
**СИСТЕМНЫЙ АНАЛИЗ
И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ**

УДК 681.3

**ВЛИЯНИЕ ПОЛИТИК РАЗМЕЩЕНИЯ ВИРТУАЛЬНЫХ МАШИН
НА ЭФФЕКТИВНОСТЬ ИСПОЛЬЗОВАНИЯ РЕСУРСОВ
ЦЕНТРОВ ОБРАБОТКИ ДАННЫХ¹**

© 2020 г. Ю. О. Васильева^{a,*}, В. А. Костенко^{a,**}, А. А. Чупахин^{a,***}

^aМГУ им. М.В. Ломоносова, Москва, Россия

*e-mail: vasilyeva.jo@gmail.com

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

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

Поступила в редакцию 12.11.2019 г.

После доработки 26.11.2019 г.

Принята к публикации 27.01.2020 г.

Использование при создании виртуальных машин политик размещения позволяет повысить надежность приложений, уменьшить задержку при взаимодействии виртуальных машин, а также упростить администрирование центров обработки данных. Задание политик размещения виртуальных машин вносит дополнительные требования к их размещению на физических ресурсах. Это может приводить к тому, что снижается загрузка физических ресурсов и уменьшается процент размещенных запросов из исходно поступивших. В работе предлагается алгоритм отображения запросов (на создание виртуальных ресурсов) на физические ресурсы центра обработки данных с опциональным учетом политик размещения виртуальных машин, и приводятся результаты экспериментального исследования влияния учета политик размещения на загрузку физических ресурсов и процент размещенных запросов.

DOI: 10.31857/S0002338820030166

Введение. В статье рассматриваются центры обработки данных (ЦОД), работающие в режиме инфраструктура как услуга (IaaS — infrastructure-as-a-service). При работе ЦОД в режиме IaaS в ЦОД поступают запросы на создание виртуальных ресурсов: виртуальных машин (ВМ), виртуальных систем хранения данных (ВСХД) и виртуальных каналов передачи данных (ВК). Для каждого элемента запроса в соглашении о качестве обслуживания (SLA — service level agreement) задаются требования к виртуальным ресурсам. Для размещенных в ЦОД виртуальных ресурсов необходимо гарантированное выполнение запрошенных в SLA требований, в качестве которых могут задаваться политики размещения виртуальных машин. Задание политик позволяет повысить надежность приложений, которые выполняются на ВМ, уменьшить задержку при взаимодействии ВМ друг с другом и упростить администрирование ЦОД. В запросе также указывается, является ли политика обязательной или опциональной. Запрос с обязательной политикой размещается только в случае, когда политика может быть выполнена. Запрос с опциональной политикой размещается, даже если она не может быть выполнена.

Задание политик размещения виртуальных машин вносит дополнительные требования к размещению ВМ на физических ресурсах. Это может приводить к снижению эффективности работы ЦОД: уменьшению загрузки физических ресурсов и снижению процента размещенных запросов из исходно поступивших.

Эффективность работы облачных платформ и использования физических ресурсов ЦОД во многом зависит от алгоритма отображения запросов на физические ресурсы ЦОД. Для планирования вычислений в ЦОД известны жадные алгоритмы и алгоритмы, сочетающие жадные стратегии и ограниченный перебор [1, 2], эвристические алгоритмы [3, 4], модифицированные алгоритмы упаковки в контейнеры [5]. Также применяются муравьиные алгоритмы [6], алгоритмы имитации отжига [7], генетические алгоритмы [8], иммунные алгоритмы [9]. На практике в основном используются жадные алгоритмы. Это обусловлено тем, что они имеют низкую вычис-

¹ Работа выполнена при финансовой поддержке РФФИ (грант № 19-07-00614).

лительную сложность и допускают простую настройку на особенности размещения запросов на создание виртуальных ресурсов в конкретных ЦОД. Недостатком жадных алгоритмов является то, что на разных классах исходных данных их точность может сильно отличаться. Алгоритмы других классов не допускают их адаптацию к опциональному учету политик размещения ВМ, поэтому адаптация сводится к разработке нового алгоритма.

Предлагаемый алгоритм основан на сочетании жадных стратегий и ограниченного перебора [10]:

на каждом шаге работы алгоритма делается локально-оптимальный выбор в соответствии с используемой жадной стратегией,

на каждом шаге проверка того, что “жадный выбор не закрывает пути к оптимальному решению”,

вызов процедуры ограниченного перебора, если проверка условия “жадный выбор не закрывает пути к оптимальному решению” дала отрицательный результат.

В работе предлагается алгоритм отображения запросов на физические ресурсы ЦОД с опциональным учетом политик размещения виртуальных машин и приводятся данные экспериментального исследования, насколько при учете политик снижается загрузка физических ресурсов и уменьшается процент размещенных запросов.

1. Задача отображения запросов на физические ресурсы ЦОД с опциональным учетом политик.

Модель физических ресурсов ЦОД будем задавать графом $H = (P \cup M \cup K, L)$, где P – множество вычислительных узлов, 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 виртуальных ресурсов будем задавать графом $G = (W \cup S, E)$, где W – множество виртуальных машин, используемых приложениями, S – множество виртуальных хранилищ данных (storage-элементов), E – множество виртуальных каналов передачи данных между виртуальными машинами и storage-элементами запроса. На множествах W , S и E определены векторные функции скалярного аргумента, задающие характеристики запрашиваемого виртуального элемента SLA.

В запросе могут быть заданы политики размещения виртуальных машин. Политики могут быть четырех типов:

1) VM-to-PM affinity – виртуальной машине ставится в соответствие набор физических серверов, на одном из которых она должна быть размещена;

2) VM-to-PM anti-affinity – виртуальной машине ставится в соответствие набор физических серверов, на одном из которых она не может быть размещена;

3) VM-to-VM affinity – задается набор виртуальных машин, которые должны быть размещены на одном физическом сервере;

4) VM-to-VM anti-affinity – задается набор виртуальных машин, которые должны быть размещены на разных физических серверах.

Политики задаются для каждого запроса независимо. Для каждой политики должно указываться, является ли политика обязательной или опциональной.

Запрос с обязательной политикой либо размещается в соответствии с ней, либо не размещается вовсе. Опциональные политики учитываются по мере возможности.

Назначением запроса называется отображение A :

$$A : G \rightarrow H \cup \{\emptyset\} = \{W \rightarrow P \cup \{\emptyset\}, S \rightarrow M \cup \{\emptyset\}, E \rightarrow \{K, L\} \cup \{\emptyset\}\}. \quad (1.1)$$

Введем три типа отношений между характеристиками запросов и соответствующими характеристиками физических ресурсов. Будем обозначать характеристику запроса как x , а соответствующую ей характеристику физического ресурса – как y . Эти отношения запишем следующим образом:

1) недопустимость перегрузки емкости физического ресурса: $\sum_{i \in R_j} x_i \leq y_i$, здесь R_j – множество запросов, назначенных на выполнение на физическом ресурсе j ;

2) соответствие типов y запрашиваемого ресурса и физического ресурса: $x = y$;

3) наличие требуемых характеристик y физического ресурса: $x \leq y$.

Отображение (1.1) будем называть *корректным*, если для всех физических ресурсов и всех их характеристик выполняются отношения 1)–3) и заданные политики для виртуальных машин.

Остаточным графом доступных ресурсов называется граф H_{res} , который получается из графа физических ресурсов H , для которого переопределены значения функций по характеристикам, которые должны удовлетворять отношению корректности 1). Значения характеристик переопределяются по правилу: значение характеристики физического ресурса уменьшается на сумму значений характеристик виртуальных ресурсов, назначенных на данный физический ресурс.

В качестве *исходных данных* задачи назначения ресурсных запросов на физические ресурсы заданы:

1) множество запросов (назначенных, частично назначенных и ожидающих назначения) $Z = \{G_i\}$, поступивших планировщику;

2) остаточный граф доступных ресурсов H_{res} .

Требуется: из множества поступивших планировщику запросов Z разместить в центре обработки данных максимальное число запросов таким образом, чтобы новое отображение запросов на физические ресурсы было корректным.

2. Алгоритм отображения запросов на физические ресурсы ЦОД с опциональным учетом политик. Требования к алгоритму совпадают с требованиями, сформулированными в [11], и к ним еще добавляется требование опционального учета политик размещения виртуальных машин.

Алгоритм сочетает в себе жадные стратегии и ограниченный перебор и состоит из следующих последовательно выполняемых шагов.

Шаг 1. Если множество $\{G_i\}$ не пусто, выбрать очередной запрос G_i в соответствии с жадным критерием K_G , иначе завершение работы алгоритма.

Шаг 1.1. Сформировать множество виртуальных машин из запроса W , для которых заданы политики размещения VM-to-VM affinity или VM-to-VM anti-affinity. Сформировать множество R , элементами которого являются множества из виртуальных машин, которые связаны одной из политик размещения.

Шаг 1.2. Если множество R пусто, перейти на шаг 2. Иначе выбираем любой элемент r из R . Остальные ВМ из множества отсортировать по неубыванию в соответствии с жадным критерием K_r .

Шаг 1.3. Если множество r пусто, удалить его из R и перейти на шаг 1.2. Для первого элемента из отсортированного множества: сформировать множество физических узлов Ph , на которые может быть назначен данный элемент.

Шаг 1.4. Если множество Ph пусто: если хотя бы у одной виртуальной машины из r указано, что политика обязательная, удалить запрос из $\{G_i\}$, снять уже назначенные элементы запроса, перейти на шаг 1, иначе удалить все ВМ из r из множества W , удалить текущее множество из R и перейти на шаг 1.2.

Шаг 1.5. Выбрать физический ресурс ph из множества Ph в соответствии с жадным критерием K_p . Для VM-to-VM affinity: если на ph можно разместить все виртуальные машины из r , то назначить их на ph . Для VM-to-VM anti-affinity: если на ph не назначена ни одна виртуальная машина из r , то назначить выбранную виртуальную машину на ph . Если элемент не назначен, удалить ph из Ph , перейти на шаг 1.4. Перейти на шаг 3.

Шаг 1.6. Удалить текущий элемент из r . Перейти на шаг 1.3.

Шаг 2. Если множество G_i пусто, то удалить его из $\{G_i\}$ и перейти на шаг 1. Выбрать элемент e из G_i в соответствии с жадным критерием K_e .

Шаг 3. Сформировать множество физических узлов Ph , на которые может быть назначен данный элемент.

Шаг 3.1. Если множество Ph пусто, вызвать процедуру ограниченного перебора.

Шаг 3.2. Если процедура вернула неуспех, удалить запрос из $\{G_i\}$, снять уже назначенные элементы запроса, перейти на шаг 1, иначе провести назначение перейти на шаг 1.

Шаг 3.3. Если множество Ph не пусто, выбрать физический ресурс ph из множества Ph в соответствии с жадным критерием K_p и провести назначение. Перейти на шаг 2.

Процедура ограниченного перебора вызывается в случае невозможности назначить некоторый элемент N из запроса G_i ни на один физический ресурс ph . Процедура выполняет поиск среди множества уже назначенных элементов запросов такого подмножества A_d , что элемент N может

быть назначен после переназначения элементов из A_d . Перебор ограничивается числом d , задающим максимальное количество физических элементов, для которых возможно переназначение. Переназначение — это снятие всех виртуальных элементов и назначение их заново согласно алгоритму. При переборе рассматриваются только такие подмножества физических ресурсов, у которых суммарное количество остаточных ресурсов достаточно для назначения текущего виртуального элемента.

3. Исследование влияния политик размещения виртуальных машин на использование ресурсов ЦОД. Критерии эффективности использования ресурсов ЦОД:

- 1) загрузка физических ресурсов ЦОД,
- 2) процент размещенных запросов.

Целями исследования является выявить:

- 1) влияние учета политик размещения виртуальных ресурсов на количество размещенных запросов и загрузку центра обработки данных;
- 2) влияние обязательных политик на количество неразмещенных запросов;
- 3) влияние учета политик на время работы алгоритма.

Методика проведения эксперимента.

Исследования проводились на вычислительной системе со следующими характеристиками:

- 1) операционная система — Ubuntu 16.04 LTS,
- 2) размер оперативной памяти — 8 ГБ,
- 3) процессор — Intel(R) Core(TM) i7-6500U CPU 2.50GHz, 2 ядра.

В рамках исследования было проведено шесть экспериментов. Для всех экспериментов использовалась одна и та же физическая топология. Она была сгенерирована со следующими параметрами.

1. Топология: дерево.
2. Количество физических серверов: 256.
3. Параметры физических узлов:
 - а) количество ядер: 16–32,
 - б) количество RAM: 32–64 ГБ.
4. Пропускные способности физических каналов сети: 200–300 Мб/с.

Для каждого эксперимента генерировались разные множества запросов. Запросы были сгенерированы со следующими параметрами.

1. Топология: дерево, звезда, кольцо. Выбор делается случайным образом.
2. Количество запросов: 190–200.
3. Количество виртуальных машин в запросе: 6–8.
4. Параметры виртуальных машин:
 - а) количество ядер: 1–8,
 - б) количество RAM: 2–16 ГБ.
5. Пропускные способности виртуальных каналов: 2–8 Мб/с.
6. Процент ВМ, для которых заданы политики:
 - а) эксперименты 1–2: 20%,
 - б) эксперименты 3–4: 50%,
 - в) эксперименты 5–6: 80%.
7. Процент обязательных политик:
 - а) эксперименты 1, 3, 5: 20%,
 - б) эксперименты 2, 4, 6: 80%.

На рис. 1 показано, каким образом учет политик размещения запросов влияет на процент размещенных запросов и загрузку ЦОД. Для каждого из шести экспериментов приведены: процент размещенных запросов от поступивших (горизонтальная штриховка), процент используемых ядер от запрашиваемых (вертикальная штриховка), процент используемой оперативной памяти от запрашиваемой (диагональная штриховка). Из приведенной гистограммы видно, что учет политик негативно сказывается на количестве размещенных запросов. Причем чем выше процент запросов, содержащих политики, и чем выше процент обязательных политик, тем ниже процент размещенных запросов и загрузка ЦОД.

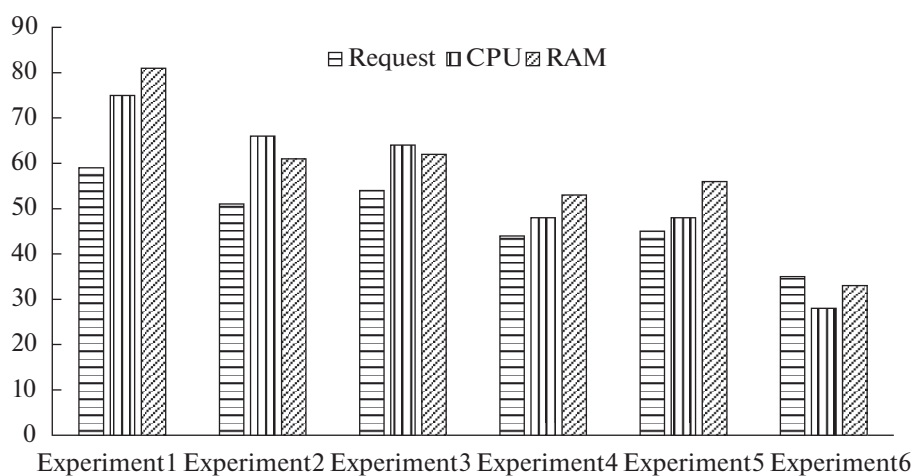


Рис. 1. Влияние учета политик на процент размещенных запросов и загрузку ЦОД

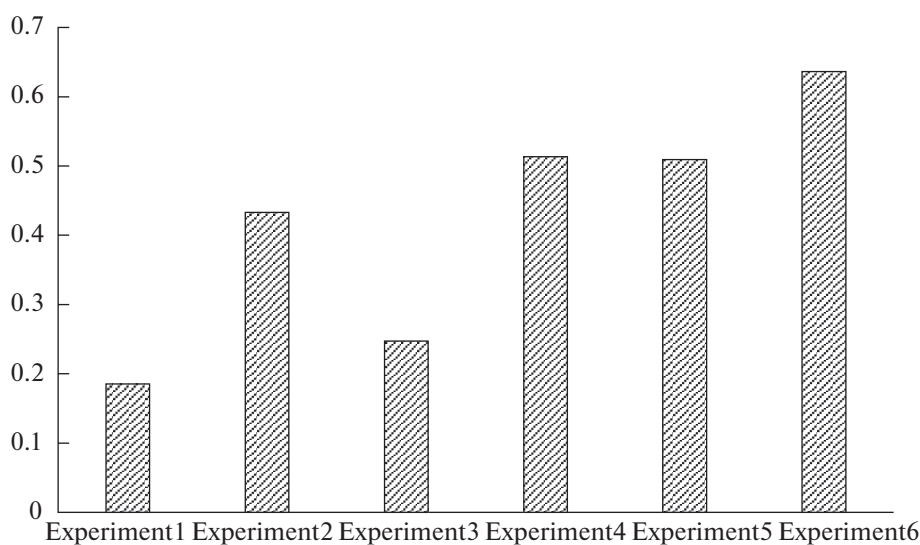


Рис. 2. Влияние обязательных политик на количество неразмещенных запросов



Рис. 3. Влияние учета политик на время работы алгоритма

На рис. 2 представлено, каким образом количество обязательных политик влияет на количество неразмещенных запросов. Для каждого из шести экспериментов приведена доля запросов с обязательными политиками от количества неразмещенных запросов. Из рассмотренной гистограммы можно сделать вывод, что при учете политик достаточно негативное влияние оказывают именно обязательные политики.

На рис. 3 показано, каким образом количество обязательных политик влияет на время работы алгоритма. Для каждого из шести сгенерированных тестов приведены: время работы алгоритма в секундах (вертикальная штриховка) и доля запросов, содержащих обязательную политику, от запросов, содержащих какую-либо политику (диагональная штриховка). Из данной гистограммы можно сделать вывод, что при высоком проценте опциональных политик размещения запросов время работы алгоритма возрастает.

Заключение. Предложенный алгоритм позволяет при построении отображения запросов на физические ресурсы ЦОД опционально учитывать политики размещения виртуальных машин в зависимости от указания, является политика обязательной или опциональной.

В ходе экспериментального исследования разработанного алгоритма было выявлено, что учет политик размещения запросов плохо сказывается на загрузке центра обработки данных, количестве размещенных запросов и времени работы алгоритма. На загрузку ЦОД и процент размещенных запросов негативно влияет высокий процент обязательных политик, так как у ВМ с политикой значительно сокращается число вариантов размещения. На время работы алгоритма отрицательно влияет высокий процент опциональных политик, так как при невозможности размещения с учетом политики алгоритм производит попытку размещения ВМ дважды.

Таким образом, необходимо учитывать, что запрос с обязательной политикой может быть не размещен и может дольше находиться в очереди запросов, ожидающих размещения на ресурсах ЦОД, с большей вероятностью, чем без указания политик или с указанием опциональных политик размещения.

СПИСОК ЛИТЕРАТУРЫ

1. Zotov I.A., Kostenko V.A. Resource Allocation Algorithm in Data Centers with a Unified Scheduler for Different Types of Resources // J. 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 // J. Computer and Systems Sciences International. 2014. V. 53. № 6. P. 854–866.
3. Rabbani M. Resource Management in Virtualized Data Center. Waterloo, Ontario, Canada, 2014. 72 p. URL: 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. 13 p. URL: <http://pages.cs.wisc.edu/~tbenson/papers/CloudNaaS.pdf> (дата обращения: 11.04.2016).
5. Ngenzi A., Selvarani R., Nair S. Dynamic Resource Management In Cloud Data Centers for Server Consolidation. Bangalore-India, 2015. 8 p. URL: <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 // Intern. J. Parallel. Emergent and Distributed Systems. 2018. V. 33. Is. 1.
7. Ricci R., Alfeld C., Lepreau J. A Solver for the Network Testbed Mapping Problem // ACM SIGCOMM 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 // 13th Intern. Conf. on Parallel and Distributed Computing, Applications and Technologies. Beijing, China, 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 // J. Network and Computer Applications. 2015. V. 53. C. P. 140–155.
10. Kostenko V.A. Combinatorial Optimization Algorithms Combining Greedy Strategies with A Limited Search Procedure // J. Computer and Systems Sciences International. 2017. V. 56. № 2. P. 218–226.
11. Костенко В.А., Чупахин А.А. Подходы к повышению эффективности эксплуатации ЦОД // Программирование. 2019. № 5. С. 36–42.