – пометка вершины v (***true***/***false***),

$p(v)$ – сосед вершины v на пути к ближайшему хосту из U,

F – фронт волны (множество вершин) от хостов из U до других хостов,

F_{next} – множество F на следующем шаге.

$F = \varnothing$; $F_{next} = \varnothing$;

for all $v \in V$ ***do*** $r(v) =$ ***false***; /* во всех вершинах нет пометок */

for all $v \in U$ ***do*** $r(v) =$ ***true***; $F = F \cup \{ v \}$; /* в хостах из U есть пометки */

while $F \neq \varnothing$ ***do*** /* пока фронт волны не пуст */

for all $v \in F$ ***do*** /* вершины v из фронта волны */

for all $w \in N(v)$ & $\neg r(w)$ ***do*** /* непомеченные соседи вершины v */

$r(w) =$ ***true***; $p(w) = v$; /* теперь сосед w помечен */

if $|N(v)| > 1$ ***then*** /* v коммутатор */

$R(v) = R(v) \cup \{ (d, w, v, p(v)) \}$; /* правило коммутатора v */

if $|N(w)| > 1$ ***then*** /* w коммутатор */

$F_{next} = F_{next} \cup \{ w \}$; /* помещаем w в следующий фронт */

return R;

Теорема 5. Алгоритм 1 правильно настраивает коммутаторы для решения задачи 2 в случае A и имеет сложность $O(m)$, где m число ребер графа G физических связей.

Доказательство. Сначала докажем, что на каждом i-ом шаге, $i = 0, 1, ...$, все вершины v фронта волныF имеют одно и то же расстояние i до ближайшего хоста из U, другие помеченные вершины находятся на меньшем расстоянии, а все их соседи помечены. Действительно, вначале, на нулевом шаге, $F = U$, т.е. для каждой вершины v из F расстояние равно 0, а других помеченных вершин нет. Пусть утверждение верно на i-ом шаге и вершины фронта волны F находятся на расстоянии i от ближайшего хоста из U. В следующий фронт волны попадают соседи вершин из фронта волны, которые являются коммутаторами и ещё не помечены, т.е. они тоже имеют одно и то же расстояние $i + 1$ до ближайшего хоста из U. Все соседи вершин из старого фронта волны оказываются помеченными.

Поскольку граф связный, при окончании алгоритма все вершины окажутся помеченными. На каждом шаге для каждого непомеченного соседа w вершины v из фронта F запоминается $p(w) = v$. Поэтому после завершения алгоритма для каждого хоста v, кроме хостов из U, который получил пометку на i-м шаге последовательность $P(v) = (p^0(v) = v, p^1(v) = p(v), p^2(v) = p(p(v)), ..., p^i(v))$ есть последовательность вершин на кратчайшем пути длиной i от хоста v до ближайшего хоста $p^i(v)$ из U. Алгоритм генерирует правила по внутренних вершинах этого пути, которые являются коммутаторами: для $j = 1, ..., p^{i-1}(v)$ в коммутаторе $p^j(v)$ устанавливается правило $(d, p^{j-1}(v), p^j(v), p^{j+1}(v))$. Суммарно эти правила для коммутаторов на пути $P(v)$ обеспечивают проход пакета с идентификатором d из хоста v по пути $P(v)$ до хоста $p^i(v)$ из U. Совокупность путей $P(v)$ для всех хостовv из $V \setminus U$ образует лес деревьев, ориентированных к своим корням, которыми являются хосты из U. Поэтому это множество путей замкнуто по дугам и не содержит циклических и дублирующих путей.

Оценим сложность алгоритма. Каждое ребро графа G просматривается ровно 2 раза (с обоих его концов), что суммарно даёт m просмотров. Поскольку в связном графе число вершин не превышает $m + 1$, сложность алгоритма $O(m)$. □

5. Задача 2 (настройка коммутаторов) в случае B (с клонированием)

Пусть выбрано некоторое решение задачи 1 в случае B, т.е. выбрана функция $f: X \to 2^H$, которая удовлетворяет требованию $\forall x \in X \; \exists! \, h \in f(x) \; x \in h.f: X \to H$ и имеет минимальную или близкую к минимальной мощность образа $|f(X)|$), отображающая объект x в множество хостов $f(x)$, и установлено взаимно-однозначное отображение d образа $f(X)$ во множество идентификаторов пакетов. Пакет, направляемый объекту x, будет иметь идентификатор $d(f(x))$.

Как и в случае A мы будем решать задачу 2, отвлекаясь от функции f. Однако в случае A пакет с идентификатором $d(f(x))$ должен был проходить один и только один полный путь до ближайшего хоста из множества U хостов. В случае B мы имеем не множество, а семейство U множеств хостов, причём хосты одного множества из семейства U реализуют одно и то же множество объектов, а каждый объект x реализован в каждом хосте одного и только одного множества из семейства U. Пакет за счёт клонирования должен проходить множество полных путей так, что множество их конечных хостов является выборкой из множеств семейства U по одному и только одному хосту из каждого множества. Когда пакет, предназначенный объекту x, попадает в хост из такого множества, происходит вызов объекта x. Если же этот пакет попадает в хост из другого множества семейства U, он игнорируется, поскольку в этих хостах он не реализован.

Для заданного идентификатора d и соответствующего ему непустого семейства U непустых множеств хостов нужно так настроить коммутаторы плоскости данных SDN, т.е. определить такие правила коммутаторов для данного идентификатора d, чтобы они порождали замкнутое по дугам множество путей $P(d)$ без зацикливания и дублирования такое, что для каждого хоста h и каждого множества U_i семейства U подмножество путей, начинающихся в хосте h, содержало один и только один полный путь, заканчивающийся в некотором хосте из

множества U_i. Для этого предлагается приведённый ниже алгоритм 2, который строит кратчайшие пути.

Алгоритм 2 является модификацией алгоритма 1, только для каждой вершины v вместо одной пометки $r(v)$ и одной ссылки на следующую вершину $p(v)$ пути используются соответствующие вектора размерности $|U|$. Вершина попадает в следующий фронт волны каждый раз, когда хотя бы в одном разряде вектора пометок $r(v)$ пометка меняется с ***false*** на ***true***. Из-за этого вершина может оказываться в нескольких (максимально $|U|$) фронтах волны.

Алгоритм_2(V, N, R, d, U) /* Настройка коммутаторов для кратчайшего доступа (без зацикливания и дублирования) от каждого хоста до одного хоста из каждого множества хостов в семействе множеств хостов U */

***Input*:** множество V вершин графа физических связей, функция N, задающая для каждой вершины v множество $N(v)$ её соседей, текущая настройка коммутаторов R, задающая для каждого коммутатора v текущее множество правил вида (d', a, s, b), где $d' \neq d$, новый идентификатор пакетов d, непустое семейство U непустых множеств хостов, соответствующее d, $k = |U|$.

***Output*:** новая настройка коммутаторов R.

$r(v) = \{ r(v)(1), ..., r(v)(k) \}$ – вектор пометок (***true***/***false***) вершины v,

$p(v) = \{ p(v)(1), ..., p(v)(k) \}$ – вектор соседей вершины v на путях к ближайшим хостам из U,

F – фронт волны (множество вершин) от хостов из U до других хостов,

F_{next} – множество F на следующем шаге.

$F = \varnothing$; $F_{next} = \varnothing$;

for all $v \in V$ ***do***

for all $i \in \{ 1, ..., k \}$ ***do***

$r(v)(i)$ = ***false***; /* во всех вершинах нет пометок */

for all $i \in \{ 1, ..., k \}$ ***do*** /* разряды i*/

for all $v \in U(i)$ ***do*** /* вершины в $U(i)$*/

$r(v)(i)$ = ***true***; $F = F \cup \{ v \}$; /* в хостах из $U(i)$ пометки в разряде i */

while $F \neq \varnothing$ ***do*** /* пока фронт волны не пуст */

for all $v \in F$ ***do*** /* вершины v из фронта волны */

for all $w \in N(v)$ ***do*** /* соседи w вершины v */

for all $i \in \{ 1, ..., k \}$ ***do*** /* разряды i*/

if $r(v)(i) \setminus r(w)(i)$ ***then*** /* в разряде i в v пометка, а в w нет */

$p(w)(i)$ = **true**; /* теперь сосед w помечен в разряде i */

if $|N(v)| > 1$ ***then*** /* v коммутатор */

$R(v) = R(v) \cup \{ (d, w, v, p(v)(i)) \}$; /* правило коммутатора v */

if $|N(w)| > 1$ ***then*** /* w коммутатор */

$F_{next} = F_{next} \cup \{ w \}$; /* помещаем w в следующий фронт */

return R;

Теорема 6. Алгоритм 2 правильно настраивает коммутаторы для решения задачи 2 в случае B и имеет сложность $O(km)$, где k мощность семейства U, а m число ребер графа G физических связей.

Доказательство. Расстояние от вершины v до ближайшего хоста из множества $U(i)$, $i = 1, ..., k$, обозначим через $d(v)(i)$. Будем говорить, что расстояние $d(v)(i)$ известно, если $r(v)(i)$ = ***true***. Обозначим последовательность вершин $P(v, i) = (v_0, v_1, ..., v_{d(v)(i)})$, где $v_0 = v$ и $v_l = p(v_{l-1})$ для $l > 0$.

Сначала докажем, что в начале каждого j-го шага $j = 0, 1, ...$, т.е. j-й итерации цикла ***while***, начиная с 0, 1) все расстояния $d(v)(i) \le j$ и только они известны, 2) последовательность $P(v, i)$ для известного расстояния $d(v)(i)$ определена и есть последовательность вершин кратчайшего пути длиной $d(v)(i)$ от вершины v до ближайшего хоста из$U(i)$, 3) для всех известных расстояний $d(v)(i) = j$ вершина v входит во фронт волны, если $j = 0$ или $j > 0$ и v коммутатор.

Действительно, в начале шага $j = 0$ 1) для $i = 1, ..., k$ хост $v \in U(i)$ имеет расстояние $d(v)(i) = 0$ (это расстояние от хоста v до самого себя), и это расстояние известно, поскольку пометка $r(v)(i)$ = ***true***, а в остальных случаях, когда $v \notin U(i)$, расстояние $d(v)(i) > 0$, и оно неизвестно, поскольку пометка $r(v)(i)$ = ***false***. 2) Для известного расстояния $d(v)(i) = 0$ последовательность $P(v, i) = (v)$ есть последовательность вершин на кратчайшем пути длиной $d(v)(i) = 0$ от вершины v до ближайшего хоста v из $U(i)$, т.е. до самого себя. 3) $j = 0$ и каждая вершина $v \in U(i)$ входит во фронт волны $F = \cup U$.

Пусть утверждение верно в начале j-ого шага и докажем, что оно остаётся верным в начале следующего $(j + 1)$-го шага. 1) Пусть вершина w для некоторого $i = 1, ..., k$ имеет расстояние $d(w)(i) = j + 1$. По предположению шага индукции все расстояния $d(v)(i) \le j$ известны. Следовательно, найдётся такая вершина v, для которой $d(v)(i) = j$ и которая является соседом вершины w. По предположению шага индукции вершина v входит во фронт волны. Поскольку на каждом шаге алгоритм просматривает всех соседей вершин из фронта волны, будет просмотрена вершина w и установлена пометка $r(w)(i)$ = ***true***, тем самым расстояние $d(w)(i) = j + 1$ станет известным к началу следующего $(j + 1)$-го шага. Если же вершина t для некоторого $i = 1, ..., k$ имеет расстояние $d(t)(i) > j + 1$, то, очевидно, её соседом не может быть вершина v, для которой $d(v)(i) = j$, поэтому на следующем шаге останется $r(t)(i)$ = ***false***, т.е. расстояние $d(t)(i)$ останется неизвестным. 2) В вершине w с расстоянием $d(w)(i) = j + 1$ также будет установлено $p(w)(i) = v$. Поскольку по предположению шага индукции последовательность $P(v, i)$ определена и есть последовательность вершин кратчайшего пути длиной $d(v)(i)$ от вершины v до ближайшего хоста из $U(i)$, последовательность $P(w) = (w) \cdot P(v)$ также определена и есть последовательность вершин кратчайшего пути длиной $d(w)(i)$ от вершины w до ближайшего хоста из $U(i)$. 3) $j + 1 > 0$ и все вершины w с расстоянием $d(w)(i) = j + 1$ для $i = 1, ..., k$, которые являются коммутаторами, и только они войдут в следующий фронт волны.

Поскольку граф связный, при окончании алгоритма все расстояния $d(v)(i)$ станут известными и для каждого хоста v последовательность $P(v, i)$ определена и есть последовательность вершин кратчайшего пути от хоста v до ближайшего хоста из $U(i)$. Поскольку на каждом шаге

алгоритм устанавливает правило $(d, w, v, p(v)(i))$ для каждого коммутатора v, каждой вершины w и каждого разряда i при условии $p(w)(i) = v$, совокупность установленных правил обеспечивает прохождение пакетов с идентификатором d из каждого хоста h путей $P(h, i)$, $i = 1, ..., k$, которые являются кратчайшими путями из h до ближайших хостов из множеств $U(i)$. Пути, проходящие дугу (w, v), в вершине v разветвляются (пакеты клонируются) на несколько ветвей по дугам $(v, p(v)(i))$, $i = 1, ..., k$ (число дуг может быть меньше k, так как дуги могут совпадать для некоторых i), через которые каждый путь $P(v, i)$ ведёт в ближайший хост из $U(i)$. Тем самым, множество полных путей $P(h, i)$, $i = 1, ..., k$, по всем хостам h, замкнуто по дугам и не содержит циклов и дублирующих путей.

Оценим сложность алгоритма. Каждое ребро графа G просматривается ровно $2k$ раз: с обоих его концов и для каждого разряда $i = 1, ..., k$, что суммарно даёт $2km$ просмотров. Поскольку в связном графе число вершин не превышает $m + 1$, сложность алгоритма $O(km)$.

5. Заключение

В статье рассматривается выполнение на плоскости данных SDN, моделируемой конечным связным неориентированным графом физических связей, программы задания, которая понимается в духе парадигмы объектно-ориентированного программирования как состоящая из объектов и сообщений, которыми объекты могут обмениваться. Объекты реализуется в хостах, причём в одном хосте может быть реализовано несколько разных объектов, а один и тот же объект может быть реализован в нескольких хостах. Сообщения между объектами, реализованными в разных хостах, помещаются в пакеты, маршрутизацию которых исполняют коммутаторы на основе идентификаторов, присвоенных пакетам и помещаемых в заголовки пакетов как набор значений некоторых параметров пакетов. В работе решаются две задачи: 1) минимизация числа идентификаторов и 2) настройка коммутаторов для реализации путей, которые должны проходить коммутаторы. Эти задачи решаются в двух случаях: A) пакет, предназначенный для некоторого объекта, должен попасть ровно в один хост, в котором реализован этот объект, B) пакет может попадать в несколько хостов, но в одном и только одном из них должен быть реализован нужный объект.

Показано, что задача 1 в случае A эквивалентна задаче о покрытии множества, а минимальное число идентификаторов в наихудшем случае равно $min\{n, m\}$, где n число объектов, а m число хостов, реализующих объекты. В случае B задача является специальной модификацией задачи о покрытии множества (задачей точного покрытия набором подмножеств [4]). Доказано, что минимальное число идентификаторов в наихудшем случае не превышает m, и для каждых n и m имеется пример, когда это число равно $min\{\lfloor lb(n+1) \rfloor, m\}$. Предложена гипотеза о том, что $min\{\lfloor lb(n+1) \rfloor, m\}$ является также верхней оценкой. В настоящее время эта гипотеза не доказана и не опровергнута. В [4]. доказана верхняя оценка $O(min\{ ln(min\{n, m\}) \cdot ln(n, m)\})$.

Для решения задачи 2 в случаях A и B предложены алгоритмы настройки коммутаторов сложности, соответственно, $O(m)$ и $O(km)$, где m число рёбер графа физических связей, а k результат решения задачи 1 в случае B как число требуемых идентификаторов пакетов.

В дальнейших исследованиях можно попытаться учесть вопросы надёжности, безопасности и оптимизации нагрузки на хосты и коммутаторы, которые игнорируются в данной работе. В частности, можно рассматривать взвешенный граф физических связей и налагать ограничения на маршрутизацию пакетов.

Список литературы / References

[1]. Н.Н. Кузюрин. Обобщенные покрытия и их аппроксимации. Труды ИСП РАН, том 6, 2004 г., стр. 85-100 / N.N. Kuzyurin. Generalized covers and their approximations. Trudy ISP RAN/Proc. ISP RAS, vol. 6, 2004, pp. 85-100 (in Russian).

[2]. Н.Н. Кузюрин. О сложности построения асимптотически оптимальных покрытий и упаковок. Доклады Академии наук, том 363, no. 1., 1998 г., стр. 11-13 / N. N. Kuzyurin, On the complexity of asymptotically optimal coverings and packing. Doklady Mathematics, vol. 58, no. 3, 1998, pp, 345-346.

[3]. А.В. Еремеев, Л.А. Заозерская, А.А. Колоколов. Задача о покрытии множества: сложность, алгоритмы, экспериментальные исследования, Дискретный анализ и исследование операций, том 7, вып. 2, 2000 г., стр. 22-46 / A.V. Eremeev, L.A. Zaozerskaya, A.A. Kolokolov. The set covering problem: complexity, algorithms, and experimental investigations. Discrete Analysis and Operations Research, vol. 7, issue 2, 2000, pp. 22-46 (in Russian).

[4]. D. Lazarev. On MAX Monotonous XSAT, Exact Cover by Sets of Subsets and Parallel Computations in Software-Defined Network. Moscow Journal of Combinatorics and Number Theory, 2023 (in print).

[5]. И.Б. Бурдонов, Н.В. Евтушенко, А.С. Косачев. Тестирование правил настройки сетевого коммутатора программно конфигурируемой сети. Труды ИСП РАН, том 30, вып. 6, 2018 г., стр. 69-88. DOI: 10.15514/ISPRAS-2018-30(6)-4 / I.B. Burdonov, N.V. Yevtushenko, A.S. Kossatchev. Testing switch rules in software defined networks. Trudy ISP RAN/Proc. ISP RAS, vol. 30, issue 6, 2018, pp. 69-88 (in Russian).

[6]. I. Burdonov, A. Kossachev et al. Preventive Model-based Verification and Repairing for SDN Requests. In Proc. of the 16th International Conference on Evaluation of Novel Approaches to Software Engineering 2021, pp. 421-428.

Информация об авторах / Information about authors

Игорь Борисович БУРДОНОВ – доктор физико-математических наук, г.н.с. ИСП РАН. Научные интересы: формальные спецификации, генерация тестов, технологиякомпиляции, системы реального времени, операционные системы, объектно-ориентированное программирование, сетевые протоколы, процессы разработкипрограммного обеспечения.

Igor Borisovich BURDONOV – Doctor of Physical and Mathematical Sciences, a Leading Researcher of ISP RAS. Research interests: formal specifications, test generation, compilation technology, real-time systems, operating systems, object-oriented programming, network protocols, softwaredevelopment processes.

Нина Владимировна ЕВТУШЕНКО, доктор технических наук, профессор, г.н.с. ИСП РАН, до 1991 года работала научным сотрудником в Сибирском физико-техническом институте. С 1991 г. работала в ТГУ профессором, зав. кафедрой, зав. лабораторией по компьютерным наукам. Её исследовательские интересы включают формальные методы, теорию автоматов, распределенные системы, протоколы и тестирование программного обеспечения.

Nina Vladimirovna YEVTUSHENKO, Doctor of Technical Sciences, Professor, a Leading Researcher of ISP RAS, worked at the Siberian Scientific Institute of Physics and Technology as a researcher up to 1991. In 1991, she joined Tomsk State University as a professor and then worked as the chair head and the head of Computer Science laboratory. Her research interests include formal methods, automata theory, distributed systems, protocol and software testing.

Александр Сергеевич КОСАЧЕВ – кандидат физико-математических наук, ведущий научный сотрудник. Научные интересы: формальные спецификации, генерация тестов, технология компиляции, системы реального времени, операционные системы, объектно-ориентированное программирование, сетевые протоколы, процессы разработки программного обеспечения.

Alexander Sergeevitch KOSSATCHEV – Candidate of Physical and Mathematical Sciences, Leading Researcher. Research interests: formal specifications, test generation, compilation technology, real-time systems, operating systems, object-oriented programming, network protocols, software development processes.

DOI: 10.15514/ISPRAS-2022-34(3)-12

Организация безопасного запроса к базе данных на облаке

[1] *С.А. Мартишин, ORCID: 0000-0001-5437-4049 <mart@ispras.ru>*
[1] *М.В. Храпченко, ORCID: 0000-0002-5147-5132 <khrap@ispras.ru>*
[1,2] *А.В. Шокуров, ORCID: 0000-0002-6801-7728 <shok@ispras.ru>*

[1] *Институт системного программирования им. В.П. Иванникова РАН, 109004, Россия, г. Москва, ул. А. Солженицына, д. 25*
[2] *Московский физико-технический институт, 141701, Россия, Долгопрудный, Институтский пер., 9*

Аннотация. Развитие облачных вычислений, включающее хранение и обработку конфиденциальных данных пользователей на серверах, которые могут быть атакованы, выдвигает новые требования к защите информации. В статье исследуется задача получения клиентом информации из базы данных таким образом, чтобы никто, кроме самого клиента не обладал сведениями о том, какую именно информацию он запросил (задача PIR – Private Information Retrieval). Эта задача в информационно-теоретической постановке была сформулирована в 1995 году Шором, Голдрайхом, Кушелевицем и Суданом. Предложена модель облачных вычислений, включающая облако, пользователя, клиентов, доверенное лицо (дилера), пассивного противника на облаке. Также предполагается, что у атакующей стороны имеется возможность создания фальшивых клиентов для формирования неограниченного числа запросов. Предложен алгоритм формирования и размещения базы данных на облаке и алгоритм запроса требуемого бита. Приведены оценки коммуникационной сложности и вероятности угадывания номера бита при совершении любого количества запросов клиентами, созданными для организации атаки.

Ключевые слова: базы данных; облачные вычисления; PIR

Для цитирования: Мартишин С.А., Храпченко М.В., Шокуров А.В. Организация безопасного запроса к базе данных на облаке. Труды ИСП РАН, том 34, вып. 3, 2022 г., стр. 173-188. DOI: 10.15514/ISPRAS–2022–34(3)–12.

Благодарности: Работа выполнена при поддержке Министерства науки и высшего образования Российской Федерации (грант 075-15-2020) и ИСП РАН. Авторы выражают особую благодарность Лазареву Д.О. за ценные замечания в процессе работы над статьей.

Organization of a secure query to a database in the cloud

[1] *S.A. Martishin, ORCID: 0000-0001-5437-4049<mart@ispras.ru>*
[1] *M.V. Khrapchenko, ORCID: 0000-0002-5147-5132 <khrap@ispras.ru>*
[2] *A.V. Shokurov, ORCID: 0000-0002-6801-7728 <shok@ispras.ru>*

[1] *Ivannikov Institute for System Programming of the Russian Academy of Sciences, 25, Alexander Solzhenitsyn st., Moscow, 109004, Russia*
[2] *Moscow Institute of Physics and Technology, 9 Institutskiy per., Dolgoprudny, Moscow Region, 141701*

Abstract. The development of cloud computing, including the storage and processing of confidential user data on servers that can be attacked, puts forward new requirements for information protection. The article explores the problem of obtaining information from the database by the client in such a way that no one except the client

himself get any information about the information the client is interested in (PIR - Private Information Retrieval). The problem was introduced in 1995 by Chor, Goldreich, Kushilevitz and Sudan in the information-theoretic setting. A model of cloud computing is proposed. It includes a cloud, a user, clients, a trusted dealer, a passive adversary in the cloud. Also, the attacking side has the ability to create fake clients to generate an unlimited number of requests. An algorithm for the organization and database distribution on the cloud and an algorithm for obtaining the required bit were proposed. Communication complexity of the algorithm was estimated. The probability of revealing required bit's number in the case when fake clients perform unlimited requests was estimated too.

Keywords: database; cloud computing; PIR

For citation: Martishin S.A., Khrapchenko M.V., Shokurov A.V. Organization of a secure query to a database in the cloud. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 3, 2022, pp. 173-188 (in Russian). DOI: 10.15514/ISPRAS–2022–34(3)–12

Acknowledgements. The work was supported by the Ministry of Science and Higher Education of the Russian Federation (grant 075-15-2020) and ISP RAS. The authors are especially grateful to D.O. Lazarev for valuable comments during the work on the article.

1. Введение

Безопасное использование облачных вычислений невозможно без решения задачи защиты данных. Хорошо известна задача получения информации из базы данных с открытым доступом, размещенной на облаке, таким образом, чтобы никто, кроме клиента, обращающегося с запросом, не обладал сведениями о запрашиваемой информации.

Неформально задача может быть сформулирована следующим образом. Алиса делает запрос к базе данных, но хочет, чтобы никто, в том числе любопытный администратор базы данных, не узнал, какие именно данные запрашиваются. Очевидным решением является запрос Алисой всей базы данных. Это решение приводит к чрезмерным затратам на передачу информации. Возникает вопрос: можно ли получить ответ с меньшей коммуникационной сложностью?

Протокол поиска конфиденциальной информации (Private information retrieval – PIR) позволяет пользователю получить интересующую его частную информацию с сервера. В области исследования PIR протокола значительный вклад был сделан в [1,2].

Сформулированная в этих работах постановка задачи приведена ниже:

Имеется база данных – бинарная строка $X = (x_1, \dots, x_n)$ длины n, хранящаяся на сервере.

Клиент хочет получить один бит информации x_iиз базы данных X так, чтобы сервер не смог определить, с какой позиции i был запрошен бит.

В первую очередь, протокол PIR должен удовлетворять **условию корректности**: для любого номера $i, 1 \le i \le n,$ клиент может сформировать такой запрос, что после завершения выполнения протокола клиент получит данные, по которым он может правильно вычислить значение бита x_i.

Помимо условия корректности, протокол PIR должен удовлетворять **условию конфиденциальности**: в результате выполнения протокола PIR противник не получает никакой информации о номере i запрашиваемого бита x_i из базы данных.

Если база данных размещена на единственном сервере, который полностью контролируется противником, то теоретико-информационное условие конфиденциальности корректного протокола PIR может быть выполнено только том случае, когда клиент запрашивает базу данных целиком. Это означает, что по каналам связи придется передавать объем информации, не меньший того, который содержится в самой базе данных. Очевидно, что такой протокол практически неприемлем из-за больших коммуникационных издержек [2].

Под коммуникационной сложностью протокола понимают общее количество бит, которыми обмениваются участники протокола за время его работы. Чтобы PIR-протокол можно было

бы применять на практике, он должен удовлетворять **условию лаконичности**: коммуникационная сложность протокола должна быть существенно меньше размера базы данных.

Таким образом, если база данных размещена только на одном сервере, и этот сервер контролируется противником, то не существует PIR-протокола, который одновременно удовлетворял бы условиям корректности, конфиденциальности и лаконичности. Возникает вопрос: можно ли построить протоколы PIR, удовлетворяющие всем трем указанным условиям одновременно?

Одной из таких перспективных моделей является модель реплицированной базы данных, в которой несколько одинаковых копий строки $X = (x_1, \dots, x_n)$ размещены на разных серверах. Предполагается, что клиент имеет связь с каждым из этих серверов, но сами серверы не имеют связи друг с другом. Также предполагается, что противник может наблюдать любой из серверов, на которых размещена база данных, причем, возможно, разные серверы во время выполнения разных сеансов протокола.

В работах [1, 2] было доказано, что для моделей с k реплицированными копиями базы данных при $k \geq 2$ возможно построение корректных конфиденциальных лаконичных протоколов PIR. Дальнейшие исследования в этом направлении были сосредоточены на изучении методов улучшения коммуникационной сложности [3, 4]. Кроме того, в работах [5-8] были исследованы возможности построения и оценки коммуникационной сложности (или лаконичности) схемы PIR на одном сервере при помощи различных техник, в том числе гомоморфного шифрования.

Однако, если рассматривать задачу построения PIR в предположении о том, что реплицированная база данных хранится на облаке, то очевидно, что ни владелец данных (пользователь), ни клиент (имеющий официальный доступ для получения информации из базы данных) не могут контролировать облако. Таким образом, в облачных информационных системах в отличие от [1, 2] нельзя обеспечить изолированность копий распределенной базы данных друг от друга, то есть нельзя исключить обмен информацией между копиями базы данных. Кроме того, говоря о хранении и использовании конфиденциальной информации, требуется знать, является ли противник активным или пассивным.

Активный противник в облаке, который способен вмешиваться в ход выполнения протокола, как правило, может быть выявлен достаточно быстро путем полного анализа результатов выполнения протокола [9].

Пассивный противник, который наделен способностью получать некоторую информацию о процессе выполнения протокола, но не в праве в него вмешиваться, не может быть обнаружен даже при исчерпывающем анализе результатов многократного выполнения протокола. Такой противник представляет даже большую опасность, поскольку он собирает и анализирует конфиденциальную информацию, не обнаруживая себя.

Для облачных вычислений, исследовались различные возможности реализации. Например, реализация простой схемы PIR с несколькими серверами, основанной на подходе, аналогичном подходу построения систем RAID (Redundant Array of Inexpensive Disks) с избыточными массивами недорогих дисков [10]. В этом случае каждый сервер хранит только часть базы данных, серверы могут быть опрошены параллельно. Предполагается, что не все они вступают в сговор, поскольку эти серверы могут работать на разных платформах или под управлением различных облачных провайдеров.

Также исследовались различные методы, делающие практическое применение PIR (CPIR - computational PIR) эффективнее. Например, в [11] рассматривалось два метода. Первый направлен на эффективное использование центрального процессора сервера. Второй метод основан на кодировании данных PBC (probabilistic batch codes – вероятностные пакетные коды) и используется для создания схемы PIR, предназначенной для обработки множества запросов. На основе этих методов была разработана библиотека CPIR – SealPIR, которая

объединяет наиболее эффективный в вычислительном отношении протокол CPIR и предлагаемый метод сжатия запросов, что уменьшает коммуникационные затраты.

Из практических реализаций также стоит отметить MuchPIR — поиск частной информации с использованием гомоморфного шифрования, реализованный как расширение C/C++ Aggregate для Postgres [12].

Таким образом, за все время изучения протокола PIR было получено большое количество результатов для различных условий его применения.

Однако, если базы данных хранятся на облачных серверах, то условия задачи PIR существенно изменяются, и ранее известные методы ее решения не подходят. Чтобы найти решение задачи PIR в облачной вычислительной среде, предлагается рассмотреть иную модель взаимодействия участников протокола с базой данных.

2. Состав модели и постановка задачи

2.1 Состав модели

Модель вычислений, в рамках которой мы исследуем облачный вариант задачи PIR, включает несколько участвующих в ней процессов.

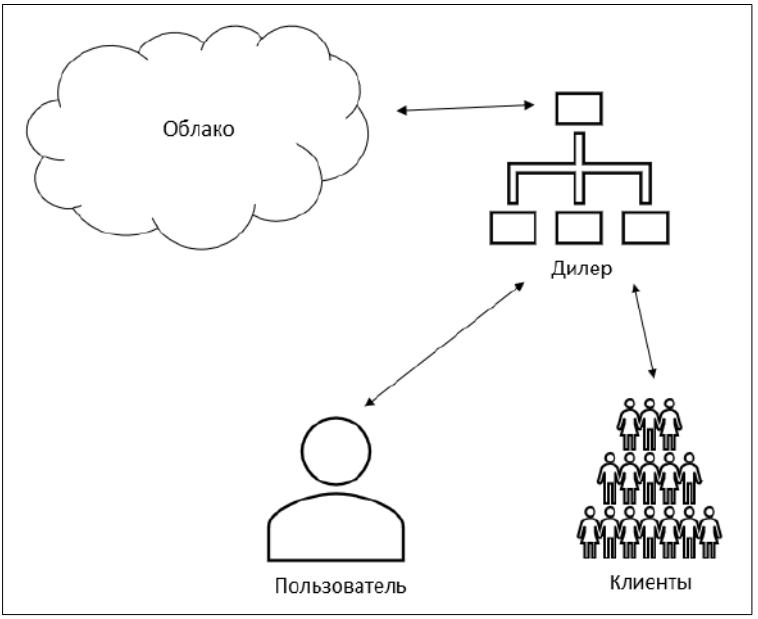


Рис. 1. Состав модели
Fig. 1. Model components

1) **Облако.** Состоит из нескольких серверов, на каждом из которых хранится копия одной и той же базы данных. Эти копии могут отличаться друг от друга при использовании различных ключей шифрования одинаковых данных. Облачные серверы способны выполнять любые вычислительные операции, присущие системам управления базами данных. Считается, что эти серверы соединены посредством незащищенных каналов связи друг с другом и дилером. По этим каналам облачные серверы обмениваются по запросу информацией (получают и передают) с дилером. Облачные серверы и все примыкающие к ним каналы связи доступны для стороннего наблюдателя (противника).
2) **Пользователь.** Хранит данные на облаке. Предполагается, что на облаке хранится k копий баз данных. Для загрузки данных пользователь обращается к дилеру, с которым соединен защищенным каналом связи. Этот канал недоступен для наблюдения.

3) **Клиенты.** Запрашивают некоторую информацию из базы данных. Обращаются для выполнения запроса к дилеру. С дилером соединены защищенными каналами связи.
4) **Дилер.** Находится вне облака, противнику недоступен. Дилер имеет защищенные каналы связи с пользователем и клиентами. Канал связи с облаком является незащищенным. Объем памяти дилера для постоянного хранения данных пренебрежимо мал по сравнению с n. Функции дилера:
   - аутентифицирует пользователя и клиентов;
   - получает от пользователя данные для размещения на облаке;
   - генерирует ключи шифрования/расшифрования;
   - осуществляет обмен данными с облаком в процессе размещения данных на облаке и в процессе обработки запроса;
   - выполняет шифрование/расшифрование;
   - производит сортировку шифротекстов.

Модель рассматривается в предположении, что на облаке имеется **противник**, который не вмешивается в выполнение криптографического протокола. Имеет доступ к базе данных на облаке, к каждой ее копии, к каналам связи на облаке. Может создавать фальшивых клиентов, работающих по протоколу. Таким образом, запрос к одной из копий баз данных сравнивается с запросом к другой копии базы данных и собирается статистическая информация. Противник не имеет доступа к дилеру.

Под **угрозой** будем понимать угадывание исходного номера бита, запрашиваемого клиентом в базе данных.

Атака включает в себя:

1) Сбор информации противником в облаке (наблюдение, сбор статистики и анализ).
2) Фальшивые клиенты. Такие клиенты с помощью следующей процедуры могут сформировать произвольное число запросов, что позволит в коалиции с облаком собрать и проанализировать информацию для всей базы данных.

2.2 Постановка задачи и условные обозначения

Имеется база данных – бинарная строка $X = (x_1, \dots, x_n)$ длины n.

Клиент хочет получить один бит информации x_iс номером i из базы данных X так, чтобы противник не узнал ничего о том, с какой позиции i был запрошен бит.

Будем размещать на облаке k копий баз данных ($k \geq 2$). Пусть $k = 2^d$,где $d \geq 1$. Без потери общности предполагается, что $n = l^d$, то есть, $d = log_2 k$ и $l = \sqrt[d]{n}$. Пусть $L_p = \frac{n}{l}$.

Так как $l = \sqrt[d]{n}$, то $L_p \geq l$ или $\frac{n}{l} \geq l$, что эквивалентно $n \geq l^2$. Учитывая, что $n = l^d$, отсюда следует, что $d \geq 2$ и $k \geq 4$.

Пусть x – элемент группы Z_2, а K, K_{enc}, K_{dec} – ключи шифрования и расшифрования, alg - алгоритм шифрования или расшифрования, $h \in \{1, \dots, k\}$ – номер соответствующей копии базы данных. $DEnc(x, K, alg, h)$ - функция однозначного шифрования. $REnc(x, K_{enc}, alg, h)$ и $RDec(x, K_{dec}, alg, h)$ – функции вероятностного шифрования и расшифрования.

3. Размещение базы данных на облаке

3.1 Предварительные замечания

Для простоты изложения будем считать, что база данных загружается одним пользователем. Предполагается, что после загрузки база данных не может быть изменена. Также будем рассматривать случай запроса одного бита.

На начальном этапе пользователь производит загрузку базы данных. Он обращается к дилеру. Дилер проводит аутентификацию пользователя. Если полномочия пользователя подтверждены, то пользователь может передавать информацию дилеру.

Предполагается, что каналы связи с дилером защищены, поэтому пользователь передает данные в незашифрованном виде.

Построим матрицу *M* (размера $n \times d$)

$$M = \begin{pmatrix} m_{11}m_{12} & \cdots & m_{1d} \\ \vdots & \ddots & \vdots \\ m_{n1}m_{n2} & \cdots & m_{nd} \end{pmatrix}$$

с помощью следующей процедуры. Строка с номером i матрицы *M* имеет вид: в первом столбце m_{i1} стоит номер i, в остальных $d-1$ столбцах – различные натуральные числа из отрезка [1,n], которые сгенерированы при помощи датчика псевдослучайных чисел PRNG(*db,i*) и не равные i.

Для инициализации датчика случайных чисел задаются два параметра. Первый параметр *db* соответствует конкретной базе данных $X = (x_1, \dots, x_n)$ и хранится все время существования этой базы данных. Второй соответствует запрашиваемому номеру бита i. Эти параметры обеспечивают повторяемость датчика псевдослучайных чисел.

Строка матрицы *M* должна содержать различные числа. Если при генерации псевдослучайных чисел для строки матрицы *M* получаем число, которое было сгенерировано ранее, датчик *PRNG(db,i)* запускается до тех пор, пока не получим число, которое не было сгенерировано ранее. Таким образом, строка матрицы *M* будет содержать d различных чисел. Разные строки могут содержать одинаковые числа. Помимо этого, требуется, чтобы датчик псевдослучайных чисел *PRNG(db,i)* позволял бы восстановить любую строку матрицы *M*. Тогда дилеру нет необходимости хранить матрицу *M* в памяти.

Формирование матрицы *M* происходит следующим образом.

Построим l интервалов, в которые попадут все n элементов:

$[1, L_p], [L_p + 1{,}2 \cdot L_p], \dots, [(l-1) \cdot L_p + 1, l \cdot L_p]$.

Число элементов в каждом интервале $L_p = \frac{n}{l}$. Рассмотрим, например, первый интервал и первую строку, то есть $i = 1$. Датчик псевдослучайных чисел *PRNG(db,i)* генерирует случайные числа в диапазоне $[2, L_p]$. Числа в строке различны, всего получаем *d* чисел. Она состоит из элементов, $m_{11}, m_{12}, .., m_{1d}$ ($m_{11} = 1$, значения всех этих элементов попадают в диапазон $[1, L_p]$.

Далее для каждой строки из первого интервала вычисляется $l-1$ строк, принадлежащих другим интервалам. На основе каждой строки из первого интервала генерируется по одной строке для каждого из оставшихся интервалов. Эти строки заполняются значениями по алгоритму, представленному ниже.

Далее осуществляется переход к следующей строке первого интервала и алгоритм повторяется. То есть для строк с номерами $i \in \{1, \dots, L_p\}$ формируем соответственно по l строк для каждой строки *i*.

Сформируем вектор-столбец $\boldsymbol{b}$, состоящий из различных случайных натуральных чисел b_m значения которых лежат в диапазоне $[1, L_p], m \in \{1, \dots, l\}$. Числа b_m генерируются датчиком псевдослучайных чисел. На вход датчика *PRNG2(db,m)* подается номер интервала m, для которого вычисляется b_m. Вектор-столбец $\boldsymbol{b}$ восстанавливается каждый раз одинаково. Хранить вектор-столбец $\boldsymbol{b}$ нет необходимости, поскольку при одинаковом начальном значении параметра *m* датчик срабатывает одинаково.

3.2 Алгоритм построения матрицы *M*

На вход алгоритма подается: L_p, l, d.

```
for i ← 1 to L_p do
/* при помощи датчика псевдослучайных чисел PRNG(db,i)
   генерируется i-я строка первого интервала */
     for m ← 2 to l do
          z ← 0 /* инициализация глобальной переменной */
          for j← 1 to d do
               /* при помощи датчика псевдослучайных чисел
                  PRNG2(db,1) генерируется b_1*/
               a ← (m_ij − b_1)modL_p
                /* при помощи датчика псевдослучайных чисел
                  (db,m) генерируется b_m */
               c ← (a + b_m)modL_p
               if  c = 0  then
                    c ← L_p
               y ← L_p · (m − 1) + c
               if  j = 1  then
                    z ← y
               m_zj = y
```

В цикле по *i* последовательно генерируются L_p строк, где первым элементом является номер строки i, а остальные $d-1$ элементов отличны от *i* и различны между собой.

Для каждой из этих строк по приведенному выше алгоритму в цикле по *m* генерируются *l*-1 строк для интервалов с номерами от 2 до *l*. Каждая из формируемых *l*-1 строк попадает в свой интервал. В каждый интервал попадает ровно одна строка.

Все элементы *i*-й генерируемой строки различны. Если датчик псевдослучайных чисел *PRNG(db,i)* генерирует число, которое встречалось ранее в этой строке, то генерируется следующий элемент до тех пор, пока не получим элемент, не совпадающий с предыдущими.

Таким образом, на *i*-м шаге ($i \in [1, L_p]$) алгоритма вычисляется l строк, причем первой формируется *i*-я строка матрицы *M*. Остальные *l*-1 строк попадают случайным образом в различные интервалы (по одной строке в интервал), в зависимости от значения вектора-столбца $\boldsymbol{b}$. Вектор-столбец $\boldsymbol{b}$ позволяет вычислять номера строк в каждом интервале не по упрощенной формуле $y = L_p \cdot (m-1)$, что позволило бы противнику легко определить номера строк в каждом интервале. Благодаря вектору-столбцу $\boldsymbol{b}$ номер строки в каждом интервале определяется случайным образом.

Поскольку матрица *M* требует для хранения значительного объема памяти, дилер не хранит матрицу *M*. Элементы матрицы *M*, которые требуются для вычисления, формируются при помощи датчика псевдослучайных чисел *PRNG(db,i)*. Датчик псевдослучайных чисел при одинаковой инициализации генерирует одинаковую последовательность чисел и восстановление *i*-ой строки матрицы *M* возможно.

3.3 Построение матрицы *G*

Построим матрицу *G*, размера $n \times 2$

$$G = \begin{pmatrix} g_{11} g_{12} \\ \vdots \quad \vdots \\ g_{n1} g_{n2} \end{pmatrix}.$$

Первый столбец матрицы *G* генерируется как случайная последовательность из элементов 0 и 1.

Во второй столбец поместим корректирующие биты с помощью следующего алгоритма.

Каждая строка матрицы *M* состоит из *d* чисел. Каждое число можно рассматривать как номер, которому поставлен в соответствие случайным образом выбранный бит g_{i1}.

Положим:

$g_{i2} = \bigoplus_{j=1}^{d} g_{m_{ij}1} \oplus x_i$, где m_{ij} – элементы i-й строки матрицы M, а x_i – элемент бинарной строки $X = (x_1, \dots, x_n)$ длины n.

Тогда:

$x_i = \bigoplus_{j=1}^{d} g_{m_{ij}1} \oplus g_{i2}$.

Дилер осуществляет поэлементное шифрование матрицы G для отправки ее на облако. Шифротексты, полученные из элементов матрицы G хранятся на облаке. При хранении матрицы G на облаке для доступа к ее строкам будем использовать шифротекст, полученный из номера строки.

Шифротексты номеров строк матрицы G будем рассматривать как указатели на шифротексты ее строк.

Заметим, что поскольку в дальнейшем расшифровка шифротекста номера строки матрицы G не выполняется, в качестве алгоритма шифрования можно использовать одностороннюю функцию $DEnc(x, K_{enc}, alg, h)$, где K_{enc} является аргументом этой функции. Для возможности поиска по базе данных односторонняя функция должна быть однозначной, то есть без коллизий (различным открытым текстам соответствуют различные шифротексты).

Для шифрования матрицы G используется вероятностное шифрование. Дилер применяет вероятностное шифрование/расшифрование значения. Вероятностное шифрование позволяет для исходных значений 0 или 1 получить различные шифротексты в случае одинаковых исходных значений.

3.4 Размещение k копий базы данных на облаке

Для размещения k копий базы данных на облаке дилер генерирует:

1. k ключей $K(h)(h \in \{1, \dots, k\})$ для шифрования номера строки матрицы G (односторонняя функция) по одному для каждой копии базы данных;
2. k пар ключей $K_{enc}(h)$ и $K_{dec}(h)$, где $h \in \{1, \dots, k\}$, для вероятностного шифрования/расшифрования значений строк матрицы G, по одной паре для каждой копии базы данных.

Дилер шифрует номера строк матрицы G сгенерированными ключами $K(h)(h \in \{1, \dots, k\})$. Дилер шифрует матрицу G своим набором ключей для каждой копии базы данных.

Обозначим через $G_{enc}(h)(h \in \{1, \dots, k\})$, матрицу, где первый столбец – зашифрованные номера строк матрицы G, а второй и третий столбцы – зашифрованные столбцы матрицы G для k копий базы данных. Таким образом размер матрицы $G_{enc}(h)$ равен $n \times 3$.

Дилер осуществляет перестановку строк матриц $G_{enc}(h)$ $(h \in \{1, \dots, k\})$ путем сортировки их по 1 столбцу и передает матрицы $G_{enc}(h)$ облаку. Это необходимо, чтобы скрыть истинный порядок номеров для каждой копии базы данных. Поскольку для каждой копии базы данных используется свой ключ шифрования, порядок строк матриц будет различным.

Для выполнения шифрования и последующей перестановки дилеру требуется иметь объем памяти, достаточный для хранения одной базы данных. При этом предполагается, что дилер работает с несколькими базами данных и все эти базы данных одновременно дилер не хранит.

Так как матрицы $G_{enc}(h)$ хранятся на облаке, после загрузки базы данных на облако память дилера очищается для работы со следующей базой данных и в памяти дилера остаются только ключи.

В приведенном ниже алгоритме дилер должен выполнять Шаги 4-6 для каждой копии базы данных последовательно. Иначе ему придется хранить в памяти все k копий базы данных, что приводит к большим расходам памяти.

3.5 Описание алгоритма загрузки данных на облако

Шаг 1. Пользователь передает дилеру полученный массив номеров битов и значений, т.е. массив $X = (x_1, \dots, x_n)$. Дилер аутентифицирует пользователя и принимает данные.

Шаг 2. Дилер формирует матрицу $G = \|g_{ij}\|$, где $i = 1,..,n, j = 1,2$.

Шаг 3. Дилер генерирует ключи $K(h)$, $K_{enc}(h)$ и $K_{dec}(h)$, где $h \in \{1, \dots, k\}$.

Ключ $K(h)$ (реализующий одностороннюю функцию) предназначен для шифрования номера строки матрицы G.

Открытый $K_{enc}(h)$ и секретный $K_{dec}(h)$ ключи предназначены для вероятностного шифрования/расшифрования значений столбцов матрицы G.

Шаг 4. Дилер шифрует ключом $K(h)$ номер строки матрицы G $DEnc(g_{i1}, K(h), alg, h)$, где $i = 1,..,n,\ h \in \{1, \dots, k\}$ и получает шифротексты, являющиеся указателями на строки матрицы G.

Дилер шифрует ключом $K_{enc}(h)$ столбцы матрицы G: $REnc(g_{ij}, K_{enc}(h), alg, h)$ где $i = 1,..,n, j = 1,2, h \in \{1, \dots, k\}$ и получает шифротексты, соответствующие элементам столбцов матрицы G для каждой копии базы данных.

Дилер получает матрицу $G_{enc}(h)$, где $h \in \{1, \dots, k\}$.

Шаг 5. Матрицу $G_{enc}(h)$, где $h \in \{1, \dots, k\}$, дилер сортирует по первому столбцу.

Шаг 6. Дилер загружает матрицу $G_{enc}(h)$, где $h \in \{1, \dots, k\}$ на облако.

Шаг 7. Шаги 4-6 дилер выполняет в цикле для каждой копии базы данных ■.

Число шифротекстов, переданных каждой из k копий базы данных равно $3n$.

4. Описание выполнения алгоритма запроса клиента дилером

4.1 Использование множества номеров в запросе для сокрытия номера *i*

Пусть клиент интересуется элементом $x_i \in X$.

С целью сокрытия номера, запрашиваемого клиентом, запрос дилера к облаку будет формироваться как множество номеров.

Поскольку предполагается наличие фальшивых клиентов, запрашивающих конкретный номер, запрос будет выполнен следующим образом: дилер передает в запросе к облаку не только этот номер (в зашифрованном виде), а также множество номеров, содержащих запрашиваемый номер.

Напомним, что отрезок [1,n] разбивается на l интервалов по $L_p = \frac{n}{l}$ элементов в каждом интервале.

По построению матрицы M каждой строке i соответствует единственная строка из первого интервала. Соответственно, каждому элементу m_{i1} матрицы M соответствует строка из первого интервала, по которой i-я строка была построена. Таким образом, по строке i может быть полностью восстановлена искомая строка из первого интервала. После восстановления строки из первого интервала по алгоритму построения матрицы M можно восстановить те l-1 строк, которые формируются на основе искомой строки из первого интервала. Столбцы этих восстановленных l строк формируют множества $Set_{ij}, j = 1, \dots, d$.

Для запрашиваемого i-го бита дилер формирует множество номеров $Set_i = \cup Set_{ij}$, содержащее i-ю строку матрицы M.

Дилер шифрует и отправляет облаку запрос, включающий в качестве параметра это множество. Далее после ответа облака, дилер получает в качестве ответа также множество значений.

Для построения множества Set_i используем следующую процедуру, которая основана на восстановлении l строк матрицы M. Для восстановления строк матрицы M по элементу m_{i1} необходимо найти:

- порядковый номер интервала (w), в который попал этот элемент для определения элемента вектора-столбца b_w;
- порядковый номер элемента в интервале для вычисления числа a;
- порядковый номер элемента, соответствующего элементу m_{i1} в первом интервале, поскольку восстановление матрицы M выполняется на базе строк первого интервала.

Приведем m_{i1} (номер искомого бита) по модулю L_p. Обозначим через $r = m_{i1}\ mod L_p$ результат приведения.

Определяется номер интервала, в который попало число m_{i1}. По определению матрицы M номер интервала определяется индексом i.

$$w = \left\lfloor \frac{m_{i1} - 1}{L_p} \right\rfloor + 1$$

Заметим, что $w \in \{1, \dots, l\}$. Положим

$$m'_{i1} = \begin{cases} r, \text{если } r \neq 0 \\ L_p, \text{если } r = 0 \end{cases}$$

порядковый номер в интервале.

Пусть $a = (m'_{i1} - b_w) mod L_p$. В этом случае номер строки i' в первом интервале вычисляется по формуле $i' = (a + b_1) mod L_p$, если i'=0, то i'=L_p. Для номера строки i' при помощи датчика псевдослучайных чисел *PRNG(db,i)* получим *d-1* различных чисел от 1 до L_p аналогично тому, как строилась матрица M.

Для номера строки i' из диапазона $[1, L_p]$, по аналогии с алгоритмом генерации матрицы M, построим l-1 строк.

Столбцами этих l строк будут множества $Set_{ij}, j = 1, \dots, d$, где множество Set_{i1} содержит элемент m_{i1}.

Мощность множества $|Set_{ij}|$ равна l. Следовательно, мощность множества $|Set_i|$ не более $l \cdot d$.

4.2 Описание алгоритма построения множества Set_i

На вход алгоритма подается номер запрашиваемого бита i. На выходе алгоритм формирует множество Set_i.

Шаг 1. Дилер находит номер интервала w, в который попал номер запрашиваемого бита i и его порядковый номер в этом интервале m'_{i1}.

Шаг 2. Дилер находит число a для номера интервала w, куда попал номер i: $a = (m'_{i1} - b_w) mod L_p$.

Шаг 3. Дилер находит номер строки i' в диапазоне $[1, L_p]$ по формуле: $i' = (a + b_1) mod L_p$, если i'=0, то i'=L_p.

Шаг 4. Дилер при помощи датчика псевдослучайных чисел *PRNG(db,i)* восстанавливает строку матрицы M для номера i', состоящую из d различных натуральных чисел, аналогично алгоритму построения строки матрицы M.

Шаг 5. Дилер строит множество: $Set_i = \cup\, Set_{ij}$, где $j = 1, \dots, d$ ■.

4.3 Запрос клиентом *i*-го бита. Предварительные замечания

Запрос клиента на получение данных поступает дилеру. Дилер проверяет полномочия клиента и, если они подтверждены, то выполнение запроса будет санкционировано.

Противник в облаке собирает статистику запросов. Также предполагается наличие фальшивых клиентов, которые могут вступать в коалицию с облаком. Поэтому дилер скрывает запрашиваемый клиентом номер среди множества номеров.

При запросе клиентом i-го бита дилер строит множество Set_i, которое содержит элементы i-й строки матрицы M. Напомним, что в i-й строке элемент m_{i1} является искомым номером. Множество Set_i является избыточным, чтобы скрыть от противника информацию о запрашиваемом бите.

При генерации множества Set_i дилер при помощи датчика псевдослучайных чисел *PRNG(db,i)* восстанавливает i-ю строку матрицы M, состоящую из d различных натуральных чисел.

Из множества элементов Set_i только элементы i-й строки матрицы M нужны дилеру для получения значения i-го бита.

Напомним, что физически в облаке хранятся матрицы $G_{enc}(h)$, $h \in \{1, \dots, k\}$. Строка матрицы $G_{enc}(h)$ состоит из шифротекстов номера строки, произвольного и корректирующего битов.

Разобьем множества Set_i на непересекающиеся подмножества.

Для каждого элемента множества Set_i случайно и равномерно выбирается номер копии базы данных. Все элементы множества Set_i, для которых выбрана копия базы данных h, обозначим через множества Set_i^h, где $h \in \{1, \dots, k\}$. Очевидно, что множества Set_i^h не пересекаются между собой, поскольку все элементы множества Set_i различны и каждому элементу ставится в соответствие ровно один номер h. Объединение множеств Set_i^h содержит все элементы множества Set_i, то есть $Set_i = \cup\, Set_i^h$, где $h \in \{1, \dots, k\}$.

Далее каждое множество Set_i^h шифруется ключом $K(h)$, полученные шифротексты сортируются для каждого множества Set_i^h отдельно. В результате получим множества $Set_i^{enc(h)}$ – зашифрованные соответствующим ключом $K(h)$ и затем отсортированные.

Для каждого множества $Set_i^{enc(h)}$ исходному номеру бита будет соответствовать номер шифротекста в перестановке. Это соответствие необходимо запомнить и хранить на время выполнения запроса, чтобы дилер мог восстановить значение запрашиваемого бита.

Дилер на время выполнения запроса формирует матрицу S_{num} из d строк и трех столбцов ($i = 1, \dots, d, j = 1,2,3$).

$$S_{num} = \begin{pmatrix} s_{11} & s_{12} & s_{13} \\ \vdots & \vdots & \vdots \\ s_{d1} & s_{d2} & s_{d3} \end{pmatrix}.$$

Первым столбцом матрицы S_{num} являются элементы восстановленной i-ой строки матрицы M.

Ранее множество Set_i, содержащее элементы i-й строки матрицы M, было разбито на множества Set_i^h. Для каждого множества Set_i^h, а следовательно, для элементов i-й строки матрицы M были выбраны копии базы данных. Второй столбец матрицы S_{num} содержит выбранные номера копий баз данных для элементов первого столбца матрицы S_{num}.

Третий столбец матрицы S_{num} – номер шифротекста в перестановке множества $Set_i^{enc(h)}$ для соответствующего элемента первого столбца. Таким образом, s_{j3} является функцией от трех параметров, которая позволяет получить соответствие между исходным номером и номером шифротекста в перестановке:

$$s_{j3} = F\left(s_{j1}, s_{j2}, Set_i^{enc(s_{j2})}\right).$$

Матрица S_{num} позволяет дилеру после получения ответа от облака без выполнения расшифрования сразу отбросить данные, кроме данных, необходимых для выполнения запроса.

Матрица S_{num} формируется у дилера и известна только дилеру. Матрица S_{num} является секретной.

На k копий базы данных дилер отправляет в общей сложности $l \cdot d$ шифротекстов для элементов множества Set_i.

После выполнения запроса дилер при помощи матрицы S_{num} определяет значения элементов для строки матрицы M из копий базы данных. После расшифрования значений, дилер определяет значение запрашиваемого бита и отправляет его клиенту.

4.4 Подготовка дилером запроса к облаку

На входе алгоритм получает номер запрашиваемого бита i. На выходе дилер получает зашифрованные и отсортированные множества $Set_i^{enc(h)}$ и матрицу S_{num}.

Шаг 1. Клиент обращается к дилеру и передает ему номер бита i, значение которого хочет получить. Дилер аутентифицирует клиента.

Шаг 2. Дилер восстанавливает i-ю строку матрицы M и генерирует множество Set_i.

Шаг 3. Дилер на время выполнения запроса i-го бита резервирует память для матрицы S_{num} из d строк и трех столбцов ($i = 1, \dots, d, j = 1,2,3$).

Шаг 4. Дилер заполняет первый столбец матрицы S_{num} элементами i-й строки матрицы M.

Шаг 5. Дилер выполняет разбиение множества Set_i на непересекающиеся подмножества Set_i^h путем случайного и равномерного выбора номера копии базы данных для каждого элемента множества Set_i.

Дилер заполняет второй столбец матрицы S_{num} номерами копий базы данных, соответствующих номерам в первом столбце.

Шаг 6. Дилер шифрует элементы множеств Set_i^h в соответствии с выбранной копией базы данных ключом $K(h), h \in \{1, \dots, k\}$, получает шифротексты и сортирует их (множества $Set_i^{enc(h)}$).

Шаг 7. Дилер заполняет третий столбец матрицы S_{num} номерами шифротекстов в перестановке множеств $Set_i^{enc(h)}$ для элементов первого столбца ■.

4.5 Выполнение запроса дилера к облаку и ответ облака

На вход алгоритма подаются зашифрованные и отсортированные множества $Set_i^{enc(h)}$. На выходе дилер получает шифротексты значений битов и корректировочных битов для элементов множеств $Set_i^{enc(h)}$.

Шаг 1. Для каждой копии базы данных дилер отправляет на облако отсортированные шифротексты номеров битов для множеств $Set_i^{enc(h)}$.

Шаг 2. Для запрошенных шифротекстов номеров битов облако возвращает дилеру шифротексты значений битов и корректировочных битов (второй и третий столбцы матрицы $G_{enc}(h)$, где $h \in \{1, \dots, k\}$). Порядок возвращаемых шифротекстов соответствует исходной сортировке шифротекстов для множеств $Set_i^{enc(h)}$ ■.

4.6 Обработка дилером ответа облака

На входе дилер получает второй и третий столбцы матриц $G_{enc}(h)$ для шифротекстов множеств $Set_i^{enc(h)}$. На выходе значение i-го бита.

Шаг 1. Дилер при помощи матрицы S_{num} выбирает зашифрованные значения бита и корректировочного бита для номеров первого столбца матрицы S_{num}. Остальные шифротексты отбрасываются.

Шаг 2. Дилер расшифровывает значения битов для номеров первого столбца матрицы S_{num} и корректировочный бит для элемента s_{11} матрицы S_{num}. По построению матрицы S_{num} запрашиваемый номер является элементом s_{11}.

Шаг 3. Дилер использует формулу $x_i = \bigoplus_{j=1}^{d} g_{m_{ij}1} \oplus g_{i2}$ для вычисления значения i-го бита и отправляет клиенту полученное значение ■.

5. Оценка требуемой памяти и сложности алгоритма

5.1 Объем информации, который необходимо хранить дилеру

В процессе работы на время загрузки базы данных дилер генерирует k ключей $K_{enc}(h)$. Эти ключи служат для вероятностного шифрования значений битов. После загрузки ни ключи, ни шифротексты дилер не хранит.

На протяжении всего времени существования базы данных дилер хранит информацию для этой базы данных объема n:

- значение для инициализации датчиков случайных чисел *PRNG(db,i)* и *PRNG2(db,m)*;
- k ключей $K(h), h \in \{1, \dots, k\}$ для шифрования односторонней функцией;
- k ключей $K_{dec}(h)$, где $h \in \{1, \dots, k\}$ для вероятностного расшифрования значений битов.

Заметим, что на практике число n битов в базе данных традиционно предполагается 2^{30} - 2^{40}, а число копий баз данных k не превышает 16 (то есть 2^4). Таким образом, k значительно меньше n, а хранение k ключей для шифрования номеров битов и k ключей для расшифрования значений битов занимает $2 \cdot k \cdot l_{key}$, где l_{key} - длина ключа. Длина ключа на практике чаще всего ограничена 256 байтами [13]. Следовательно, дилер хранит объем информации значительно меньше n.

5.2 Основные результаты

Утверждение 1. Общая коммуникационная сложность для получения значения номера бита без раскрытия его номера равна $l \cdot d \cdot 3$ шифротекстов.

Доказательство. Дилер посылает копиям базы данных $l \cdot d$ (где $l = \sqrt[d]{n}$) шифротекстов для запроса значений. Облако отвечает $l \cdot d \cdot 2$ шифротекстами значений ■.

Проанализируем вероятность угадывания противником номера запрашиваемого бита.

Если запрошены два номера бита, то они могут принадлежать либо одному, либо разным множествам Set_i. Противник может увидеть, выполнен ли запрос к одному из множеств Set_i, на которые разбита база данных или к разным. Если запросы выполнены к одному множеству, то они неразличимы. Таким образом, если противник совершает n или более запросов, он не узнает номер бита. Максимум, какую информацию противник может получить - такое подмножество множества Set_i, что при запросе к каждому элементу из подмножества, на облаке осуществляется доступ ко всем битам из множества Set_i и только к ним. Что будет означать, что противник не знает, а только угадывает, какой именно бит из данного множества был выбран ■.

Заметим, что так как любым двум элементов x_{i_1} и x_{i_2}, $x_{i_1}, x_{i_2} \in Set_{i1}$ соответствует единственная строка из первого интервала, то множества $Set_{i2}, \dots, Set_{id}$ совпадают для элементов x_{i_1} и x_{i_2}. Следовательно, для всех элементов множества Set_{i1} элементы множества Set_i совпадают. Поэтом запросы к облаку будут неотличимы для всех элементов множества Set_{i1}.

Утверждение 2. При запросе фиктивными клиентами не менее n номеров битов и наличии пассивного противника на облаке вероятность угадывания противником номера бита не более $\frac{1}{l}$.

Доказательство. Множество Set_{i1} ($Set_{i1} \subset Set_i$) порождается одинаково для всех l чисел, в него входящих. То есть существует одинаковое множество для l различных номеров битов. Вероятность угадывания номера из множества Set_{i1} равна $\frac{1}{l}$, так как Set_{i1} имеет мощность l.

Выполнив n запросов, противник определяет элементы, входящие в каждое множество Set_i (в предположении, что $k < ld$).

Если фиктивные клиенты будут выполнять любое количество запросов большее n, противник все равно не получит никакой дополнительной информации.

В этом случае вероятность угадывания номера *i* будет не более $\frac{1}{l}$ ■.

Так как значение k по сравнению с n мало, хранение k пар ключей для шифрования/расшифрования значений битов не требует много памяти и занимает $2 \cdot k \cdot l_{key}$.

Докажем, что предложенный протокол удовлетворяет условию лаконичности.

Теорема. Предложенная схема организации базы данных размера n и запроса к ней удовлетворяет следующим свойствам.

На время загрузки базы данных дилер генерирует:

- k ключей $K_{enc}(h)$ для вероятностного шифрования значений битов, которые после окончания загрузки не хранятся.

На протяжении всего времени существования базы данных дилер хранит только следующую информацию для данной базы данных объема n:

- значение для инициализации датчиков случайных чисел *PRNG(db,i)* и *PRNG2(db,m)*;
- k ключей $K(h), h \in \{1, \dots, k\}$ для шифрования односторонней функцией;
- k ключей $K_{dec}(h)$, где $h \in \{1, \dots, k\}$, для вероятностного расшифрования значений битов.

Объем хранимой информации много меньше n.

Коммуникационная сложность запроса $l \cdot d \cdot 3$ шифротекстов.

При атаке с использованием фальшивых клиентов, выполняющих любое число запросов и предположении о наличии пассивного противника на облаке вероятность угадывания номера бита противником не более

$$\frac{1}{l}.$$

Доказательство. Из Утверждения 1 следует, что коммуникационная сложность запроса равна $l \cdot d \cdot 3$ шифротекстов.

Из Утверждения 2 следует, что при любом числе запросов номеров битов фиктивными клиентами и наличии пассивного противника на облаке вероятность угадывания номера бита не более $\frac{1}{l}$ ■.

Комментарий. Так как k – число копий баз данных, и это число мало на практике, а дилер хранит не более $O(k)$ информации, то дилер в состоянии обслуживать запросы к значительному числу баз данных. То факт, что $k \ll n$ позволяет эффективно использовать облако как место хранения значительного объема информации.

Список литературы / References

[1] Chor B., Goldreich O. et al. Private Information Retrieval. In Proc. of the IEEE Annual Symposium on Foundations of Computer Science, 1995, pp. 41-50.

[2] Chor B., Goldreich O. et al. Private Information Retrieval. Journal of the ACM, vol. 45, no. 6, 1998, pp. 965-982.

[3] Gasarch W. A survey on private information retrieval. Bulletin of the EATCS, 2004, pp. 72-107

[4] Yekhanin S. Locally Decodable Codes and Private Information Retrieval Schemes. Springer-Verlag Berlin Heidelberg, 2010, 82 p.

[5] Kushilevitz E., Ostrovsky R. Replication is not needed: Single database, computationally-private information retrieval (extended abstract). In Proc. of the 38th Annual Symposium on Foundations of Computer Science, 1997, pp. 364-373.

[6] Kushilevitz E., Ostrovsky R. One-way trapdoor permutations are sufficient for non-trivial single-server private information retrieval. Lecture Notes in Computer Science, vol. 1807, 2000, pp. 104-121.

[7] Ostrovsky R., Skeith III W. E. A Survey of Single-Database Private Information Retrieval: Techniques and Applications. Lecture Notes in Computer Science, vol. 4450, 2007, pp. 393-411.

[8] Aguilar-Melchor C., Barrier J., Fousse L. XPIR: Private Information Retrieval for Everyone, Proceedings on Privacy Enhancing Technologies Symposium, 2016, issue 2, pp. 155-174.

[9] Варновский Н.П., Нестеренко Ю.В., Ященко В.В. Введение в криптографию. Из-во МЦНМО, 2012, 348 стр. / Varnovsky N.P., Nesterenko Yu.V., Yashchenko V.V. Introduction to cryptography. MCCME, 2012, 348 p. (in Russian).

[10] Demmler D., Herzberg A., Schneider T. RAID-PIR: Practical multi-server PIR. In Proc. of the 6th edition of the ACM Workshop on Cloud Computing Security, 2014, pp. 45-56.

[11] Angel S., Chen H. et al. PIR with compressed queries and amortized computation. In Proc. of the IEEE Symposium on Security and Privacy, 2018, pp. 1-18.

[12] "MuchPIR Demo". URL: https://github.com/ReverseControl/MuchPIR, accessed 22.08.2022.

[13]Wahid M.N.A., Ali A. et al. A Comparison of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish for Guessing Attacks Prevention. Journal of Computer Science Applications and Information Technology, vol. 3, no. 2, 2018, pp. 1-7.

Информация об авторах / Information about authors

Сергей Анатольевич МАРТИШИН – кандидат физико-математических наук, научный сотрудник отдела теоретической информатики. Его научные интересы включают параллельные алгоритмы, базы данных, облачные вычисления,

Sergey Anatolievich MARTISHIN – PhD, researcher of the Department of Theoretical Computer Science. His research interests include parallel algorithms, databases, cloud computing.

Марина Валерьевна ХРАПЧЕНКО – научный сотрудник отдела теоретической информатики. Её научные интересы включают параллельные алгоритмы, базы данных, облачные вычисления,.

Marina Valerievna KHRAPCHENKO – researcher of the Department of Theoretical Computer Science. Her research interests include parallel algorithms, databases, cloud computing.

Александр Владимирович ШОКУРОВ – кандидат физико-математических наук, доцент, заведующий отделом теоретической информатики. Сфера научных интересов: алгебраические структуры в полях Галуа, базисы Гребнера, модулярная арифметика, нейрокомпьютерные технологии, цифровая обработка сигналов, криптографические методы защиты информации.

Alexander Vladimirovich SHOKUROV – PhD of Physical and Mathematical Sciences, Professor, Head of the Department of Theoretical Computer Science. Research interests: algebraic structures in the Galois fields, modular arithmetic, neurocomputer technologies, Grobner bases, digital signal processing, cryptographic methods for protecting information.