DOI: 10.15514/ISPRAS–2020–33(2)–12

Выполнимость мю-исчисления с арифметическими ограничениями

[1] *Й. Лимон, ORCID: 0000-0002-3754-6926 <zs16017367@estudiantes.uv.mx>*
[2] *Э. Барсенас, ORCID: 0000-0002-1523-1579 <barcenas@unam.mx>*
[1] *Э. Бенитес-Герреро, ORCID: 0000-0001-5844-4198 <edbenitez@uv.mx>*
[2] *Г. Молеро-Кастильо, ORCID: 0000-0002-6330-6408 <gmoleroca@fi-b.unam.mx>*
[2] *А. Веласкес Мена, ORCID: 0000-0003-3509-6236 <mena@fi-b.unam.mx>*

[1] *Университет Веракрузана.*
Мексика, 91000, Веракрус, Халапа
[2] *Национальный автономный университет Мексики.*
Мексика, 04510, Мехико, Мэрия Койоакана

Анотация. Пропозициональное модальное μ-исчисление является хорошо известным языком спецификаций систем помеченных переходов. В этой работе мы изучаем расширение этой логики с помощью обратных модальностей и арифметических ограничений Пресбургера, интерпретируемых на древовидных моделях. Мы описываем алгоритм выполнимости, основанный на построении по уровням моделей Фишера-Ларднера. Также сообщается о совместном выполнении нескольких экспериментов. Кроме того, мы описываем применение алгоритма для решения задач статического анализа полуструктурированных данных.

Ключевые слова: модальные логики; автоматизированное построение логического вывода; формальная верификация; пресбургеровская арифметика; полуструктурированные данные

Для цитирования: Лимон Й., Барсенас Э., Бенитес-Герреро Э., Молеро-Кастильо Г., Веласкес Мена А. Выполнимость мю-исчисления с арифметическими ограничениями. Труды ИСП РАН, том 33, вып. 2, 2021 г., стр. 191-200. DOI: 10.15514/ISPRAS–2021–33(2)–12

Mu-Calculus Satisfiability with Arithmetic Constraints

[1] *Y. Limón, ORCID: 0000-0002-3754-6926 < zs16017367@estudiantes.uv.mx>*
[2] *E. Bárcenas, ORCID: 0000-0002-1523-1579 <barcenas@unam.mx>*
[1] *E. Benítez-Guerrero, ORCID: 0000-0001-5844-4198 <edbenitez@uv.mx>*
[2] *G. Molero-Castillo, ORCID: 0000-0002-6330-6408 <gmoleroca@fi-b.unam.mx>*
[2] *A. Velázquez-Mena, ORCID: 0000-0003-3509-6236 <mena@fi-b.unam.mx>*

[1] *Universidad Veracruzana,*
Xalapa, Veracruz, México, 91000
[2] *National Autonomous University of Mexico,*
Coyoacan Mayor's Office, Mexico City, CDMX, C.P. 04510, Mexico

Abstract. The propositional modal μ-calculus is a well-known specification language for labeled transition systems. In this work, we study an extension of this logic with converse modalities and Presburger arithmetic constraints, interpreted over tree models. We describe a satisfiability algorithm based on breadth-first construction of Fischer-Lardner models. An implementation together several experiments are also reported. Furthermore, we also describe an application of the algorithm to solve static analysis problems over semi-structured data.

Keywords: Modal Logics; Automated Reasoning; Formal Verification; Presburger Arithmetic; Semi-Structured Data

For citation: Limón Y., Bárcenas E., Benítez-Guerrero E., Molero-Castillo G., Velázquez-Mena A. Mu-Calculus Satisfiability with Arithmetic Constraints. Trudy ISP RAN/Proc. ISP RAS, vol. 33, issue 2, 2021, pp. 191-200 (in Russian). DOI: 10.15514/ISPRAS–2021–33(2)–12.

1. Введение

Модальные логики широко известны как языки спецификации для верификации аппаратных и программных систем. Эти логики получили известность в сообществе формальной верификации благодаря присущему им тонкому балансу между их выразительной силой и вычислительной сложностью связанных с ними алгоритмов формирования рассуждений.

Можно рассматривать μ-исчисление как расширение модальной логики высказываний K операторами неподвижной точки. Известно, что проблема разрешимости μ-исчисления, а существования или отсутствия алгоритма для проверки валидности любой формулы, является EXPTIME-полной [1]. Обратные модальности (converse modality) – это конструкторы, способные моделировать свойства в прошлом, если мы рассматриваем переходы в моделях как временны*е* отношения. Номиналы (nominal) – это отдельные формулы, обозначающие одиночные состояния в моделях. Градуированные модальности (graded modaly) ограничивают количество последующих состояний модели некоторым натуральным числом. Если расширить μ-исчисление чем-то одним – обратными модальностями, номиналами или градуированными модальностями – проблема разрешимости останется EXPTIME-полной. Расширение любыми двумя из этих трех конструкций также оставляет проблему разрешимости EXPTIME-полной. Однако полностью обогащенное μ-исчисление, включающее обратные модальности, номиналы и градуированные модальности и номиналами, разрешимым не является [1]. При интерпретации над древовидными структурами для полностью обогащенного μ-исчисления EXPTIME-полная разрешимость сохраняется [2]. Кроме того, показано, если полностью обогащенное μ-исчисление расширить еще и ограничениями арифметики Пресбургера (Mojżesz Presburger) – ограничениями последующих узлов арифметическими выражениями Пресбургера), – проблема разрешимости останется EXPTIME-полной [3]. Доказательство основано на алгоритме выполнимости. Однако о реализации не сообщается.

В настоящей работе мы описываем реализацию вместе с несколькими экспериментами. Начальные результаты этой работы были впервые представлены в [4].

Кроме расширения μ-исчисления градуированными модальностями, изучались и другие логики с числовыми ограничениями. В [5] показана PSPACE-полная разрешимость расширения модальной логики K ограничениями арифметики Пресбургера. В [6] показано, что EXPTIME-полной разрешимостью обладает логика с фиксированной точкой над деревьями с ограничениями Пресбургера. В [7] показана неразрешимость монадической логика второго порядка с арифметикой Пресбургера. Хотя в некоторых из этих работ явно или неявно сообщается о соответствующем алгоритме принятия решений, реализация не упоминается. Высокая вычислительная сложность этих алгоритмов исключает возможность непосредственной реализации. В настоящей работе мы предлагаем бинарно-векторное представление узлов в древовидных моделях. Это позволяет справиться с потенциально экспоненциальным размером моделей.

В литературе известно несколько арифметических решателей. Z_3 – это инструмент, разработанный над арифметическим расширением логики первого порядка [8]. Расширение Z_3 оператором неподвижной точки, называемое μZ, описано в [9]. Этот конструктор позволяет выражать рекурсивные свойства над реляционными структурами. Авторы работы [11] представили би-интуиционистский модальный алгоритм μ-исчисления, который переписывает логические формулы первого порядка в формулы модальной логики и решает

рекурсивные неравенства, а его модели являются древовидными [11]. Еще одним известным инструментом, способным производить логический вывод при наличии арифметических ограничениях, является Mona [10]. Этот инструмент построен на монадической теории второго порядка. Следует отметить, что Мона может представлять или выражать то же самое, что и μ-исчисление. Хотя доказано, что вычислительная сложность этой теории не элементарна, обсуждаются несколько приложений с логическим выводом над полуструктурированными данными.

В отличие от этих инструментов, реализация алгоритма, описываемая в настоящей статье, строится на менее дорогостоящей теории.

Насколько нам известно, в современной литературе упоминаются два фреймворка для логического вывода над полуструктурированными данными [12-14]. Точнее, эти фреймворки решают несколько проблем вывода для разрешимых фрагментов запросов XPath и XML-схем, таких как вложенность результатов (query containment) запросов и проверка типов. Оба схемы основаны на алгоритмах выполнимости μ-исчисления без арифметических ограничений. В этой статье мы описываем несколько экспериментов по логическому выводу над запросами XPath с арифметическими ограничениями.

2. μ-исчисление с арифметическими ограничениями

В этом разделе мы вводим синтаксис и семантику μ-исчисления с арифметическими ограничениями на древовидных моделях. Пусть заданы алфавиты $PROP$ и MOD. $PROP$ используется как множество высказываний, а MOD – как множество модальностей. Множеством модальностей, например, может быть $\{1,2,3,4\}$. Тогда на интуитивном уровне в древовидных моделях 1 обозначает отношение непосредственных потомков, 2 обозначает правого брата, 3 – родителя и 4 – левого брата. Множество формул μ-исчисления определяется следующей грамматикой:

$$\varphi ::= p|X|\neg\varphi|\varphi \vee \varphi|\langle m\rangle\varphi|mX.\varphi|\gamma > b,$$
$$\gamma ::= \alpha\varphi\,|\gamma + \gamma,$$

где p – высказывание, m – модальность, X - переменная, $\alpha \in \mathbb{Z} \setminus \{0\}$, и $b \in \mathbb{N}$.

Теперь мы дадим интуитивное представление об интерпретации формул как подмножества узлов деревьев: высказывания интерпретируются как метки узлов; отрицание – как дополнение множества; дизъюнкция – как объединение множеств; модальные формулы $\langle 1\rangle\varphi$ – как узлы с m-доступностью для узла, верифицирующего φ; формулы с неподвижной точкой – как рекурсия и арифметические формулы $\gamma > b$ – как узлы с дочерними элементами, удовлетворяющими арифметическому выражению. Другие конструкции могут быть определены обычным образом: $\varphi \wedge \psi \coloneqq \neg(\neg\varphi \vee \neg\psi)$, $\gamma \leq b \coloneqq \neg(\gamma > b)$. Например, формула φ: $\varphi = a \wedge \langle 1\rangle b$ обозначает узлы с именем a или с потомком a.

Для того чтобы описать точную семантику формулы, рассмотрим определение дерева в стиле структуры Крипке (Saul Aaron Kripke) (N, R^m, L), представленное в [3]. Рассмотрим дерево T и валюацию $V: Var \to 2^N$, где Var – множество переменных. В [3] представлено формальное определение μ-исчисления со счетчиками. Касательно дерева T мы говорим, что формула φ выполнима тогда и только тогда, когда существует интерпретация φ над деревом T такая, что валюация V не пуста, то есть $[\![\varphi]\!]_V^T \neq \emptyset$. Формула φ валидна тогда и только тогда, когда она выполнима для каждого дерева.

3. Выполнимость

В этом разделе мы описываем алгоритм выполнимости, как он представлен в [3]. Алгоритм основан на восходящем построении деревьев Фишера-Ладнера (Michael John Fischer, Richard Emil Ladner) [15]. Сначала мы опишем это понятие. Для технического удобства и без потери

общности мы рассматриваем только бинарные деревья. Будем считать, что при интерпретации формул в бинарных деревьях модальности 1, 2, 3, 4 обозначают первого ребенка (слева направо), следующего родного брата, родителя и предыдущего родного братом соответственно. В этой работе мы для определения алгоритма выполнимости рассматриваем формулы только в нормальной форме отрицания (negation normal form) [3].

Для заданной входной формулы φ узлы в соответствующем дереве Фишера-Ладнера интуитивно определяются как наборы подформул φ, такие что каждая подформула выполняются в соответствующем узле. Поскольку арифметические формулы подсчитывают дочерние узлы, мы вводим в дочерние узлы новые высказывания, чтобы отслеживать число узлов. Мы называем эти предложения счетчиками, и они кодируются в двоичной системе. Это кодирование основано на булевой комбинации высказываний, встречающихся без отрицаний. Затем мы определяем $(C(\varphi = b) \coloneqq \overline{\psi^i}$, где $i \in \{0, \dots, [log(b)]\}$, $\overline{\psi^i}$ – последовательность высказываний, а b – целое число.

Определение 1 (дерево Фишера-Ладнера).

Дерево Фишера-Ладнера T определяется как

- пустое дерево или
- (n_r, T_1, T_2), где n_r – корневой узел, а T_1 или T_2 являются первым потомком и последующими поддеревом соответственно.

Пример 1: Рассмотрим формулу: $\varphi = a \wedge (b - c > 0) \wedge \langle 1\rangle\mu X.(a \vee \langle 2\rangle X)$, и соответствующий набор узлов N^φ. Тогда $T = (n_1, (n_2, \emptyset, (n_3, (n_4, , \emptyset), \emptyset)), \emptyset)$, где

- $n_1 = \{a, (b - c > 0), \langle 1\rangle\mu X.(a \vee\langle 2\rangle X), \langle 1\rangle\top, \langle 1\rangle\mu X.(b\vee\langle 2\rangle X), \langle 1\rangle\mu X.(c\vee\langle 2\rangle X)\}$;
- $n_2 = \{b, C(b) = 2, C(c) = 1, \langle 2\rangle\mu X.(a \vee\langle 2\rangle X), \langle 2\rangle\top,\langle 2\rangle\mu X.(a\vee\langle 2\rangle X)$;
- $n_3 = \{c, C(b) = 1, C(c) = 1, \langle 2\rangle\mu X.(a \vee\langle 2\rangle X), \langle 2\rangle\top,\langle 2\rangle\mu X.(b\vee\langle 2\rangle X)\}$;
- $n_4 = \{a, b, C(b) = 1, C(c) = 0, \langle 4\rangle\top\}$;
- $n_5 = \{a, C(b) = 0, C(c) = 0, \langle 4\rangle\top\}$;
- $n_6 = \{p', C(b) = 0, C(c) = 0, \langle 4\rangle\top\}$;
- $n_7 = \{c, C(b) = 0, C(c) = 1, \langle 4\rangle\top\}$.

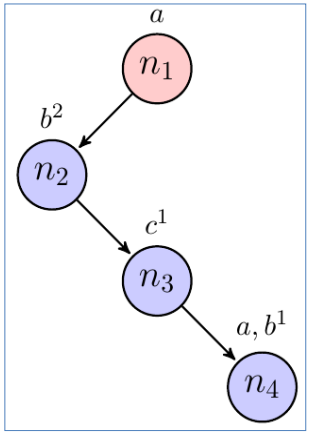

Рис. 1. Дерево Фишера-Ладнера для $a \wedge (b - c > 0) \wedge \langle 1\rangle\mu X.(a \vee \langle 2\rangle X)$
Fig. 1. A Fischer-Ladner tree for $a \wedge (b - c > 0) \wedge \langle 1\rangle\mu X.(a \vee \langle 2\rangle X)$

Множество узлов N^{φ} определяется исходя из множества подформул, называемого $lean\ (\varphi)$ и содержащего высказывания, модальные формулы и счетчики. На рис. 1 изображено графическое представление соответствующего дерева Фишера-Ладнера. В корневом узле n_1 счетчики установлены в 0. Это связано с тем, что счетчики появляются только на узлах-братьях: n_2,n_3 и n_4. Счетная формула выполняется в корневом узле n_1 потому что дочерние узлы n_1 удовлетворяют ограничениям Пресбургера.

Теперь определим алгоритм проверки выполнимости. Для входной формулы φ алгоритм строит восходящим образом дерево Фишера-Ладнера, выполняющее φ, если такое дерево существует. Алгоритм выполнимости определен на рис. 2. Функция *Init* вычисляет листья дерева. Между узлами Фишера-Ладнера и формулами можно определить отношение выполнимости. Функция *Update* последовательно добавляет родителей к ранее построенным деревьям.

$Y \leftarrow N_i{}^{\Phi}$
$X \leftarrow Init(\Phi)$
$X' \leftarrow \emptyset$
while $N_i{}^{\Phi} \nvdash \Phi$ *and* $X = N^{\Phi}$ **do**
$X' \leftarrow X$
$X \leftarrow Update(X', Y)$
$Y \leftarrow Y \backslash root(X)$
end while
if $X \vdash \Phi$ **then**
return Φ **is satisfiable**
end if
return Φ **is not satisfiable**

Рис. 2. Алгоритм определения выполнимости для μ-исчисления с арифметическими ограничениями
Fig. 2. Satisfiability algorithm for the μ-calculus with arithmetic constraints

Теорема 1 (Корректность [3])

Для заданной входной формулы φ алгоритм устанавливает выполнимость φ тогда и только тогда, когда существует древовидная структура T, такая что $[\![\varphi]\!]_V^T \neq \emptyset$, для любой валюации V.

4. Реализация

Для реализации алгоритма мы используем битово-векторное представление узлов Фишера-Ладнера. Поскольку узлы определяются как подмножества *lean*, формулы в узлах в векторном представлении битовым единицам, а битовые нули соответствуют отсутствующим формулам. Например, предположим, что *lean* определяется следующим образом: $lean(\varphi) = \{\psi_1, \psi_2, \ldots, \psi_n\}$. Если узел $n \subseteq lean(\varphi)$ содержит следующие формулы $n = \{\psi_{i_1}, \psi_{i_2}, \ldots, \psi_{i_k}\}$, то битово-векторное представление *n* содержит единицы в битах $i_1, i_2, \ldots i_k$ и нули во всех остальные битах. Это битово-векторное кодирование узлов дерева Фишера-Ладнера подразумевает битово-векторное кодирование всех функций алгоритма.

Алгоритм выполнимости был реализован на языке Java с использованием технологии JavaService Faces. Для тестирования этой реализации мы использовали компьютер со характеристиками: операционная система Windows 10, процессор Intel i7-6700HQ 2,6 ГГц, 16 ГБ оперативной памяти.

Таблица 1. Выполнимые формулы
Table 1. Satisfiable Formulas

Формула	N^{φ}	***Lean***	**Время (мс)**
a	36	6	22
$\neg a$	35	5	11
$a \vee b$	84	7	20
$a \wedge b$	84	7	124
$a \vee b \vee c \vee d \vee e \vee f$	756	10	30
$a \wedge b \wedge c \wedge d \wedge e \wedge f$	756	10	20
$\langle 1 \rangle a$	72	7	523
$\langle 1 \rangle a \wedge b$	168	8	2199
$\langle 1 \rangle \langle 1 \rangle > \top$	24	6	24814
$\langle 1 \rangle \langle 2 \rangle a$	144	8	100907
$1 * a > 0$	198	10	12333
$(1 * a > 0) \wedge c$	462	11	9189
$(a > 0) \wedge c \wedge \langle 1 \rangle$b	2070	13	9189
$1 * a > 1$	3258	10	179309
$1 * a + 1 * b > 0$	966	13	1560
$1 * a + 1 * b > 2$	966	19	20359493
$a <= 0$	198	10	120.0

В табл. 1 и 2 приводится количество узлов N^{φ}, сгенерированных при выполнении алгоритма, размер *lean* и время выполнения алгоритма в миллисекундах. Табл. 1 демонстрирует лучшую производительность для булевых формул, например $(a \wedge b)$. Для модальных формул, например, $\langle 1 \rangle \varphi$ соответствующее время выполнения меньше 1 секунды, но по мере увеличения числа модальностей время выполнения также увеличивается (см., например, формулу $\langle 1 \rangle \langle 1 \rangle a$). В случае формул с арифметическими ограничениями эти ограничения проверяются в деревьях на более низком уровне, то есть ограничиваются дочерние узлы. Требуется построение более глубоких деревьев Фишера-Ладнера, что вызывает увеличение времени выполнения. Алгоритмы выполнимости, основанные на явном построении моделей, такие как деревья Фишера-Ладнера, обычно работают плохо, потому что при наличии небулевых противоречивых условий, как правило, требуется построение всех возможных древовидных моделей. Это видно из табл. 2.

Табл. 2. Невыполнимые формулы
Table 2. Unsatisfiable Formulas

Формула	N^{φ}	*Lean*	Время (мс)
¬	12	5	3631
$\neg a \wedge a$	36	5	204811
$(\neg a \wedge a) \vee (\neg b \wedge b)$	36	7	94799
$(a \wedge \neg a) \vee (b \wedge \neg b) \vee (c \wedge \neg c)$	45	8	433768
$\langle 1 \rangle (\neg a \wedge a)$	72	7	1129042
$(1 * a > 0) \wedge (1 * a <= 0)$	19810	10	12812479

5. Логические рассуждения о запросах

Язык путевых выражений XML (XML Path Language, XPath) – это язык запросов для полуструктурированных данных, стандартизованный W3C [16]. В этом разделе мы покажем представление навигационного ядра XPath, расширенного арифметическими ограничениями, в терминах формул μ-исчисления. Вот синтаксис XPath с арифметическими ограничениями.

$$P ::= \alpha :: p \mid P/P \mid P\,[Q],$$
$$Q ::= A > b \mid P \mid Q \vee Q \mid \neg Q,$$
$$A ::= children :: p \mid A + A,$$

где α – направление навигации – *непосредственный потомок, непосредственный предок, следующий брат, предыдущий брат, потомок или предок;* b – положительное целое число. Формальная семантика этой версии XPath с арифметическими ограничениями представлена в работе [3].

Выражения XPath могут быть записаны в терминах формул μ-исчисления. Например $children :: p_1[children :: p_2 > 5]$ можно записать следующим образом: $(p \wedge \langle 3 \rangle >) \wedge (p_2 > 5)$. В работе [3] описаны правила преобразования выражений XPath в формулы μ-исчисления.

К числу общих проблем области логических рассуждений о запросах относятся проблема пустоты результатов запроса (emptiness), проблема включения результатов одного запроса во множество результатов другого запроса (containment) и проблема эквивалентности запросов (equivalence). Проблема пустоты заключается в том, что требуется определить, является ли данный запрос пустым в какой-либо базе данных (модели). Запрос P_1 включается в запрос P_2 тогда и только тогда, когда множество результатов P_1 содержится в множестве результатов P_2 в любой модели. Два запроса эквивалентны, если каждый из них включает другой запрос. Решение этих проблем для языка XPath на основе проверки выполнимости формул μ-исчисления обеспечивает следующая теорема.

Теорема 2 ([3]).

Для заданных выражений XPath P, P_1 и P_2

- P пусто тогда и только тогда, когда $F(P, \top)$ выполнима и
- P_1содержится в P_2 тогда и только тогда, когда $F(P_1, \top) \wedge \neg F(P_2, \top)$ невыполнимо.

Здесь $F(P, \top)$ – формула μ-исчисления, представляющая запрос P.

В табл. 3 и 4 приведены результаты применения алгоритма выполнимости для обнаружения свойств запросов XPath.

Табл. 3. Непустые запросы XPath
Table 3. Non-empty XPath queries

Формула	N^{φ}	*Lean*	Время (мс)
a	8	144	6
$\top$	7	48	7
$\downarrow : a$	9	240	1667
$\uparrow : a$	10	406	1048
$\uparrow : *$	9	192	147
$\downarrow : *$	8	80	212
$\downarrow^* : *$	9	40	335
$\downarrow^* : a$	10	336	784
$\downarrow^* : a \cup \downarrow^* : b$	11	1008	8968
$a/\downarrow : b$	8	1008	7
$a[\downarrow : b]$	11	144	6158
$a[\downarrow : *]$	10	1344	1122
$a[\downarrow : b > 0]$	13	1932	490488
$a[\downarrow : b + \downarrow : c > 0]$	16	8460	77116123
$\downarrow a/\downarrow : a$	12	1232	13143385

Табл. 4. Пустые запросы XPath
Table 4. Empty XPath queries

Формула	N^{φ}	*Lean*	Время (мс)
$a[\neg a]$	8	144	343387
$a[\neg \top]$	8	144	390785
$a[\neg \top]$	9	240	2368813
$a[\neg a]$	9	204	1815198
$\top[\neg \downarrow : a] \cap \top[\downarrow : a]$	8	144	336591
$a[\downarrow : b \wedge \neg \downarrow : b]$	9	336	4206084

5. Выводы

В этой работе мы описали, как и в [3], алгоритм выполнимости для μ-исчисления, расширенного обратными модальностями, номиналами и арифметическими ограничениями Пресбургера. Алгоритм основан на построении моделей Фишера-Ладнера восходящим образом.

Известно, что вычислительная сложность μ-исчисления относится к категории EXPTIME, поэтому непосредственная реализация невозможна. Мы предложили битово-векторное представление узлов в деревьях Фишера-Ладнера. Это позволило справиться с экспоненциальным количеством узлов в памяти. Насколько нам известно, это первая реализация алгоритма принятия решений для μ-исчисления с арифметическими ограничениями.

Также мы сообщили о нескольких экспериментах. Удалось проверить некоторые формулы даже с арифметическими ограничениями. Однако сочетание неподвижных точек, формул с глубокой модальностью и арифметических ограничений по-прежнему выходит за рамки наших возможностей. В перспективе нас интересуют более эффективные конструкции деревьев Фишера-Ладнера [13].

В этой статье мы также говорим о представлении запросов XPath формулами μ-исчисления. Это позволяет решать проблемы логических рассуждений об XPath, такие как пустота, включение и эквивалентность, на основе алгоритма выполнимости μ-исчисления. В [3] мы показали, что фрагмент языка XPath с арифметическими ограничениями является EXPTIME-разрешимым, и нам неизвестен никакой другой фреймворк логических рассуждений для этого фрагмента языка XPath.

Мы также описали некоторые эксперименты с определением пустоты XPath-запросов.

Мы будем продолжать работать над формализацией XML и SXML-схем в терминах μ-исчисления, чтобы гарантировать согласованность этих спецификаций, поскольку из-за количества элементов в них может иметься некоторая несогласованность [17]. Кроме того, мы будем использовать инструменты машинного обучения в промежуточном процессе оптимизации в алгоритме, представленном в этой статье, чтобы использовать его для решения проблем на основе логического вывода в контекстно-зависимых системах [18]. Авторы [19] представили методы машинного обучения и продемонстрировали хорошие результаты. С помощью машинного обучения они проанализировали информацию в Интернете с целью выявления возможных терактов. Иные интересы исследователей касаются верификации контекстно-зависимых систем. В частности, сложной задачей оказались моделирование и верификация временных и количественных свойств этих систем. Мы считаем, что μ-исчисление с арифметическими ограничениями можно использовать в качестве языка спецификации. Соответствующий эффективный алгоритм выполнимости позволил бы осуществить практическую верификацию контекстно-зависимых систем.

Список литературы / References

[1]. P. A. Bonatti, C. Lutz, A. Murano, and M. Y. Vardi. The complexity of enriched mu-calculi. Logical Methods in Computer Science, vol. 4, no. 3, 2008, pp. 1-27.

[2]. E. Bárcenas and J. Lavalle. Global numerical constraints on trees. Logical Methods in Computer Science, vol. 10, no. 2, 2014, pp. 1-28.

[3]. E. Bárcenas, E. Benítez-Guerrero, J. Lavalle, and G. Molero-Castillo. Presburger constraints on tree. Computación y Sistemas, vol. 24, no. 1, 2020, pp. 281-303.

[4]. Y. Limón, E. Bárcenas, E. Benítez-Guerrero, G. Molero-Castillo, and A. Velázquez-Mena. A satisfiability algorithm for the mu-calculus for trees with presburger constraint. In Proc. of the 7th International Conference in Software Engineering Research and Innovation (CONISOFT), 2019, pp. 72-79.

[5]. S. Demri and D. Lugiez. Complexity of modal logics with Presburger constraints. Journal of Applied Logic, vol. 8, no. 3, 2010, pp. 233–252.

[6]. H. Seidl, T. Schwentick, and A. Muscholl. Counting in trees. In Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas, vol. 2. Amsterdam University Press, 2008, pp. 575-612.

[7]. H. Seidl, T. Schwentick, and A. Muscholl. Numerical document queries. In Proc. of the 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, 2003, pp. 155-166.

[8]. L. de Moura and N. Bjørner. Z3: An efficient SMT solver. Lecture Notes in Computer Science, vol. 4963, 2008, pp. 337-340.

[9]. K. Hoder, N. Bjørner, and L. de Moura. μz – an efficient engine for fixed points with constraints. Lecture Notes in Computer Science, vol. 6806, 2011, pp. 457-462.

[10]. W. Conradie, Y. Fomatati, A. Palmigiano, and S. Sourabh. Algorithmic correspondence for intuitionistic modal mu-calculus. Theoretical Computer Science, vol. 564, 2015, pp. 30-62.

[11]. M. Biehl, N. Klarlund, and T. Rauhe. Mona: Decidable arithmetic in practice. Lecture Notes in Computer Science, vol. 1135, 1996, pp. 459-462.

[12]. P. Genevès, N. Layaïda, and A. Schmitt. Efficient static analysis of XML paths and types. In Proc. of the ACM SIGPLAN Conference on Programming Language Design and Implementation, 2007, pp. 342-351.

[13]. Y. Limón, E. Bárcenas, E. Benítez-Guerrero, and M. A. M. Nieto. Depth-first reasoning on trees. Computación y Sistemas, vol. 22, no. 1, 2018, pp. 189-201.

[14]. M. Junedi, P. Genevès, and N. Layaïda. XML query-update independence analysis revisited. In Proc. of the ACM Symposium on Document Engineering, 2012, pp. 95-98.

[15]. M. J. Fischer and R. E. Ladner. Propositional modal logic of programs. In Proc. of the ACM Symposium on Theory of Computing, 1977, pp. 286-294.

[16]. The World Wide Web Consortium (W3C), XPath 2.0 W3C Recommendation, 2010.

[17]. К.Ю. Лисовский. Разработка XML-приложений на языке Scheme. Программирование, том 28, no. 4, 2002 г., стр. 20-32 / K.Yu. Lisovsky. XML applications development in Scheme. Programming and Computer Software, vol. 28, no. 4, 2002, pp. 197–206.

[18]. Y. Limón, E. Bárcenas, E. Benítez-Guerrero, and G. Molero. On the consistency of context-aware systems. Journal of Intelligent and Fuzzy Systems, vol. 34, no. 5, 2018, pp. 3373–3383.

[19]. И.В. Машечкин, М.И. Петровский, Д.В. Царев, М.Н. Чикунов. Методы машинного обучения для задачи обнаружения и мониторинга экстремистской информации в сети Интернет. Программирование, том 45, no. 3, 2019 г., стр. 18-37 / I.V. Mashechkin, M.I. Petrovskiy, D.V. Tsarev, and M.N. Chikunov. Machine learning methods for detecting and monitoring extremist information on the Internet. Programming and Computer Software, vol. 45, no. 3, 2019, pp. 99–115.

Информация об авторах / Information about authors

Йенсен ЛИМОН-ПРИЕГО – кандидат наук. Научные интересы: модальные логики, логический вывод.

Yensen LIMÓN-PRIEGO, Ph. D. Research interests: Modal Logic, Automated Reasoning.

Исмаэль Эверардо БАРСЕНАС-ПАТИНЬО – кандидат наук, доцент. Научные интересы: искусственный интеллект, логический вывод, формальные методы.

Ismael Everardo BÁRCENAS-PATIÑO, Ph. D. in Computer Science, Assistant Professor. Research interests: Artificial Intelligence, Automated Reasoning, Formal Methods.

Эдгард Иван БЕНЕТЕС-ГЕРРЕРО – кандидат наук, профессор. Область научных интересов: искусственный интеллект, взаимодействие человека с компьютером, коллективная работа с использованием компьютеров, системы баз данных.

Edgard Iván BENÍTEZ-GUERRERO, Ph. D. in Computer Science, Full-time Professor. Research interests: Artificial intelligence, human computer interaction, Computer-Supported Cooperative Work, database systems.

Гильермо Хильберто МОЛЕРО-КАСТИЛЬО – кандидат наук, доцент. Научные интересы: искусственный интеллект, наука о данных, интеллектуальный анализ данных, машинное обучение.

Guillermo Gilberto MOLERO-CASTILLO, Ph.D. in Information Technologies, Associate Professor. Research interests: Artificial Intelligence, Data Science, Data Mining, Machine Learning.

Алехандро ВЕЛАСКЕС-МЕНА – магистр, доцент. Область научных интересов: сетевые вычисления, туманные вычисления, распределенные вычисления.

Alejandro VELAZQUEZ-MENA, Master of Science, Associate Professor. Research interests: Network Computing, Fog Computing, Distributed Computing.