ПРОГРАММИРОВАНИЕ, 2019, № 5, с. 36–42

ПАРАЛЛЕЛЬНОЕ И РАСПРЕДЕЛЕННОЕ ПРОГРАММИРОВАНИЕ

УДК 681.3

ПОДХОДЫ К ПОВЫШЕНИЮ ЭФФЕКТИВНОСТИ ЭКСПЛУАТАЦИИ ЦОД

© 2019 г. В. А. Костенко[a],*, А. А. Чупахин[a],**

[a] *Московский государственный университет имени М.В. Ломоносова*
119991 Москва, Ленинские горы, д. 1, Россия

** E-mail: kost@cs.msu.su*

*** E-mail: andrewchup@lvk.cs.msu.ru*

Поступила в редакцию 18.11.2018 г.
После доработки 26.11.2018 г.
Принята к публикации 26.11.2018 г.

В статье предложены методы повышения эффективности использования ресурсов центров обработки данных. Они основаны на расширении функциональных возможностей и повышении точности алгоритма отображения запросов на физические ресурсы используемого в планировщике облачной платформы.

DOI: 10.1134/S0132347419050042

ВВЕДЕНИЕ

В работе рассматриваются центры обработки данных (ЦОД) работающие в режиме инфраструктура как услуга (IaaS – Infrastructure-as-a-Service). При работе ЦОД в режиме IaaS необходима возможность задания критериев качества сервиса (SLA) для всех типов ресурсов: систем хранения данных, вычислительных и сетевых ресурсов. Вычислительные ресурсы, системы хранения данных и сетевые ресурсы должны рассматриваться как планируемые типы ресурсов. Для размещенных в ЦОД виртуальных ресурсов необходимо гарантированное выполнение запрошенных SLA.

Эффективность эксплуатации ЦОД будем оценивать по следующим критериям:

1. Загрузка физических ресурсов (загрузка вычислительных ресурсов и систем хранения данных максимизируется, загрузка сетевых ресурсов минимизируется).

2. Процент размещенных запросов на создание виртуальных сетей из исходно поступивших.

3. Производительность виртуальных машин.

Предлагаемые в работе подходы к повышению эффективности эксплуатации ЦОД основаны на расширении функциональных возможностей и повышении точности алгоритма отображения запросов на физические ресурсы, используемого в планировщике облачной платформы.

1. ЗАДАЧА ОТОБРАЖЕНИЯ ЗАПРОСОВ НА ФИЗИЧЕСКИЕ РЕСУРСЫ ЦОД

Модель физических ресурсов ЦОД задается размеченным графом $H = (P \cup M \cup K, L)$, где P – множество вычислительных узлов (серверов), M – множество хранилищ данных, K – множество коммутационных элементов сети обмена ЦОД, L – множество физических каналов передачи данных. На множествах P, M, K и L определены векторные функции скалярного аргумента, задающие соответственно характеристики вычислительных узлов, хранилищ данных, коммутационных элементов и каналов передачи данных.

Запрос на создание виртуальной сети задается размещенным графом $G = (W \cup S, E)$, где W – множество виртуальных машин, S – множество виртуальных систем хранения данных (storage-элементов), E – множество виртуальных каналов передачи данных. На множествах W, S и E определены векторные функции скалярного аргумента, задающие характеристики запрашиваемого виртуального элемента (требуемое качество сервиса (SLA)). Для виртуальных машин дополнительно к характеристикам могут быть заданы следующие политики размещения:

1. Каждой виртуальной машине можно поставить в соответствие набор физических серверов, на одном из которых она должна быть размещена – VM-to-PM affinity.

2. Каждой виртуальной машине можно поставить в соответствие набор физических серверов,

на которых она не может быть размещена – VM-to-PM anti-affinity.

3. Набор виртуальных машин, который должен быть размещен на одном физическом сервере – VM-to-VM affinity.

4. Набор виртуальных машин, который должен быть размещен на разных физических серверах – VM-to-VM anti-affinity.

Размещение нескольких виртуальных машин на одном сервере снижает задержку при их взаимодействии. Размещение виртуальных машин на разных серверах повышает надежность, например, таким образом можно размещать два экземпляра одного и того же веб-сервера. Политики VM-to-PM имеют административную функцию: перемещение виртуальных машин на определенный сервер или удаление виртуальных машин с некоторого сервера, который будет вскоре отключен.

Назначением запроса будем называть отображение:

$$A:\ G \to H = \{W \to P, S \to M, E \to \{K, L\}\}.$$

Между характеристиками запросов и соответствующих характеристик физических ресурсов возможно три типа отношений. Обозначим через x характеристику элемента запроса и через y соответствующую ей характеристику физического ресурса. Тогда эти отношения можно записать следующим образом:

1. Недопустимость перегрузки “емкости” физического ресурса:

$$\sum_{i \in R_j} x_i \le y_j,$$

R_j – множество запросов, назначенных на выполнение на физическом ресурсе j.

2. Соответствие типа физического и виртуального ресурса:

$$x_i = y_j.$$

3. Наличие требуемых характеристик у физического ресурса:

$$x_i \le y_j.$$

Отображение A называется корректным, если для всех физических ресурсов и всех их характеристик выполняются отношения 1–3 и заданные политики для виртуальных машин.

Остаточным графом доступных ресурсов H_{res} называется граф H, в котором переопределены значения функций по характеристикам, которые должны удовлетворять отношению 1. Значения каждой характеристики физического ресурса уменьшаются на сумму значений соответствующей характеристики виртуальных ресурсов, назначенных на этот физический ресурс.

Входными данными задачи построения отображения виртуальных ресурсов на физические ресурсы являются:

1. Остаточный граф доступных ресурсов $H_{res} = (P \cup M \cup K, L)$.

2. Множество назначенных запросов $Ten = \{(T_i, flag_i)\}$, где $flag_i = (0|1)$ – возможность миграции запроса T_i.

3. Отображение назначенных запросов: $Assigned = \{A: T_i \to H_{res}\}$.

4. Множество запросов, поступивших планировщику $Z = \{G_i\}$;

5. Множество дополнительно запрашиваемых ресурсов (элементов запроса) к ранее назначенным виртуальным сетям $V = \{(W, i_w), (S, i_s), (E, i_e)\}$, где i_w, i_s, i_e – индексы запросов, к которым относятся данные виртуальные ресурсы.

Введем множество $R = Z \cup V$ – все поступившие запросы для назначения на физические ресурсы ЦОД. Требуется из множества R разместить на выполнение в ЦОД максимальное количество запросов и если есть виртуальные ресурсы, отображение которых изменилось, то построить для этих ресурсов план миграции. План миграции виртуальных ресурсов должен строиться с учетом требований гарантированного выполнения соглашений о качестве сервиса размещенных в ЦОД виртуальных ресурсов при выполнении плана и ограничений на допустимое время выполнения плана.

2. ТРЕБОВАНИЯ К АЛГОРИТМУ

Эффективность работы облачных платформ и использования ресурсов ЦОД во многом зависит от алгоритма отображения запросов на физические ресурсы ЦОД используемого в планировщике облачной платформы. Применяются жадные алгоритмы и алгоритмы, сочетающие жадные стратегии и ограниченный перебор [1, 2], эвристические алгоритмы [3, 4]. В частности, применяются модифицированные алгоритмы упаковки в контейнеры [5]. Также применяются муравьиные алгоритмы [6], алгоритмы имитации отжига [7], генетические алгоритмы [8], иммунные алгоритмы [9].

Для повышения эффективности использования физических ресурсов ЦОД алгоритм должен обладать следующими свойствами:

1. Максимизируется количество размещенных запросов.

2. Вычислительные ресурсы, хранилища данных являются планируемыми ресурсами.

3. Сетевые ресурсы являются планируемыми ресурсами.

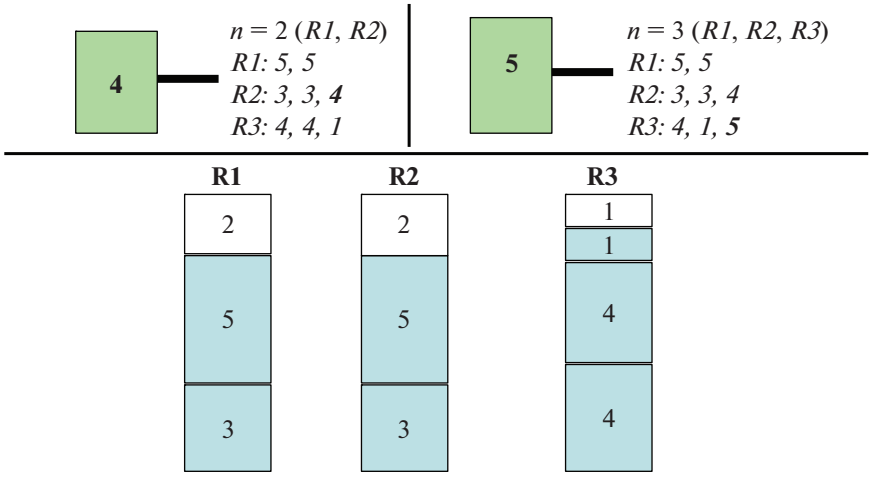

Рис. 1. Пример выполнения процедуры ограниченного перебора.

4. Учет политик размещения виртуальных ресурсов.

5. Учет NUMA архитектуры серверов.

6. Возможность построения плана миграции ранее размещенных виртуальных ресурсов. Данная возможность необходима для устранения фрагментации физических ресурсов возникающей в ходе эксплуатации ЦОД.

Ни один из известных авторам алгоритмов не обладает в совокупности всеми перечисленными свойствами.

3. АЛГОРИТМ ОТОБРАЖЕНИЯ ЗАПРОСОВ И ПОДХОДЫ К ПОВЫШЕНИЮ ЭФФЕКТИВНОСТИ ЭКСПЛУАТАЦИИ ЦОД

Алгоритм основан на сочетании жадных стратегий и ограниченного перебора [10] и состоит из двух шагов:

1. Назначение виртуальных машин на вычислительные узлы и виртуальных систем хранения данных на сервера хранения данных.

2. Для полученных отображений виртуальных машин и виртуальных систем хранения данных на физические ресурсы производится построение маршрутов виртуальных каналов в сети обмена ЦОД.

Алгоритм выполнения шага 1:

1. Из множества ресурсных запросов R выбрать очередной запрос G_i в соответствии с жадным критерием K_G.

2. Выбрать очередной элемент e (виртуальную машину $e \in W$, $W \in G_i$ или storage-элемент $e \in S$, $S \in G_i$) в соответствии с жадным критерием K_e. Если G_i пусто, то перейти к шагу 1.

3. Сформировать множество физических ресурсов Ph ($Ph \subseteq P$ или $Ph \subseteq M$ соответственно), на которые может быть назначен выбранный элемент e, то есть для которого выполнены отношения корректности отображения в случае назначения запроса e на физический ресурс.

3.1. Если множество Ph пусто, то вызвать процедуру ограниченного перебора. Если:

1. Процедура возвращает неуспех, то запрос G_i не может быть назначен: удалить ранее назначенные элементы запроса G_i и переопределить значения характеристик физических ресурсов, которые должны удовлетворять отношению 1.

2. Процедура возвращает новое отображение: вызвать процедуру построения плана миграции и в случае его успешного построения провести назначение элементов в соответствии с новым отображением и переопределить значения характеристик физических ресурсов, которые должны удовлетворять отношению 1.

3. Если множество R не пусто, перейти на шаг 1; в противном случае завершить работу алгоритма.

3.2. Если множество Ph не пусто, то выбрать физический элемент $ph \subseteq Ph$ для назначения в соответствии с жадным критерием K_{ph} и назначить элемент запроса e на физический элемент ph и переопределить значения его характеристик. Перейти на шаг 2, если есть нерассмотренные элементы запроса. Перейти на шаг 1, если множество R не пусто; в противном случае завершить работу алгоритма.

Принцип работы процедуры ограниченного перебора проиллюстрирован на рис. 1.

Глубина перебора ограничивается заданным числом n, которое указывает максимальное количество физических элементов, среди которых можно производить переназначение (то есть при $n = 2$ назначенные элементы могут сниматься и переназначаться одновременно не более чем с

Таблица 1. Эффективность использования ресурсов ЦОД

	Нет ограничений на время миграции			Миграция запрещена		
	Загрузка CPU	Загрузка RAM	Процент размещенных запросов	Загрузка CPU	Загрузка RAM	Процент размещенных запросов
Тест 1	0.97	0.33	100%	0.83	0.29	75%
Тест 2	0.97	0.33	100%	0.83	0.29	75%
Тест 3	0.89	0.49	85%	0.83	0.47	75%
Тест 4	0.86	0.48	79%	0.83	0.47	75%
Тест 5	0.85	0.47	77.5%	0.83	0.47	75%

двух физических серверов). На рис. 1 слева изображена ситуация, в которой виртуальная машина, требующая 4 ядра, не может быть назначена для выполнения ни на один из физических серверов, несмотря на то, что суммарного количества ядер достаточно для ее назначения. Цветом закрашены назначенные на сервера R1, R2, R3 виртуальные машины и для каждой машины указано количество занимаемых ядер, белый прямоугольник – количество свободных ядер на сервере. При переназначении виртуальных машин, назначенных на сервера R1, R2, фрагментация устраняется, и виртуальная машина может быть назначена. То есть в этом случае достаточно глубины перебора $n = 2$. На рис. 1 справа изображена ситуация, в которой виртуальная машина требует 5 ядер. В этом случае если глубина перебора равна 2, машина не может быть назначена. Но если задана глубина перебора $n = 3$, то виртуальная машина будет назначена. То есть, увеличивая допустимую глубину перебора, можно повышать точность алгоритма, но при этом увеличивается его вычислительная сложность.

Описание жадных критериев, используемых в алгоритме, и описание алгоритма построения маршрутов виртуальных каналов в сети обмена ЦОД приведено в работах [1, 2].

4. УВЕЛИЧЕНИЕ ЗАГРУЗКИ ФИЗИЧЕСКИХ РЕСУРСОВ И ПРОЦЕНТА РАЗМЕЩЕННЫХ ЗАПРОСОВ

В ходе эксплуатации ЦОД происходит снятие виртуальных ресурсов с выполнения и запуск новых. При попытке разместить новый виртуальный ресурс в ЦОД, может возникнуть следующая ситуация. Суммарного количества свободных физических ресурсов по всем запрошенным критериям качества сервиса хватает для размещения виртуального ресурса, но не существует физического ресурса, на который можно было бы разместить этот виртуальный ресурс с выполнением всех запрошенных для него критериев SLA и ограничений на корректность отображения. То есть в ходе эксплуатации ЦОД возникает фрагментация физических ресурсов, которая приводит к снижению эффективности их использования. В предложенном алгоритме фрагментация ресурсов может устраняться за счет миграции работающих виртуальных ресурсов, что позволило увеличить загрузку физических ресурсов и процент размещенных запросов. Было исследовано влияние миграции на эффективность работы ЦОД по критериям загрузка физических ресурсов и количество размещенных запросов.

В таблице 1 приведены результаты работы алгоритма отображения запросов на физические ресурсы ЦОД с построением плана миграции работающих виртуальных ресурсов для двух случаев:

1. Миграция работающих виртуальных ресурсов запрещена.

2. Нет ограничений на время миграции виртуальных ресурсов, поменявших свое расположение.

В таблице приведены процент размещенных запросов, учитывая и ранее размещенные, и новые запросы, также приведена загрузка вычислительных ресурсов.

При наличии возможности проведения миграции загрузка ресурсов и процент размещенных запросов для всех тестов выше, чем при отсутствии такой возможности. Выигрыш будет тем больше, чем чаще вызывается процедура ограниченного перебора и чем чаще она успешно завершается.

Но если есть ограничение на время миграции, то результаты работы алгоритма будут тем хуже, чем меньше время, отведенное для миграции. Данная зависимость вытекает из результатов работы алгоритма, представленных в таблице 2. В качестве условных единиц для измерения времени рассматриваются секунды.

Для того чтобы показать влияние миграции на качество получаемого алгоритмом планирования решения, были выбраны такие данные, на которых бы часто вызывалась процедура ограниченного перебора. Физические ресурсы представляют связанные между собой кластеры из 3 серве-

Таблица 2. Эффективность использования ресурсов ЦОД при наличии ограничения на время миграции

	Тест 1	Тест 2	Тест 3	Тест 4	Тест 5
Время миграции = 500 усл. ед.					
Загрузка CPU	0.97	0.97	0.89	0.86	0.85
Загрузка RAM	0.33	0.33	0.49	0.48	0.47
Процент размещенных запросов	100%	100%	85%	79%	77.5%
Время миграции = 400 усл. ед.					
Загрузка CPU	0.94	0.95	0.89	0.86	0.85
Загрузка RAM	0.325	0.328	0.49	0.48	0.47
Процент размещенных запросов	95%	97%	85%	79%	77.5%
Время миграции = 300 усл. ед.					
Загрузка CPU	0.9	0.91	0.89	0.86	0.85
Загрузка RAM	0.31	0.32	0.49	0.48	0.47
Процент размещенных запросов	87.5%	90%	85%	79%	77.5%
Время миграции = 200 усл. ед.					
Загрузка CPU	0.87	0.87	0.89	0.86	0.85
Загрузка RAM	0.3	0.3	0.49	0.48	0.47
Процент размещенных запросов	82.5%	82%	85%	79%	77.5%
Время миграции = 100 усл. ед.					
Загрузка CPU	0.85	0.85	0.86	0.86	0.85
Загрузка RAM	0.3	0.29	0.48	0.48	0.47
Процент размещенных запросов	77.5%	78%	80%	79%	77.5%
Время миграции = 50 усл. ед.					
Загрузка CPU	0.85	0.84	0.84	0.85	0.85
Загрузка RAM	0.3	0.29	0.47	0.47	0.47
Процент размещенных запросов	77.5%	76%	76.2%	77.5%	77.5%

Таблица 3. Описание параметров для генерации исходных данных

	Физические сервера				Виртуальные машины		Доля размещ. запросов
Тест, №	Кол-во	CPU	RAM	Каналы	CPU	RAM	0.75
1	30	10	8192	50	1-5	1024	0.75
2	45	10	8192	80	1-5	1024	0.75
3	60	10	5120	150	1-5	1024	0.75
4	90	10	5120	200	1-5	1024	0.75
5	120	10	5120	300	1-5	1024	0.75

ров и коммутатора. На эти сервера назначены виртуальные машины таким образом, что без вызова процедуры ограниченного перебора назначение новой виртуальной машины невозможно. Описанные кластеры связаны в кольцо каналами, которые проходят между коммутаторами.

В таблице 3 представлено описание параметров для генерации исходных данных: количество серверов в топологии; интервалы, в которых могут меняться количество CPU и объем RAM у виртуальных машин и у физических серверов; пропускная способность физических каналов связи (измеряется в Мб/с). Процент размещенных запросов на момент запуска алгоритма означает следующее. Сначала генерируются N запросов. Затем из этих N запросов выбирается и размещается M запросов. Доля размещенных запросов равняется M/N. На вход алгоритму подается $N-M$ не размещенных запросов.

5. ПОВЫШЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ВИРТУАЛЬНЫХ МАШИН

Современные облачные платформы не гарантируют стабильную производительность виртуальных машин. Авторы статьи [11] исследовали в течение месяца производительность виртуальных машин в облачной платформе Amazon. Результаты исследования (коэффициент вариации) основных характеристик, влияющих на производительность виртуальных машин, представлены ниже:

3. Производительность CPU – 24%.

4. Скорость оперативной памяти – 10%.

5. Последовательное чтение с диска – 17%, последовательная запись на диск – 19%.

6. Случайные чтения/записи на диск – от 9% до 20%.

Производительность CPU и производительность оперативной памяти авторы статьи оценивали при помощи утилиты Ubench [12]. Скорость последовательного чтения и последовательной записи измерялась в Кбайт/с. Для операций случайного чтения/записи измерялось время операций в секундах.

Причины таких больших значений коэффициентов вариации:

1. Гетерогенная инфраструктура физических ресурсов ЦОД [13]. Виртуальные машины могут размещаться на серверах с разным типом процессора, так как при заказе виртуальной машины нельзя указать тип процессора.

2. Переиспользование ресурсов. Суммарное количество ресурсов, необходимых для работы виртуальных машин, может быть больше, чем доступно на сервере. Такая ситуация называется overcommit [14]. Ресурсы сервера в этом случае становятся разделяемыми между виртуальными машинами. Они не могут постоянно использовать некоторый набор ресурсов, за их использованием следит гипервизор [15].

3. NUMA архитектура. Виртуальные машины могут быть размещены на серверах с NUMA архитектурой.

Переиспользование ресурсов в предложенном алгоритме устраняется введением ограничения на корректность отображения “Недопустимость перегрузки “емкости” физического ресурса”.

Производительность виртуальных машин на серверах с NUMA архитектурой можно повысить за счет учета архитектуры серверов. Наибольшую производительность виртуальная машина будет иметь, если все ее ядра имеют прямой доступ к одному и тому же блоку памяти. В предложенном алгоритме это можно учесть введение матрицы расстояния между ядрами. Элементы матрицы отражают затраты времени на обмен данными между ядрами.

Экспериментальные исследования показали, что учет NUMA архитектуры снижает количество размещенных запросов, но при этом увеличивается производительность виртуальных машин. Производительность может упасть у 20% виртуальных машин, если не учитывать NUMA архитектуру.

Гетерогенность инфраструктуры физических ресурсов ЦОД может быть учтена введением в набор критериев SLA запрашиваемого типа процессора. В предложенном алгоритме это будет учитываться при проверке ограничения на корректность отображения “Соответствие типа физического и виртуального ресурса”.

ЗАКЛЮЧЕНИЕ

Предложенный в работе алгоритм отображения запросов на создание виртуальных сетей на ресурсы ЦОД позволяет увеличить загрузку физических ресурсов и процент размещенных запросов из исходно поступивших за счет устранения возникающей в ходе эксплуатации фрагментации ресурсов. Производительность виртуальных машин увеличивается за счет учета архитектуры серверов при построении отображения.

Введенная типизация отношений между характеристиками ресурсов и критериями SLA, а также возможность задавать для виртуальных машин политики размещения позволяют осуществлять настройку алгоритма на особенности облачной платформы. Например, задавать набор критериев SLA, значения которых должны быть указаны в запросе.

Используемый метод построения алгоритма сочетающий жадные стратегии и ограниченный перебор позволяет выбирать требуемый баланс между точностью и вычислительной сложностью алгоритма.

БЛАГОДАРНОСТИ

Работа выполнена при финансовой поддержке РФФИ (грант № 19-07-00614).

СПИСОК ЛИТЕРАТУРЫ

1. *Zotov I.A., Kostenko V.A.* Resource Allocation Algorithm in Data Centers with a Unified Scheduler for Different Types of Resources // Journal of Computer and Systems Sciences International. 2015. V. 54. № 1. P. 59–68.
2. *Vdovin P.M., Kostenko V.A.* Algorithm for Resource Allocation in Data Centers with Independent Schedulers for Different Types of Resources // Journal of Computer and Systems Sciences Intern. 2014. V. 53. № 6. P. 854–866.
3. Md Rabbani. Resource Management in Virtualized Data Center. Waterloo, Ontario, Canada, 2014. 72c. https://uwspace.uwaterloo.ca/ bitstream/handle/10012/8280/Rabbani_Md.pdf?sequence=3.

4. *Benson T., Akella A., Shaikh A., Sahu S.* CloudNaaS: A Cloud Networking Platform for Enterprise Applications. Madison, WI, USA. 2011, 13c. http://pages.cs.wisc.edu/~tbenson/papers/CloudNaaS.pdf
5. *Ngenzi A., Selvarani R., Nair S.* Dynamic Resource Management In Cloud Data Centers For Server Consolidation, Bangalore-India, 2015, 8 c. https://arxiv.org/ftp/arxiv/papers/1505/1505.00577.pdf
6. *Ashraf A., Porres I.* Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system // International Journal of Parallel, Emergent and Distributed Systems. 2018. V. 33. № 1.
7. *Ricci R., Alfeld C., Lepreau J.* A Solver for the Network Testbed Mapping problem // ACM Special Interest Group on Data Communication, Computer Communication Review. 2003. V. 33. P. 65–81.
8. *Mi X., Chang X., Liu J., Sun L., Xing B.* Embedding Virtual Infrastructure Based on Genetic Algorithm, Proceedings of the 13th International Conference on Parallel and Distributed Computing, Applications and Technologies, December 14–16. 2012. P. 239–244.
9. *Zhang Z., Su S., Lin Y., Cheng X., Shuang K., Xu P.* Adaptive multi-objective artificial immune system based virtual network embedding // Journal of Network and Computer Applications, July 2015. V. 53. № C. P. 140–155.
10. *Kostenko V. A.* Combinatorial Optimization Algorithms Combining Greedy Strategies with A Limited Search Procedure // Journal of Computer and Systems Sciences International. 2017. V. 56. № 2. P. 218–226.
11. *Schad J., Dittrich J., Jorge-Arnulfo Quiané-Ruiz J.-A.* Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance // Journal Proceedings of the VLDB Endowment, September 2010. V. 3. № 1–2. P. 460–471.
12. Unix benchmark Utility. http://phystech.com/download/ubench.html.
13. *Ou Z. et al.* Is the Same Instance Type Created Equal? Exploiting Heterogeneity of Public Clouds. IEEE Transactions on Cloud Computing, July–December 2013. V. 2. № 2.
14. *Simao J., Veiga L.* Partial Utility-Driven Scheduling for Flexible SLA and Pricing Arbitration in Clouds. IEEE Transactions on Cloud Computing. 2016. V. 4. № 4. P. 467–480.
15. *Cherkasova L., Gupta D., Vahdat A.* Comparison of the three CPU schedulers in Xen. Newsletter ACM SIGMETRICS Performance Evaluation Review, September 2007. V. 35. № 2. P. 42–51.