DOI: 10.15514/ISPRAS-2019-31(2)-9

Ориентированное на данные планирование с применением отказоустойчивого метода динамической кластеризации для поддержки потоков научных работ в облаках

З. Ахмад, ORCID: 0000-0002-4787-1511 <zulfiqarahmad@hu.edu.pk>
А.И. Джехангири, ORCID: 0000-0001-5920-433X <ali_imran@hu.edu.pk>
М. Ифтихар, ORCID: 0000-0002-7053-2040 <mehreeniftikhar@hu.edu.pk>
А.И. Умар, ORCID: 0000-0002-4148-5062 <arifiqbalumar@yahoo.com>
И. Афзал, ORCID: 0000-0002-3565-1319 <ibrar@hu.edu.pk>

Факультет информационных технологий, Университет Хазары, Мансехра, Пакистан

Аннотация. Облачные вычисления – одна из наиболее распространенных парадигм параллельных и распределенных вычислений. Они используются для поддержки огромного количества научных и бизнес-приложений. В частности, на основе облачных вычислений могут выполняться крупномасштабные научные приложения, которые организованы как потоки научных работ. Потоки научных работ являются приложениями, интенсивно использующими данные, поскольку один поток научных работ может состоять из сотен тысяч задач. Дополнительные затруднения могут вызываться сбоями при выполнении задач, ограничениями по срокам, бюджетными ограничениями и неправильным управлением задачами. Поэтому обеспечение отказоустойчивых методов с использованием ориентированного на данные планирования является важным подходом для поддержки выполнения потоков научных работ в облачных средах. В этой статье мы представляем усовершенствованный механизм планирования, ориентированного на данные, с использованием отказоустойчивой техники динамической кластеризации (EDS-DC) подходом для поддержки выполнения потоков научных работ в облачных средах. Для оценки эффективности EDS-DC, мы сравниваем его результаты с тремя хорошо известными политиками эвристического планирования: (a) MCT-DC, (b) Max-min-DC и (c) Min-min-DC. В качестве примера мы рассматриваем поток научных работ CyberShake, потому что он обладает большинством характеристик потоков научных работ, таких как интеграция, дезинтеграция, параллелизм и конвейеризация. Результаты показывают, что EDS-DC позволил сократить время цикла обработки на 10,9% по сравнению с MCT-DC, на 13,7% по сравнению с Max-min-DC и на 6,4% по сравнению с политикой планирования Min-min-DC. Аналогично, EDS-DC позволил снизить стоимость на 4% по сравнению с MCT-DC, на 5,6% по сравнению с Max-min-DC и на 1,5% по сравнению с политиками планирования Min-min-DC. При использовании EDS-DC по отношению к временным и стоимостным ограничениям не нарушается SLA, в то время как оно нарушается несколько раз при применении политик планирования MCT-DC, Max-min-DC и Min-min-DC.

Ключевые слова: потоки научных работ; отказоустойчивость; планирование потоков работ; cybershake; ориентация на данные

Для цитирования: Ахмад З., Джехангири А.И., Ифтихар М., Умер А.И., Афзал И. Ориентированное на данные планирование с применением отказоустойчивого метода динамической кластеризации для поддержки потоков научных работ в облаках. Труды ИСП РАН, том 31, вып. 2, 2019 г., стр. 121-136. DOI: 10.15514/ISPRAS-2019-31(2)-9

Data-Oriented scheduling with Dynamic-Clustering fault-tolerant technique for Scientific Workflows in Clouds

Z. Ahmad, ORCID: 0000-0002-4787-1511 <zulfiqarahmad@hu.edu.pk>
A.I. Jehangiri, ORCID: 0000-0001-5920-433X <ali_imran@hu.edu.pk>
M. Iftikhar, ORCID: 0000-0002-7053-2040 <mehreeniftikhar@hu.edu.pk>
A.I. Umer, ORCID: 0000-0002-4148-5062 <arifiqbalumar@yahoo.com>
I. Afzal, ORCID: 0000-0002-3565-1319 <ibrar@hu.edu.pk>
Department of Information Technology, Hazara University, Mansehra, KPK, Pakistan

Abstract. Cloud computing is one of the most prominent parallel and distributed computing paradigm. It is used for providing solution to a huge number of scientific and business applications. Large scale scientific applications which are structured as scientific workflows are evaluated through cloud computing. Scientific workflows are data-intensive applications, as a single scientific workflow may consist of hundred thousands of tasks. Task failures, deadline constraints, budget constraints and improper management of tasks can also instigate inconvenience. Therefore, provision of fault-tolerant techniques with data-oriented scheduling is an important approach for execution of scientific workflows in Cloud computing. Accordingly, we have presented enhanced data-oriented scheduling with Dynamic-clustering fault-tolerant technique (EDS-DC) for execution of scientific workflows in Cloud computing. We have presented data-oriented scheduling as a proposed scheduling technique. We have also equipped EDS-DC with Dynamic-clustering fault-tolerant technique. To know the effectiveness of EDS-DC, we compared its results with three well-known enhanced heuristic scheduling policies referred to as: (a) MCT-DC, (b) Max-min-DC, and (c) Min-min-DC. We considered scientific workflow of CyberShake as a case study, because it contains most of the characteristics of scientific workflows such as integration, disintegration, parallelism, and pipelining. The results show that EDS-DC reduced make-span of 10.9% as compared to MCT-DC, 13.7% as compared to Max-min-DC, and 6.4% as compared to Min-min-DC scheduling policies. Similarly, EDS-DC reduced the cost of 4% as compared to MCT-DC, 5.6% as compared to Max-min-DC, and 1.5% as compared to Min-min-DC scheduling policies. These results in respect of make-span and cost are highly significant for EDS-DC as compared with above referred three scheduling policies. The SLA is not violated for EDS-DC in respect of time and cost constraints, while it is violated number of times for MCT-DC, Max-min-DC, and Min-min-DC scheduling techniques.

Keywords: scientific workflows; fault-tolerant; workflows scheduling; cybershake; data-oriented

For citation: Ahmad Z., Jehangiri A.I., Iftikhar M., Umer A.I., Afzal I. Data-Oriented scheduling with Dynamic-Clustering fault-tolerant technique for Scientific Workflows in Clouds. Trudy ISP RAN/Proc. ISP RAS, vol. 31, issue 2, 2019. pp. 121-136 (in Russian). DOI: 10.15514/ISPRAS-2019-31(2)-9

1. Введение

Облачные вычисления – это универсальная и масштабная парадигма распределенных вычислений. Облачная среда предполагает наличие пула настраиваемых и реконфигурируемых, виртуализированных, абстрагированных и динамически доступных вычислительных услуг и ресурсов [35]. Услуги и ресурсы, предоставляемые облачными вычислениями в подписных средах, формируются в форме: (a) серверов, (b) хранилищ данных, (c) средств обработки и (d) приложений [1]. Кроме того, облачные сервисы и ресурсы обладают высоким уровнем масштабируемости и предоставляются внешним клиентам с архитектурой, включающих три сегментов, а именно: инфраструктура как услуга (Infrastructure as a Service, IaaS), платформа как услуга (Platform as a Service, PaaS) и программное обеспечение как услуга (Software as a Service, SaaS). Многие организации переходят на облачные сервисы для повышения своей эффективности [2] [3] [36] [39].

Упрощенно можно считать, что облачные сервисы и ресурсы ориентированы на подержку деловых и научных приложений. Научные приложения, связанные с объемными вычислениями и хранимыми данными, организуются как потоки научных работ [4]. В потоках научных работ для решения каждой задачи требуется значительный объем

вычислений и памяти, и, таким образом, сбой даже одной задачи наносит значительный ущерб всей системе [5] [38] [42].

Научные приложения, в частности, относятся к областям науки о землетрясениях, астрономии, биологии и гравитационной физики [6]. CyberShake [7] является потоком научных работ в реальном времени, связанным с сейсмологией (наукой о землетрясениях). В потоке научных работ CyberShake сейсмические опасности для конкретного местоположения количественно оцениваются сейсмологом с помощью вероятностного анализа сейсмической опасности (probabilistic seismic hazard analysis, PSHA). В PSHA предусмотрен механизм для оценки вероятности уровней движения грунта при землетрясении в конкретном месте и в течение оговоренного периода времени с применением меры интенсивности (intensity measure, IM) – пиковая скорость или пиковое ускорение движения грунта. Эти вероятностные меры полезны для инженеров-строителей, градостроителей и страховых агентств, поскольку они влияют на эффективность затрат в миллиарды долларов каждый год. CyberShake требует разумного хранения данных и соответствующих вычислительных ресурсов [8]. Облачные вычисления обеспечивают повсеместную доступность и неограниченность ресурсов с приемлемой ценой в среде, в которой ресурсы получаются и освобождаются динамически. Поэтому для поддержки потоков научных работ облачная среда является одной из лучших платформ [9] [10] [11].

Когда поток научных работ, такой как CyberShake, выполняется в реальном времени, требуется решить несколько проблем.

- Предположим, что в потоке научных работ имеются задачи $A_1, A_2, A_3, \ldots, A_n$, которые выполняются на нескольких уровнях $B_1, B_2, B_3, \ldots, B_n$ на ресурсах $C_1, C_2, C_3, \ldots, C_n$.
- Несколько задач из $A_1, A_2, A_3, \ldots, A_n$ выполняются в критическом режиме, или на критическом уровне, и их отказ делает выполнение всего потока работ бесполезным; следовательно, требуется отказоустойчивая техника.
- На каждом уровне существует несколько задач, и иногда у них имеются одинаковые требования к услугам и ресурсам, и поэтому для такого выполнения очень полезен отказоустойчивый механизм кластеризации [40].
- Предположим, что для каждой задачи имеется заранее установленное время передачи данных каждому ресурсу. В потоке научных работ есть несколько задач из $A_1, A_2, A_3, \ldots, A_n$ с различными потребностями в ресурсах из $C_1, C_2, C_3, \ldots, C_n$ на нескольких уровнях $B_1, B_2, B_3, \ldots, B_n$. Таким образом, требуется эффективная политика планирования, ориентированного на данные.

В своей работе мы последовательно исследуем и решаем отмеченные проблемы применительно к выполнения потоков научных работ в облачных средах. Для этого мы усовершенствовали ориентированное на данные планирование потоков научных работ с применением отказоустойчивого метода динамической. В качестве параметров оценки мы рассматривали время, стоимость, сроки, бюджет и нарушение SLA [12]. Работа нацелена на то, чтобы ответить на основной вопрос, который волнует исследователей: насколько будет улучшена производительность, если мы будем совместно использовать ориентированное на данные планирование и отказоустойчивую технику для организации потоков научных работ в облаках?

Основные результаты нашей работы состоят в следующем.

- Мы предложили усовершенствованное ориентированное на данные планирование потоков научных работ в облаках с применением отказоустойчивого метода динамической кластеризации (Enhanced Data-oriented Scheduling with Dynamic-Clustering fault-tolerant technique, EDS-DC) [13].
- Чтобы оценить эффективность EDS-DC, мы сравнили его результаты с результатами трех хорошо известных политик планирования: (a) Minimum-Completion-Time-Dynamic-Clustering (MCT-DC) [16], (b) Maximum-minimum-Dynamic-Clustering (Max-min-DC) [17] и (c) Minimum-minimum-Dynamic-clustering (Min-min-DC) [17].

- Для оценки эффективности EDS-DC, мы провели симуляцию с использованием WorkflowSim [18] и предоставили патч для выполнения этой симуляции.
- В качестве примера мы выполнили поток научных работ реального времени CyberShake [7] с 30, 50, 100 и 1000 задачами. Результаты моделирования показывают, что наш подход превосходит существующий решения.

Оставшаяся часть статьи организована следующим образом: обзор родственных работ представлен в разд. 2. Модель и схема решения описываются в разд. 3, эксперименты и результаты обсуждаются в разд. 4, и, наконец, разд. 5 завершает работу.

2. Родственные работы

Мы подробно изучили родственные работы, касающиеся выполнения потоков научных работ, отказоустойчивых методов и механизмов планирования потоков научных работ. Мы также обнаружили методы отказоустойчивой кластеризации и различные типы политики планирования, ориентированного на данные, применяемые именно в потоках научных работ.

С самого начала было проведено исследование пяти реалистичных потоках научных работ из различных научных приложений [6]. Были изучены структуры, данные и вычислительные требования каждого из этих потоках научных работ. На рис. 1 представлена структура небольшого экземпляра каждого потока научных работ, Рисунок показывает, что основные компоненты потоков работ обладают различными структурными и функциональными свойства: применение конвейера, параллелизм по данным, агрегирование данных, распределение, перераспределение данных и их композиция.

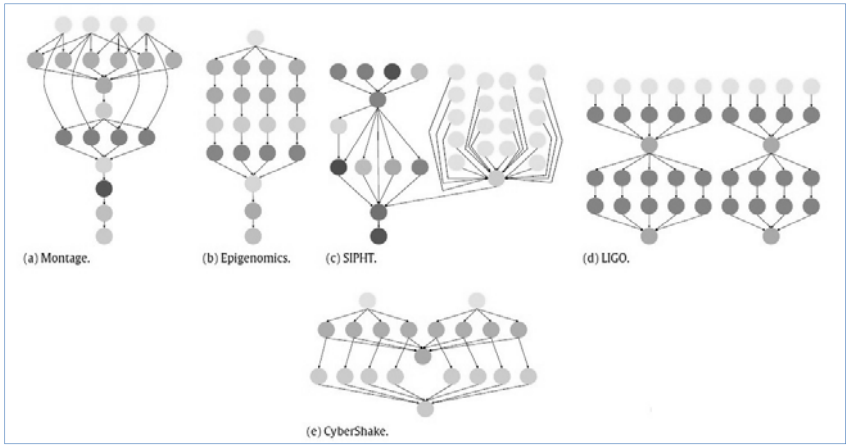

Рис. 1. Общее представление пяти реалистичных потоков научных работ
Fig. 1: An overview of THE five realistic scientific workflows

Большое количество научных приложений, таких как картирование генома, моделирование в физике высоких энергий и моделирование климата [19], являются приложениями, интенсивно использующими данные, и для них требуется планирование с учетом данных [14] [43]. Stork [19] является планировщиком, учитывающим данные, он был специально разработан для учета передачи данных, выделения памяти, удаления данных, освобождения памяти и регистрации метаданных, связанных с традиционными методами планирования. Для учета времени завершения рабочего процесса и использования ресурсов в [20] был предложен метод планирования Adaptive Data Aware Scheduling (ADAS).

Планировщик является основным компонентом, поддерживающим выполнения потоков работ, и его возможностями нельзя пренебрегать на любой стадии выполнения. Поэтому в [21] был предложен алгоритм планирования для выполнения потока работ под названием

«Отказоустойчивое планирование потоков работ с использованием спот-инстансов в облаках» (Fault-Tolerant Workflow Scheduling, FTWS), который является устойчивым к изменениям производительности.

У облачных сред имеются три основные особенности: (а) предоставление ресурсов по требованию, (б) согласованная пропускная способность между сервиса, предоставляемыми облачными поставщиками услуг и (в) модель ценообразования с оплатой по мере использования в различных коммерческих облачных средах [22]. Поэтому с учетом этих особенностей в [9] были предложены два новых алгоритма планирования потоков работ для облаков категории IaaS: IaaS Cloud-Partial Critical Path (IC-PCP) и IaaS Cloud-Partial Critical Path with Deadline Distribution (IC-PCPD2). Эти алгоритмы предназначены для создания расписаний, которые удовлетворяют заданным пользователями срокам, а также сводят к минимуму стоимость выполнения.

Три алгоритма планирования для групп потоков работ были предложены в [23]: Dynamic Provisioning Dynamic Scheduling (DPDS), Workflow Aware DPDS(WA-DPDS) и Static Provisioning Static Scheduling (SPSS). В этих алгоритмах учитываются только ограничения на время выполнения и расходы. Имеется несколько эвристических подходов, таких как (a) Minimum Completion Time (MCT) [24], (b) Maximum-minimum (Max-min) [25] и (c) Minimum-minimum (Min-min) [26], которые также использовались для планирования независимых задачи при выполнении потоков научных работ [16].

Что касается отказоустойчивых методов, эффективным и простым отказоустойчивым методом является повторное выполнение задач [27]: аварийно завершенная задача повторно выполняется либо на тех же ресурсах, либо на других доступных ресурсах [28]. Механизм кластеризации Fault-Tolerant Clustering (FTC) для потоков научных работ был представлен в [15]. В работе описаны три отказоустойчивых метода кластеризации: Dynamic-Clustering (DC), Selective Re-Clustering (SR) и Dynamic Re-Clustering (DR). Эти алгоритмы несовершенны по параметрам стоимости и времени.

The brief comparison of literature review is shown in Table 1.

Краткое сравнение особенностей рассмотренных подходов приведено в таблице 1.

Табл. 1. Сравнение результатов родственных работ
Table 1: Comparison of Related Work

Ссылка	Политика планирования	Механизм отказоустойчивости	Управление ресурсами	Качество обслуживания (QoS)	
				Время	Стоимость
[29]	Pegasus WMS	✖	✓	✖	✖
[20]	ADAS	✖	✓	✓	✖
[21]	FTWS	Контрольные точки	✓	✖	✓
[15]	✖	FTC	✓	✖	✖
[30]	WSA	✖	✓	✓	✖
[31]	BTC	✖	✓	✓	✖
[9]	IC-PCP & IC-PCPD2	✖	✓	✓	✓
[23]	DPDS, WA-DPDS & SPSS	✖	✓	✓	✓
[27]	✖	Перезапуск задач	✖	✖	✖

Ссылка	Особенности	Ограничения
[29]	Обеспечивает структуру WMS	Отсутствует механизм QoS
[20]	Улучшает показатель времени завершения потока работ	Отсутствует метод отказоустойчивости
[21]	Используются две ценовые модели: спот-инстансы и инстансы по запросу	Отсутствует метод сокращения времени цикла обработки как показателя QoS
[15]	Обеспечиваются три метода поддержки отказоустойчивости: DC, SRC & DRC	
[30]	Используется механизм набора маркеров (token bucket) для сокращения времени выполнения	Отсутствует механизм отказоустойчивости
[31]	Обеспечивается метод балансировки нагрузки	Накладные расходы на анализ показателей и зависимостей
[9]	Задачи планируются с учетом ограничений времени, задаваемых пользователями	Работает только в пределах IaaS
[23]	Обеспечивает ресурсы для групп потоков работ	Пригоден только для групп потоков работ
[27]	Обеспечивает гибридный механизм поддержки отказоустойчивости	Отсутствует метод учета QoS

3. Схема и прототип системы

Мы разработали усовершенствованный планировщик потоков научных работ, ориентированный на данные и использующий отказоустойчивую технику динамической кластеризации [15] (EDS-DC). Планирование в EDS-DC ориентировано на данные [14], но в качестве параметров оценки мы используем (a) время, (b) стоимость, (c) бюджет, (d) крайний срок и (d) нарушение SLA [32].

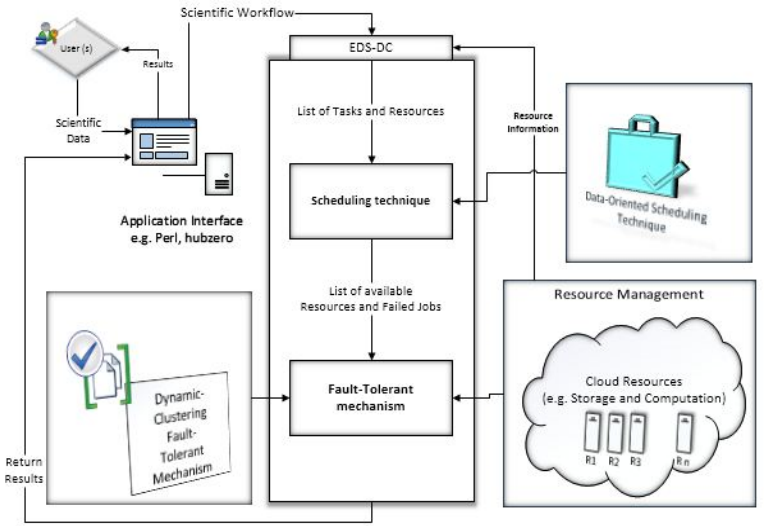

Рис. 2. Архитектура прототипа EDS-DC
Fig 2: An overview of EDS-DC Model

Чтобы оценить эффективность EDS-DC, мы сравнили его результаты с тремя хорошо известными эвристическими политиками планирования: Minimum Completion Time (MCT) [16], Max-min [17] и Min-min [17]. Для оценки эффективности EDS-DC мы провели моделирование с использованием WorkflowSim [18]. В качестве примера мы выполнили поток научных работ реального времени CyberShake [7] с 30, 50, 100 и 1000 задачами. Результаты моделирования показывают, что EDS-DC превосходит существующие решения.

Общая модель показана на рис. 2. Пользователь передает научные данные через интерфейс приложения, например, Perl или hubzero, и модель EDS-DC обрабатывает их с помощью политики планирования, ориентированной на данные, и отказоустойчивой техники динамической кластеризации. Ресурсы и услуги получаются из облачной инфраструктуры в форме IaaS. Наконец, результаты оценки возвращаются пользователю через интерфейс приложения.

3.1 Ориентированное на данные планирование в EDS-DC

Планирование потоков научных работ в EDS-DC ориентировано на данные [14] [34] [41] [44]. В планировании, ориентированном на данные, каждая задача потока научных работ назначается доступному ресурсу, для которого время передачи данных минимально. Алгоритм 1 показывает процедуру планирования, ориентированного на данные, в EDS-DC. Планировщик получает список задач $L(Tn)$, список ресурсов $L(Rn)$ и возвращает отображение списка задач $M(Tn)$ ресурсам $M(Rn)$. Каждая задача отображается на наиболее подходящий доступный ресурс с минимальным временем передачи данных.

```
      Input: L(Tn, Rn) – List of tasks and available resources
      Output: M(Tn, Rn) – Mapped list of tasks to the resources
 1.   procedure Data-Oriented Scheduling (L(Tn, Rn))
 2.      Queue ← Tn
 3.      Ri ← Rn
 4.      M(Tn, Rn) ← 0 ▷ Mapped list of tasks to resources
                         (initially empty)
 5.      while (Queut is not empty) do
 6.         Ti ← Delete each task from Queue
 7.         for all available resources (Rn) do
 8.             find resource Ri with minimum data transfer time
 9.             for task Ti
10.         end for
11.         M(Tn, Rn) ← M(Tn, Rn) + M(Ti, Ri) ▷ Mapped and submit to
                                                 list
12       end while
13.      return M(Tn, Rn)                  ▷ The output is M(Tn, Rn)
14.   end procedure
```

Алгоритм 1. Ориентированное на данные планирование в EDS-DC
Algorithm 1. Data-oriented scheduling in EDS-DC

3.2 Отказоустойчивый метод динамической кластеризации в EDS-DC

Алгоритм 2 демонстрирует процедуру Dynamic-Clustering [15] [34] [41] [44], используемую для обеспечения отказоустойчивости. Поскольку в потоках научных работ одна или несколько связанных задач объединяются в одно задание, метод отказоустойчивой динамической кластеризации выбирает задания, которые содержат одну или большее число аварийно завершившихся задач, динамически разбивает их на k кластеров задач и повторно выполняет. Входными данными является списки ресурсов $L(Rn)$ и неудачных заданий $L(Fj)$. Алгоритм возвращает список невыполненных заданий $M(Fj)$, сопоставленных требуемым ресурсам $M(Rn)$.

```
      Input: L(Rn, Fi) – List of available resources and failed jobs
      Output: M(Fi, Rn) – Mapped list of failed jobs to resources
 1.   procedure Dynamic-Clustering (L(Rn, Fi))
 2.      Lj ← Fi   ▷ List of failed jobs
 3.      LR ← Rn   ▷ List of available resources
 4.      M(Fi, Rn) ← 0 ▷ Mapped list failed jobs to resources
 5.      Split each job from Lj into k clusters of tasks dynamically
 6.      for all clusters (k) and available resources (LR) do
 7.          find resource(s) Ri for cluster ki
 8.          as per data-oriented scheduling algorithm
 9.      end for
10.         M(Fi, Rn) ← M(Fi, Rn) + M(ki, Ri) ▷ Mapped and submit to
                                                 list
13.     return M(Fi, Rn)                  ▷ The output is M(Fi, Rn)
14.   end procedure
```

Алгоритм 2. Отказоустойчвый метод динамической кластеризации в EDS-DC
Algorithm 2. Dynamic-Clusterind Fault-tolerant technique in EDS-DC

4. Эксперименты, результаты и обсуждение

В этом разделе рассматриваются подготовка имитации, моделирование ресурсов и приложений, а затем приводятся результаты и обсуждение.

4.1 Подготовка имитации

In terms of characteristics of resources and specifications of scientific workflows submitted by the user, the detail description of the simulation environment is given below.

Применительно к характеристикам ресурсов и спецификациям потоков научных работ, предоставленным пользователями ниже приводится подробное описание среды симуляции.

4.1.1 Моделирование ресурсов

Для моделирования использовался WorkflowSim [18] [37], «инструментарий для симуляции потоков научных работ». WorkflowSim – это инструмент симуляции потоков работ, используемый для реализации методов планирования и управления потоками, однако предлагаемая политика планирования и отказоустойчивый механизм в WorkflowSim ранее не были реализованы. Поэтому мы реализовали в WorkflowSim Алгоритм 1 (Политика планирования) и Алгоритм 2 (Механизм отказоустойчивости). Использовались общие ресурсы адресного пространства, а также такие характеристики, как стоимость, период обработки, бюджет и крайний срок выполнения.

Остальные технические характеристики приведены в табл. 2.

Табл. 2. Спецификация русурсов, используемых для симуляции
Table 2: The specification of resources used for simulation

Число VM	**Память**	**BW**	**VM**	**Arch**
100	1 GB	1000	Xen	X86
OS	**Стоимость VM $/час**	**Стоимость памяти $/сек**	**Стоимость хранения данных $/сек**	**Стоимость передачи данных $/сек**
Linux	3.0	0.05	0.1	0.1

4.1.2 Моделирование приложений

При моделировании мы считали, что имитируется один пользователь, который запускает поток научных работ реального потока научных работ рабочего процесса CyberShake является то, что CyberShake обладает большинством характеристик потоков научных работ, таких как интеграция, дезинтеграция, параллелизм и конвейеризация.

4.2 Параметры оценки производительности

Ниже подробно рассматриваются параметры оценки производительности с описанием того, как они рассчитываются в наших сценариях.

4.2.1 Make-span

Make-span – это время, требуемое для завершения выполнения пакета задач. В контексте потоков научных работ это общее время, необходимое для выполнения всего потока научных работ [21]. Оно обозначается как *M.S* и вычисляется по формуле (1).

$$M.S = F.T - S.T \quad (1),$$

где *F.T* – время окончания потока научных работ, а *S.T* – время начала потока научных работ.

4.2.2 Крайний срок

Крайний срок – это заранее определенное время окончания выполнения пакета задач. В контексте потоков научных работ заранее задается общее время выполнения всего потока научных работ [17]. Оно обозначается как *D.L* и может быть вычислено по формуле (2).

$$D.L = Comp.T + Comm.Time + Overhead \quad (2)$$

Накладные расходы – это дополнительное время, затрачиваемое на повторное выполнение неудачных заданий/задач.

4.2.3 Стоимость

Стоимость – это бюджет, требуемый для выполнения пакета задач. В контексте потоков научных работ это общий бюджет, необходимый для выполнения всего потока научных работ [21]. Стоимость может быть рассчитан по формуле (3).

$$Cost = Cost\ on\ F.T - Cost\ on\ S.T \quad (3)$$

Кроме того, стоимость выполнение каждой задачи потока научных работ обозначается через $Cost_t$ и может быть вычислена по формуле (4).

$$Cost_t = ProcessingCost + StorageCost + MemoryCost + BandwidthCost \quad (4)$$

4.2.4 Бюджет

Бюджет – это общий объем доступных финансовых ресурсов для выполнения пакета задач [17]. В контексте потоков научных работ это предопределенная стоимость, необходимая для выполнения потоков научных работ целиком. Бюджет может быть рассчитан с помощью формулы (5).

$$Budget = Comp.Cost + Comm.Cost + Overhead \quad (5)$$

Накладные расходы ($Overhead$) – это дополнительные затраты, потребляемые при повторном выполнении неудачных заданий/заданий.

4.2.5 Нарушение SLA

В контексте потоков научных работ нарушением соглашения об уровне обслуживания (SLA Violation) называется ситуация, когда стоимость выполнения потока научных работ превышает размер установленного бюджета или время выполнения выходит за пределы крайнего срока [17]. Формулы (6) и (7) показывают условия, касающиеся нарушения SLA [33].

$$SLAV = SLAV_{I.T} \quad (6)$$

$$SLAV = SLAV_{I.C} \quad (7),$$

где *SLAV* – это нарушение SLA, $SLAV_{I.T}$ – это нарушение SLA из-за увеличения времени выполнения, а $SLAV_{I.T}$ – увеличение стоимости за пределы доступного бюджета.

4.3 Результаты и обсуждение

Мы определили сценарий для масштабного моделирования своего подхода. Мы рассматриваем одного пользователя, который запускает поток научных работ реального времени CyberShake с 30, 50, 100 и 1000 задачами. В EDS-DC мы использовали отказоустойчивую технику динамической кластеризации вместе с ориентированным на данные планированием. Кроме того, мы выполняли тот же поток научных работ реального времени с применением трех хорошо известных эвристических политик планирования MCT, Max-min и Min-min. Затем мы сравнили результаты EDS-DC с результатами этих политик.

4.3.1 Сценарий симуляции

Цель сценария – оценить EDS-DC и сравнить эффективность нашего подхода с результатами использования MCT, Max-min и Min-min. Параметры оценки: время выполнения, стоимость, крайний срок, бюджет и нарушение SLA. Среднее значение рассчитывается по времени выполнения и стоимости, а затем мы анализируются результаты. Количество пользователей – 1, а время моделирования – 24 часа. Общая спецификация сценария приведена в табл. 3.

Табл. 1. Спецификации сценария 1
Table 3: Scenario 1 specifications

Механизм отказоустойчивости	Политика планирования	Потоки научных работ
Динамическая кластеризация	EDS-DC	CyberShake-30
		CyberShake-50
		CyberShake-100
		CyberShake-1000
	MCT	CyberShake-30
		CyberShake -50
		CyberShake -100
		CyberShake -1000
	Max-min	CyberShake-30
		CyberShake -50
		CyberShake -100
		CyberShake -1000
	Min-min	CyberShake-30
		CyberShake -50
		CyberShake -100
		CyberShake -1000

Время выполнения: результаты по времени выполнения для EDS-DC по сравнению с MCT, Max-min и Min-min представлены на рисунке. 3. Результаты показывают, что EDS-DC сократил время выполнения на 10,9% как о сравнению с MCT-DC, на 13,7% по сравнению с Max-min-DC и на 6,4% по сравнению с политикой планирования Min-min-DC. Причина в том, что в планировании EDS-DC время выполнения рассматривается как параметр, ориентированный на данные.

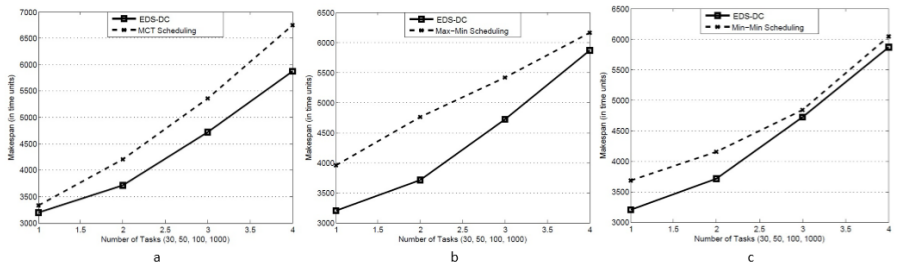

Рис. 3: Сравнение EDS-DC с (a) планированием MCT, (b) планированием Max-Min и (c) планированием Min-Min для CyberShake с использованием метода отказоустойчивости Dynamic-Clustering по времени выполнения

Fig. 3: Comparison of EDS-DC with (a) MCT scheduling, (b) Max-Min scheduling, and (c) Min-Min scheduling for CyberShake in respect of make-span by using Dynamic-Clustering Fault-Tolerant Technique

Стоимость: результаты по стоимости для EDS-DC по сравнению с MCT, Max-min и Min-min планировкой редставлены на рис. 4. Результаты показывает, что EDS-DC позволил снизить стоимость на 4% по сравнению с MCT-DC, на 5,6% по сравнению с Max-min-DC и на 1,5% по сравнению с политикой Min-min-DC. Причина в том, что в планировании EDS-DC мы рассматриваем стоимость как параметр, ориентированный на данные.

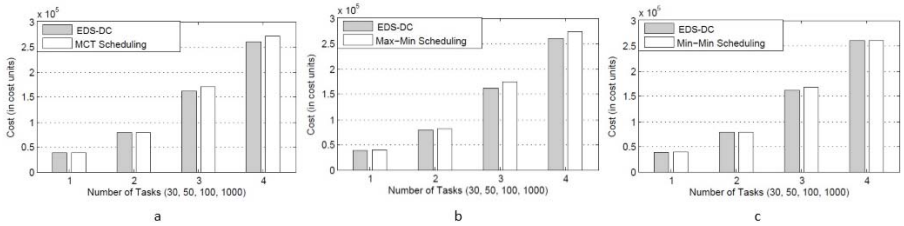

Рис. 4. Сравнение EDS-DC с (a) планированием MCT, (b) планированием Max-Min и (c) планированием Min-Min для CyberShake с использованием метода отказоустойчивости Dynamic-Clustering по стоимости

Fig. 4: Comparison of EDS-DC scheduling with (a) MCT scheduling, (b) Max-Min scheduling, and (c) Min-Min scheduling for CyberShake in respect of cost by using Dynamic-Clustering Fault-Tolerant Technique

Нарушение SLA: результаты в табл. 4 показывают, что при использовании EDS-DC SLA не нарушается ни для временных, ни для стоимостных ограничений для всех четырех потоков работ. При использовании MCT, Max-min и Min-min SLA нарушается четыре, восемь и два раза соответственно. Причина в том, что в планировании EDS-DC мы рассматривали время выполнения, стоимость, бюджет, крайний срок и нарушение SLA как параметры, ориентированные на данные.

Табл. 4. Крайний срок, бюджет и нарушение SLA
Table 4: Deadline, budget and SLA Violation

Поток научных работ	Политика планирования	Время выполнения (сек)	Стоимость (центы)
CyberShake-30	***EDS-DC***	***3197.626***	***38876.576***
	MCT	3335.92	38594.654
	Max-min	3955.87	40389.53
	Min-min	3677.02	40232.162
CyberShake - 50	***EDS-DC***	***3711.602***	***78421.248***
	MCT	4207.002	79516.86
	Max-min	4764.942	81624.304
	Min-min	4152.506	79128.504
CyberShake - 100	***EDS-DC***	***4725.288***	***161065.554***
	MCT	5353.238	170969.164
	Max-min	5413.854	174789.57
	Min-min	4835.968	167245.486
CyberShake - 1000	***EDS-DC***	***5871.774***	***260137.452***
	MCT	6745.066	271745.562
	Max-min	6162.442	273448.45
	Min-min	6041.182	260303.37

Поток научных работ	Крайний срок (сек)	Бюджет (центы)	Нарушение SLA	
			For Time	For Cost
CyberShake-30	3600.00	40000.00	***No***	***No***
			No	No
			Yes	Yes
			Yes	Yes
CyberShake - 50	4500.00	80000.00	***No***	***No***
			No	No
			Yes	Yes
			No	No
CyberShake - 100	5000.00	170000.00	***No***	***No***
			Yes	Yes
			Yes	Yes
			No	No
CyberShake - 1000	6500.00	270000.00	***No***	***No***
			Yes	Yes
			No	Yes
			No	No

5. Заключение

Мы представили EDS-DC для планирования потоков научных работ. EDS-DC – это планировщик, ориентированный на данные и использующий отказоустойчивую технику динамической кластеризации. Мы рассмотрели пример потока научных работ реального времени CyberShake с 30, 50, 100 и 1000 задачами. Чтобы узнать эффективность EDS-DC, мы сравнили его результаты с тремя известными политиками планирования MCT-DC, Max-min-DC и политики Min-min-DC. Результаты по времени выполнения для CyberShake (1180 задач) составляют 17506,29, 19641,23, 20297,11 и 18706,68 секунд для EDS-DC, MCT-DC, Max-min-DC и Min-min-DC соответственно. Результаты по стоимости CyberShake (1180 задач) составляют 538500,8, 560826,2, 570251,9 и 546909,5 цента для политик планирования EDS-DC, MCT-DC, Max-min-DC и Min-min-DC соответственно. Изучение результатов моделирования показывает, что EDS-DC уменьшил время выполнениязон на 10,9% по сравнению с MCT-DC, на 13,7% по сравнению с Max-min-DC и на 6,4% по сравнению с политикой планирования Min-min-DC. Аналогично, EDS-DC снизил стоимость на 4% по сравнению с MCT-DC, на 5,6% по сравнению с Max-min-DC и на 1,5% по сравнению с политикой планирования Min-min-DC. SLA не нарушается для EDS-DC по отношению к временных и стоимостных ограничениям, но нарушается несколько раз при использовании MCT-DC, Max-min-DC и Min-min-DC.

В будущих исследованиях мы рассчитываем разработать основанные на QoS энергосберегающие и ориентированные на данные политики отказоустойчивого планирования для потоков научных работ в облачных средах.

Список литературы / References

[1] J. Shi, J. Luo, F. Dong, J. Zhang, and J. Zhang, “Elastic resource provisioning for scientific workflow scheduling in cloud under budget and deadline constraints,” Cluster Comput., vol. 19, no. 1, pp. 167–182, 2016.

[2] D. Sun, G. Chang, C. Miao, and X. Wang, “Analyzing, modeling and evaluating dynamic adaptive fault tolerance strategies in cloud computing environments,” J. Supercomput., vol. 66, no. 1, pp. 193–228, 2013.

[3] D. Lifka, “XSEDE Cloud Survey Report,” no. September, 2013.

[4] X. Li, J. Song, and B. Huang, “A scientific workflow management system architecture and its scheduling based on cloud service platform for manufacturing big data analytics,” Int. J. Adv. Manuf. Technol., pp. 119–131, 2016.

[5] B. P. Abbott et al, “LIGO: the Laser Interferometer Gravitational-Wave Observatory,” Reports Prog. Phys., vol. 72, no. 7, p. 76901, 2009.

[6] S. Bharathi, E. Deelman, G. Mehta, K. Vahi, A. Chervenak, and M. Su, “Characterization of Scientific Workflows,” in The 3rd Workshop on Workflows in Support of Large Scale Science, 2008.

[7] S. Callaghan, P. Maechling, P. Small, K. Milner, G. Juve, T. H. Jordan, E. Deelman, G. Mehta, K. Vahi, D. Gunter, K. Beattie, and C. Brooks, “Metrics for heterogeneous scientific workflows : A case study of an earthquake science application,” 2011.

[8] S. Callaghan, P. Maechling, E. Deelman, K. Vahi, G. Mehta, K. Milner, R. Graves, E. Field, D. Okaya, D. Gunter, and T. Jordan, “Reducing Time-to-Solution Using Distributed High-Throughput Mega-Workflows – Experiences from SCEC CyberShake,” pp. 151–158, 2008.

[9] S. Abrishami, M. Naghibzadeh, and D. H. J. Epema, “Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Clouds,” Futur. Gener. Comput. Syst., vol. 29, no. 1, pp. 158–169, 2013.

[10] D. Chakraborty, V. V. Mankar, and A. A. Nanavati, “Enabling runtime adaptation of workflows to external events in enterprise environments,” Proc. - 2007 IEEE Int. Conf. Web Serv. ICWS 2007, no. Icws, pp. 1112–1119, 2007.

[11] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, “The cost of doing science on the cloud: The montage example,” 2008 SC - Int. Conf. High Perform. Comput. Networking, Storage Anal. SC 2008, no. November, 2008.

[12] D. Serrano, S. Bouchenak, Y. Kouki, F. A. De Oliveira, T. Ledoux, J. Lejeune, J. Sopena, L. Arantes, and P. Sens, “SLA guarantees for cloud services,” Futur. Gener. Comput. Syst., vol. 54, pp. 233–246, 2016.

[13] J. Choi, T. Adufu, and Y. Kim, “Data-Locality Aware Scientific Workflow Scheduling Methods in HPC Cloud Environments,” Int. J. Parallel Program., vol. 45, no. 5, pp. 1128–1141, 2017.

[14] W. Tang, J. Jenkins, F. Meyer, R. Ross, R. Kettimuthu, L. Winkler, X. Yang, T. Lehman, and N. Desai, “Data-aware resource scheduling for multicloud workflows: A fine-grained simulation approach,” Proc. Int. Conf. Cloud Comput. Technol. Sci. CloudCom, vol. 2015–Febru, no. February, pp. 887–892, 2015.

[15] W. Chen and E. Deelman, “Fault tolerant clustering in scientific workflows,” Proc. - 2012 IEEE 8th World Congr. Serv. Serv. 2012, pp. 9–16, 2012.

[16] B. Santhosh, P. K. A, and D. H. Manjaiah, “Comparative Study of Workflow Scheduling Algorithms in Cloud Computing,” pp. 31–37, 2014.

[17] T. Mathew, “Study and Analysis of Various Task Scheduling Algorithms in the Cloud Computing Environment,” Int. Conf. Adv. Comput. Informatics, pp. 658–664, 2014.

[18] W. Chen and E. Deelman, “WorkflowSim: A toolkit for simulating scientific workflows in distributed environments,” 2012 IEEE 8th Int. Conf. E-Science, e-Science 2012, 2012.

[19] T. Kosar and M. Balman, “A new paradigm: Data-aware scheduling in grid computing,” Futur. Gener. Comput. Syst., vol. 25, no. 4, pp. 406–413, 2009.

[20] L. Zeng, B. Veeravalli, and A. Y. Zomaya, “An integrated task computation and data management scheduling strategy for workflow applications in cloud environments,” J. Netw. Comput. Appl., vol. 50, pp. 39–48, 2015.

[21] D. Poola, K. Ramamohanarao, and R. Buyya, “Fault-tolerant workflow scheduling using spot instances on clouds,” Procedia Comput. Sci., vol. 29, pp. 523–533, 2014.

[22] D. Kumar, G. Baranwal, Z. Raza, and D. P. Vidyarthi, “A systematic study of double auction mechanisms in cloud computing,” J. Syst. Softw., vol. 125, pp. 234–255, 2017.

[23] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, “Algorithms for cost-and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds,” Futur. Gener. Comput. Syst., vol. 48, pp. 1–18, 2015.

[24] X. He, X. Sun, and G. Laszewski, “A QoS Guided Scheduling Algorithm for Grid Computing,” Office, vol. 18, no. 4, pp. 1–15, 2002.

[25] A. M. Madureira and A. B. Definitions, “Ordered Minimum Completion Time Heuristic for Unrelated Parallel-Machines Problems.”

[26] R. J. Priyadarsini, “Performance Evaluation of Min-Min and Max-Min Algorithms for Job Scheduling in Federated Cloud,” vol. 99, no. 18, pp. 47–54, 2014.

[27] K. Qureshi, F. G. Khan, P. Manuel, and B. Nazir, “A hybrid fault tolerance technique in grid computing system,” J. Supercomput., vol. 56, no. 1, pp. 106–128, 2011.

[28] A. Bala and I. Chana, “Fault Tolerance- Challenges , Techniques and Implementation in Cloud Computing,” Int. J. Comput. Sci., vol. 9, no. 1, pp. 288–293, 2012.

[29] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. Ferreira Da Silva, M. Livny, and K. Wenger, “Pegasus, a workflow management system for science automation,” Futur. Gener. Comput. Syst., vol. 46, pp. 17–35, 2015.

[30] R. Tolosana-Calasanz, J. Á. Bañares, C. Pham, and O. F. Rana, “Enforcing QoS in scientific workflow systems enacted over Cloud infrastructures,” J. Comput. Syst. Sci., vol. 78, no. 5, pp. 1300–1315, 2012.

[31] W. Chen, R. Ferreira, E. Deelman, and R. Sakellariou, “Balanced Task Clustering in Scientific Workflows,” pp. 1–8.

[32] A. F. Antonescu and T. Braun, “Simulation of SLA-based VM-scaling algorithms for cloud-distributed applications,” Futur. Gener. Comput. Syst., vol. 54, pp. 260–273, 2016.

[33] S. Mustafa, B. Nazir, A. Hayat, A. ur Rehman Khan, and S. A. Madani, “Resource management in cloud computing: Taxonomy, prospects, and challenges,” Comput. Electr. Eng., vol. 47, p. , 2015.

[34] Barladian, B. Kh, et al. "An efficient mulithreading algorithm for the simulation of global illumination." Programming and Computer Software 43.4 (2017): 217-223.

[35] Fursova, N. I., et al. "A lightweight method for virtual machine introspection." Programming and Computer Software 43.5 (2017): 307-313.

[36] Gusev, A. D., Andrey V. Nasonov, and Andrey S. Krylov. "Fast parallel grid warping-based image sharpening method." Programming and Computer Software 43.4 (2017): 230-233.

[37] Kuplyakov, D., Evgeny Shalnov, and Anton Konushin. "Markov chain Monte Carlo based video tracking algorithm." Programming and Computer Software 43.4 (2017): 224-229.

[38] Massobrio, Renzo, et al. "Towards a cloud computing paradigm for big data analysis in smart cities." Programming and Computer Software 44.3 (2018): 181-189.

[39] Muruganantham, R., and P. Ganeshkumar. "Quality of Service Enhancement in Wireless Sensor Network Using Flower Pollination Algorithm." Programming and Computer Software44.6 (2018): 398-406.

[40] Raja, R., and P. Ganeshkumar. "QoSTRP: A Trusted Clustering Based Routing Protocol for Mobile Ad-Hoc Networks." Programming and Computer Software 44.6 (2018): 407-416.

[41] Pashchenko, N. F., K. S. Zipa, and A. V. Ignatenko. "An algorithm for the visualization of stereo images simultaneously captured with different exposures." Programming and Computer Software 43.4 (2017): 250-257.

[42] Varnovskiy, N. P., et al. "Secure cloud computing based on threshold homomorphic encryption." Programming and Computer Software 41.4 (2015): 215-218.

[43] Zelenova, Sophia A., and Sergey V. Zelenov. "Schedulability Analysis for Strictly Periodic Tasks in RTOS." Programming and Computer Software 44.3 (2018): 159-169.

[44] Zipa, Kristina S., and Alexey V. Ignatenko. "Algorithms for the analysis and visualization of high dynamic range images based on human perception." Programming and Computer Software 42.6 (2016): 367-374.

Информация об авторах / Information about authors

Зульфикар АХМАД – преподаватель фпкультета информационных технологий Университета Хазара, Мансехра, Пакистан.

Zulfiqar AHMAD is a lecturer in the Department of Information Technology, Hazara Universtiy, Mansehra, Pakistan.

Али Имран ДЖЕХАНГИРИ – преподаватель факультета информационных технологий Университета Хазара, Мансехра, Пакистан. Он получил степень PhD в 2015 году в Университете им. Георга Августа Геттингена, Германия. Занимается исследовательской деятельностью, связанной с параллельными вычислениями, грид-вычислениями, облачными вычислениями и большими данными.

Ali Imran JEHANGIRI is a lecturer in the Department of Information Technology, Hazara Universtiy, Mansehra, Pakistan. He received Ph.D. degree in Computer Science from the Georg-August-Univesity Goettingen, Germany in 2015. He is involved in research activities dealing with parallel, Grid computing, Cloud computing and Big data.

Мехрин ИФТИХАР, преподаватель факультета информационных технологий, Университет Хазара, Мансехра, Пакистан

Mehreen IFTIKHAR is a lecturer in the Department of Information Technology, Hazara Universtiy, Mansehra, Pakistan.

Ариф Икбал УМАР получил докторскую степень в области компьютерных наук в Университете Бэйхан (БУАА), Пекин, Китай. В настоящее время он работает доцентом (компьютерные науки) на факультете информационных технологий Университета Хазара, Мансехра, Пакистан. Его исследовательские интересы включают интеллектуальный анализ данных, машинное обучение, поиск информации, цифровую обработку изображений, безопасность компьютерных сетей и сенсорных сетей.

Arif Iqbal UMAR obtained his PhD in computer science from BeiHang University (BUAA), Beijing, China. Currently, he is working as an assistant professor (computer science) at the Information Technology Department of Hazara University, Mansehra, Pakistan. His research interests include data mining, machine learning, information retrieval, digital image processing, computer networks security, and sensor networks.

Ибрар АФЗАЛ – преподаватель факультета информационных технологий Университета Хазара, Мансехра, Пакистан.

Mr.Ibrar AFZAL is a lecturer in the Department of Information Technology, Hazara Universtiy, Mansehra, Pakistan.