Алгоритм мультиагентного диспетчирования ресурсов в гетерогенной облачной среде

И.А. Каляев[1,2], А.И. Каляев[2,*], Я.С. Коровин[2]

[1]Южный федеральный университет, Ростов-на-Дону, Россия
[2]НИИ многопроцессорных вычислительных систем ЮФУ, Таганрог, Россия

[*]Контактный e-mail: anatoly@kalyaev.net

Работа посвящена проблеме адаптивного диспетчирования (распределения) ресурсов в облачной вычислительной среде (ОВС), включающей в свой состав разнотипные (гетерогенные) вычислительные ресурсы, при решении потока крупномасштабных научных задач, поступающих в произвольные моменты времени и состоящих из множества информационно взаимосвязанных подзадач. Предложенный ранее метод мультиагентного диспетчирования ресурсов ОВС развивается для наиболее общего случая, когда все ресурсы ОВС имеют как различную производительность при решении тех или иных задач, так и различную пропускную способность канала связи с облачной инфраструктурой. Дана формальная постановка задачи диспетчирования ОВС при таких условиях и предложены принципы ее решения с помощью множества программных агентов, физически реализуемых на отдельных ресурсах ОВС и представляющих их "интересы" в процессе диспетчирования. Разработан алгоритм работы программного агента мультиагентного диспетчера, обеспечивающий адаптивное распределение всех свободных в текущий момент времени вычислительных ресурсов ОВС по задачам потока с учетом их реальной производительности и пропускной способности каналов связи, приводятся результаты его экспериментальных исследований с помощью программной модели гетерогенной ОВС.

Ключевые слова: облачная вычислительная среда, гетерогенные вычислительные ресурсы, поток крупномасштабных задач, мультиагентное диспетчирование, программный агент, адаптивное распределение ресурсов.

Введение

В настоящее время уровень развития телекоммуникационных технологий открывает новые возможности организации распределенных вычислений на основе множества пространственно удаленных вычислительных ресурсов с помощью сервис-ориентированной инфраструктуры, обеспечивающей "вычисления по требованию". Эти возможности породили новую парадигму облачных вычислений, т. е. крупномасштабных распределенных вычислений на основе пула абстрактных, виртуализованных, динамически перераспределяемых вычислительных ресурсов, предоставляемых по запросу внешним пользователям через Интернет [1]. Облачные вычислительные среды (ОВС) находят все

более широкое применение для решения крупномасштабных научных и технических задач в различных предметных областях, например, таких как физика высоких энергий, науки о земле, химия и биология, космология и астрофизика, экологическая и техногенная безопасность, промышленность, фармакология и фармацевтика, материаловедение, нефтегазодобыча, медицина и т. п. [2–4]. Такие крупномасштабные научные задачи характеризуются сложной внутренней структурой, состоящей, как правило, из ряда информационно взаимосвязанных подзадач, причем эффективность решения которых зачастую в сильной степени зависит от типа используемого вычислительного ресурса.

Одним из основных преимуществ концепции облачных вычислений является возможность использования в составе ОВС разнотипных (гетерогенных) вычислительных ресурсов, которые имеют различную реальную производительность при решении тех или иных задач. Это обстоятельство, с одной стороны, открывает возможность повышения эффективности (сокращения времени) выполнения крупномасштабных пользовательских задач за счет использования тех вычислительных ресурсов ОВС, которые обеспечивают максимальную реальную производительность при их решении (или решении их подзадач), но, с другой стороны, порождает проблему оптимального распределения (диспетчирования) решаемых задач (или их подзадач) по разнотипным вычислительным ресурсам, входящим в состав ОВС. Данная проблема многократно усложняется в случае, если ОВС должна обеспечивать решение не одиночной задачи, а некоторого множества (потока) разнородных пользовательских задач, поступающих в произвольные моменты времени. Все это требует разработки новых методов и алгоритмов адаптивного диспетчирования ресурсов гетерогенных ОВС при обработке потока крупномасштабных задач.

В работе [5] предложен метод мультиагентного диспетчирования ресурсов гетерогенной ОВС, обеспечивающий квазиоптимальное автоматическое распределение ресурсов ОВС в процессе решения потока пользовательских задач с учетом их реальной производительности. Однако при построении данного метода не учитывался тот факт, что различные ресурсы, входящие в состав ОВС, могут иметь также различную пропускную способность каналов связи с облачной инфраструктурой, что в общем случае снижает эффективность его применения. Поэтому в настоящей работе предлагается расширение метода, предложенного в [5], на случай, когда ресурсы гетерогенной ОВС имеют как различную производительность, так и различную пропускную способность канала связи с облачной инфраструктурой.

1. Постановка задачи

Проблему диспетчирования ОВС можно формализовать в следующем виде. Предположим, что ОВС должна решать некоторое множество (поток) различных крупномасштабных задач $\mathbf{Z} =< Z_1, Z_2, \ldots, Z_M >$, которые могут поступать от пользователей в случайные моменты времени. При этом под крупномасштабной задачей будем понимать задачу, состоящую из некоторого множества информационно зависимых (взаимосвязанных) подзадач, каждая из которых имеет значительную вычислительную трудоемкость. Формально каждая такая крупномасштабная задача $Z_l \in \mathbf{Z}$ может быть представлена в виде некоторого графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ (рис. 1), вершины $q_j \in \mathbf{Q}_l$ которого соответствуют некоторым подзадачам A_j, принадлежащим множеству подзадач $\mathbf{A} =< A_1, A_2, \ldots, A_c >$, а дуги $x(q_j, q_{j+1}) \in \mathbf{X}_l$ определяют информационные взаимосвязи между подзадачами (т. е. если две вершины q_j и q_{j+1} соединены дугой $x(q_j, q_{j+1})$,

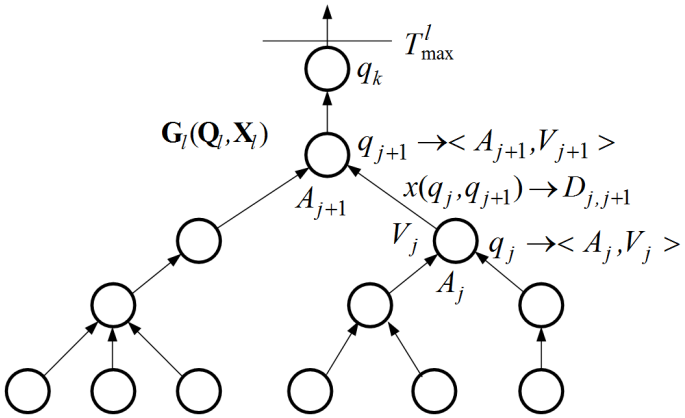

Рис. 1. Граф $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ задачи Z_l

то это означает, что данные, полученные в результате решения подзадачи A_j, приписанной вершине q_j, необходимы для выполнения подзадачи A_{j+1}, приписанной вершине q_{j+1}). Будем считать, что каждой вершине $q_j \in \mathbf{Q}_l$ приписан тип решаемой подзадачи $A_j \in \mathbf{A}$, а также определена ее вычислительная трудоемкость V_j, оцениваемая числом элементарных вычислительных операций, выполняемых при ее решении, а дуге $x(q_j, q_{j+1}) \in \mathbf{X}_l$ приписан объем данных $D_{j,j+1}$, передаваемых от подзадачи A_j, приписанной вершине q_j, подзадаче A_{j+1}, приписанной вершине q_{j+1} (рис. 1). Кроме того, положим, что пользователь устанавливает и приписывает конечной вершине q_k графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ момент времени $T^l_{\max}$, к которому он желает получить результат решения своей задачи Z_l, а также определяет количество премиальных баллов (бонусов) O, которое он готов заплатить за это.

Пусть в состав ОВС входит множество вычислительных ресурсов $\mathbf{R} =< R_1, R_2, \ldots, R_N >$. При этом под вычислительным ресурсом будем понимать некоторое вычислительное устройство (сервер, многопроцессорную вычислительную систему и т. п.), подключенное к облачной инфраструктуре. Будем считать, что каждый ресурс $R_i \in \mathbf{R}$ может решать некоторый набор (подмножество) подзадач $\mathbf{A}_i =< A^i_1, A^i_2, \ldots A^i_L > \subseteq \mathbf{A}$ $(i = 1, 2, \ldots, N)$, причем реальная производительность ресурса R_i при решении подзадачи $A^i_j \in \mathbf{A}_i$ составляет $S^i_j = F(A^i_j)$ $(j = 1, 2, \ldots, L)$ операций в секунду. Кроме того, пропускная способность канала связи ресурса $R_i \in \mathbf{R}$ с облачной инфраструктурой составляет Y_i байт/с.

Целью работы диспетчера ОВС является такое распределение подзадач потока задач $\mathbf{Z}$ по вычислительным ресурсам $R_1, R_2, \ldots, R_N$, при котором все задачи $Z_l \in \mathbf{Z}$ будут выполнены к установленным пользователями моментам времени $T^l_{\max}$.

До недавнего времени проблема диспетчирования ОВС решалась, как правило, посредством специально выделенных серверных узлов, функции которых заключались в приеме пользовательских задач и их распределении по вычислительным ресурсам ОВС [6,7] (в качестве примера можно привести проект “Корпоративное облако СО РАН” [8]). Однако такая централизованная организация диспетчера имеет ряд недостатков. Во-первых, при большом числе разнородных ресурсов в ОВС, отличающихся производительностью и пропускной способностью каналов связи, их оперативное распределение по задачам (а тем более по отдельным информационно связанным подзадачам) с помощью одного центрального диспетчера (или дерева диспетчеров) проблематично. Данная проблема многократно усложняется в случае, если ОВС должна решать не одиночную

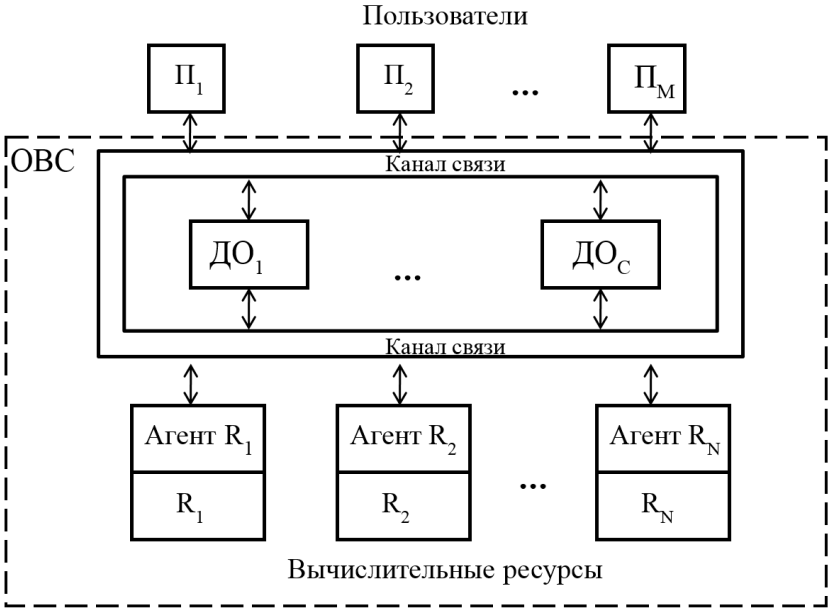

Рис. 2. Структура ОВС с мультиагентным диспетчером

задачу, а поток пользовательских задач, поступающих в заранее неизвестные моменты времени. Во-вторых, существенно затрудняется масштабирование ОВС (т. е. добавление в ее состав новых вычислительных ресурсов), что требует внесения изменений в алгоритмы работы диспетчера. В-третьих, ОВС с централизованным диспетчером имеет низкую отказоустойчивость, поскольку выход из строя служебного серверного узла, реализующего функции диспетчера, приводит к катастрофическим последствиям для всей ОВС в целом.

Указанные недостатки устраняются при использовании принципов мультиагентного диспетчирования ресурсов в гетерогенных ОВС [9 – 12]. В качестве основных активных элементов ОВС предлагается использовать программные агенты, физически реализуемые на каждом вычислительном ресурсе $R_i \in \mathbf{R}$ $(i = 1, N)$, входящем в состав ОВС, и представляющие его "интересы" в процессе диспетчирования. Взаимодействие пользователей и агентов осуществляется посредством некоторого множества специально выделенных пассивных узлов, составляющих служебный уровень облачной инфраструктуры и выполняющих функции досок объявлений (ДО) (рис. 2).

Основное преимущество данного подхода к диспетчированию ресурсов в ОВС заключается в том, что вычислительный процесс постоянно адаптируется к вычислительным возможностям ресурсов, задействованных в ее составе. Кроме того, по сравнению с классической, централизованной организацией диспетчера облачной среды в данном случае упрощаются требования к служебным серверам (доскам объявлений), что позволяет существенно снизить накладные расходы на организацию облачной среды, а также упростить процесс ее масштабирования, что, как следствие, ведет к снижению стоимости облачных вычислений.

Однако алгоритмы мультиагентного диспетчирования, предложенные в [5], не учитывают объемы передаваемой между подзадачами информации, а также пропускную способность каналов связи отдельных ресурсов ОВС. Это обстоятельство существенно ухудшает качество диспетчирования и соответственно снижает эффективность работы ОВС в целом.

2. Принципы мультиагентного диспетчирования ресурсов ОВС с учетом их коммуникационных возможностей

Прежде чем приступить к разработке нового алгоритма мультиагентного диспетчирования ОВС, лишенного указанных выше недостатков, расширим понятие “нити”, введенное в [5]. Как и раньше, под нитью будем понимать некоторую последовательность вершин $\mathbf{H}_f =< q_1^f, q_2^f, \ldots, q_k^f >$ графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ задачи $Z_l \in \mathbf{Z}$, в которой вершины q_j^f и q_{j+1}^f $(j = 1, 2, \ldots, k-1)$ соединены дугой $x(q_j^f, q_{j+1}^f)$ (рис. 3). Иными словами, нить определяет некоторую последовательность подзадач задачи Z_l, в которой каждая последующая подзадача использует в качестве исходных данных результат выполнения предыдущей подзадачи. Очевидно, что подзадачи, приписанные вершинам нити $\mathbf{H}_f$, могут выполняться только последовательно. При этом под длиной t_f^p нити $\mathbf{H}_f$ будем понимать суммарное время, затрачиваемое на ее решение, определяемое как

$$t_f = \sum_{i=1}^{k} \left(\frac{V_i^f}{S_i^p} + t_{i,i+1}^f \right),$$

где V_i^f — вычислительная трудоемкость подзадачи A_i, приписанной вершине $q_i^f (i = 1, 2, \ldots, k)$ нити $\mathbf{H}_f$; S_i^p — производительность ресурса R_p, решающего подзадачу A_i; $t_{i,i+1}^f$ — время передачи данных, полученных в результате выполнения подзадачи A_i, приписанной вершине q_i^f, подзадаче A_{i+1}, приписанной смежной вершине q_{i+1}^f.

Будем считать, что

$$t_{i,i+1}^f = \begin{cases} 0, & \text{если подзадачи } A_i \text{ и } A_{i+1} \text{ решаются одним и тем же ресурсом } R_p \in \mathbf{R}, \\ \dfrac{D_{i,i+1}^f}{Y_p}, & \text{если подзадачи } A_i \text{ и } A_{i+1} \text{ решаются различными ресурсами } R_p \text{ и } R_c, \end{cases}$$

где $D_{i,i+1}^f$ — объем данных, передаваемых от подзадачи A_i подзадаче A_{i+1}; Y_p — пропускная способность канала связи ресурса R_p.

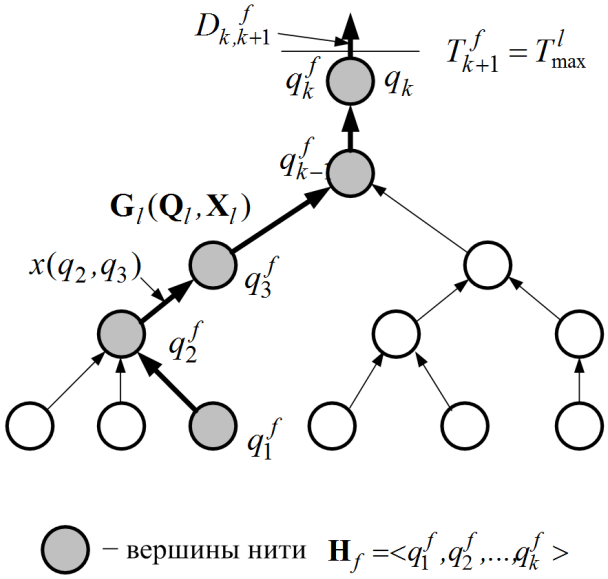

Рис. 3. Выделение нити $\mathbf{H}_f$ в графе $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ задачи Z_l

Тогда, очевидно, если вся нить $\mathbf{H}_f$ выполняется одним ресурсом R_p^f, то ее длина будет составлять

$$t_f^p = \sum_{i=1}^{k} \frac{V_i^f}{S_i^p} + \frac{D_{k,k+1}^f}{Y_p}, \tag{1}$$

где $D_{k,k+1}^f$ — объем данных, приписанный исходящей (конечной) дуге нити $\mathbf{H}_f$ (рис. 3).

При этом, если известен требуемый момент времени T_{k+1}^f исполнения всей нити $\mathbf{H}_f$, то допустимые моменты времени T_d^f начала выполнения всех подзадач A_d, приписанных ее вершинам $q_d^f \in \mathbf{H}_f$ $(d = 1, 2, \ldots, k)$, при которых ресурс R_p успевает выполнить всю нить $\mathbf{H}_f$ к требуемому моменту времени T_{k+1}^f, можно определить как

$$T_d^f = T_{k+1}^f - \left(\sum_{i=d}^{k} \frac{V_i^f}{S_i^p} + \frac{D_{k,k+1}^f}{Y_p} \right), \quad d = 1, 2, \ldots, k-1. \tag{2}$$

Исходя из этих соображений, можно предложить следующую процедуру мультиагентного диспетчирования ресурсов $R_1, R_2, \ldots, R_N$, входящих в состав гетерогенной ОВС, с учетом их вычислительных и коммуникационных возможностей при решении потока задач $\mathbf{Z}$.

Пользователь формирует свою задачу $Z_l \in \mathbf{Z}$ в виде графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ и определяет требуемый момент времени $T_{\max}^l$, к которому ее решение должно быть получено, а также величину бонусов (премиальных баллов) O, которую он готов заплатить за ее решение. Представленный таким образом дескриптор задачи $Z_l \in \mathbf{Z}$ размещается на одной из досок объявлений (см. рис. 2).

Агенты, ресурсы которых не задействованы в решении каких-либо задач, обращаются к ДО в поисках работы. Если агент свободного ресурса $R_p \in \mathbf{R}$ обнаруживает на ДО дескриптор задачи Z_l, то он предпринимает попытку войти в сообщество (подмножество) ресурсов $\mathbf{R}_l \subseteq \mathbf{R}$, задействованных в ее решении.

Поскольку, как мы приняли выше, каждый ресурс ОВС имеет некоторую специализацию (т. е. решает ограниченный набор подзадач), может оказаться, что ресурс R_p способен выполнять далеко не все подзадачи, приписанные вершинам графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ задачи Z_l. Поэтому в графе $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ необходимо предварительно выделить подграф $\mathbf{G}_l^p(\mathbf{Q}_l^p, \mathbf{X}_l^p)$, вершинам которого приписаны подзадачи множества $\mathbf{A}_p$, выполняемые ресурсом R_p (рис. 4). После этого необходимо проанализировать, есть ли в графе $\mathbf{G}_l^p(\mathbf{Q}_l^p, \mathbf{X}_l^p)$ вершины, которым приписано требуемое время их исполнения. Отметим, что изначально, в момент размещения дескриптора задачи Z_l на ДО, требуемое время исполнения $T_{\max}^l$ приписано только конечной вершине q_k графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ (см. рис. 3). Если таковых вершин нет, то агент R_p пока что не может вступить в сообщество $\mathbf{R}_l$ по выполнению задачи Z_l и поэтому он вновь переходит к режиму опроса ДО с целью поиска других задач.

В противном случае агент R_p выделяет в графе $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$, хранящемся в дескрипторе задачи Z_l на ДО, наиболее длинную нить $\mathbf{H}_1 =< q_1^1, q_2^1, \ldots, q_k^1 >$ согласно выражению (1), конечной вершине q_k^1 которой приписан момент времени ее исполнения T_{k+1}^1 (см. рис. 4). Последнее может быть осуществлено с помощью одного из известных алгоритмов поиска экстремальных путей на графах [11, 12].

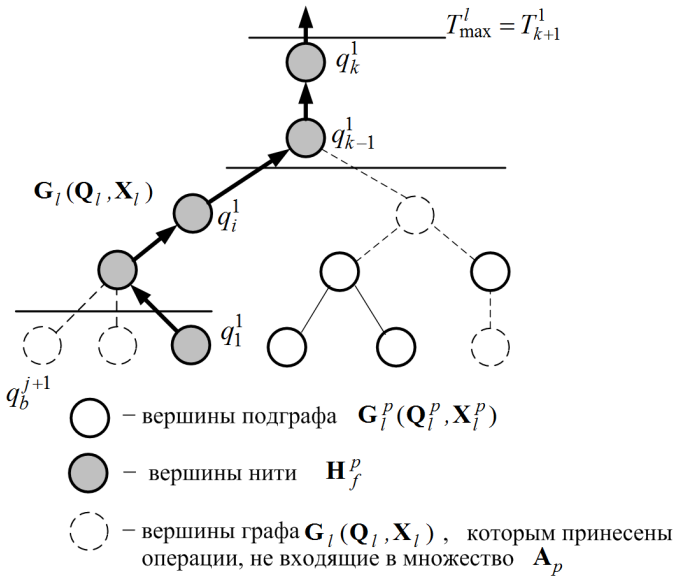

Рис. 4. Выделение подграфа $\mathbf{G}^p_l(\mathbf{Q}^p_l, \mathbf{X}^p_l)$ в графе $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ задачи Z_l

Далее агент R_p определяет согласно (2) момент времени T^1_1, когда ему необходимо приступить к выполнению первой подзадачи нити $\mathbf{H}_1$, т. е. подзадачи A^1_1, приписанной вершине $q^1_1 \in \mathbf{H}_1$, чтобы успеть завершить исполнение всей нити $\mathbf{H}_1$ к заданному моменту времени T^1_{k+1}.

Если при этом оказывается, что $T^1_1 < T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, то это означает, что ресурс R_p не может обеспечить выполнение всей последовательности подзадач нити $\mathbf{H}_1 =< q^1_1, q^1_2, \ldots, q^1_k >$ к заданному моменту времени T^1_{k+1}. Однако, поскольку мы приняли, что все ресурсы ОВС имеют различную производительность при решении идентичных подзадач, это вовсе не означает, что вся задача Z_l невыполнима. Действительно, через какое-то время может появиться другой свободный ресурс $R_c \in \mathbf{R}$, который сможет выполнить данную нить к требуемому моменту времени.

Поэтому в случае, если $T^1_1 < T_{\text{тек}}$, агент R_p должен выделить в графе заданий следующую по длине нить, конечной вершине которой также приписано требуемое время ее исполнения, и проанализировать возможность ее исполнения к данному моменту времени и т. д., до тех пор, пока он не найдет нить, которую “его” ресурс может выполнить за отведенное время, либо безуспешно переберет все нити графа $\mathbf{G}^p_l(\mathbf{Q}^p_l, \mathbf{X}^p_l)$ в порядке убывания их длины.

Отметим, поскольку в момент размещения задачи Z_l на ДО требуемый момент времени исполнения $T^l_{\max}$ приписан только конечной вершине q_k графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ (см. рис. 3), процесс распределения подзадач этой задачи между ресурсами ОВС не “сдвинется с места” до тех пор, пока не появится свободный ресурс, способный выполнить нить $\mathbf{H}_1 =< q^1_1, q^1_2, \ldots, q^1_k >$, заканчивающуюся на конечной вершине q_k графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$, к моменту времени $T^1_{k+1} = T^l_{\max}$.

Если в результате такого перебора агент R_p не обнаружит нить, которую он сможет выполнить к установленному моменту времени с помощью своего ресурса, то он вновь переходит к процедуре опроса досок объявлений с целью поиска работы.

Если же условие $T^1_1 \geq T_{\text{тек}}$ выполняется для некоторой нити $\mathbf{H}_1$, то в этом случае агент R_p принимает на себя исполнение последовательности подзадач, приписанных

ее вершинам. При этом агент R_p осуществляет модификацию дескриптора задачи Z_l на ДО, а именно:

1) его идентификатор записывается в список членов сообщества $\mathbf{R}_l$ по решению задачи Z_l;
2) вершины, входящие в нить $\mathbf{H}_1$, исключаются из графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ задачи Z_l, в результате чего формируется новый граф $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1) = \mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)/\mathbf{H}_1$ (рис. 5);
3) всем вершинам графа $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$, инцидентным вершинам нити $\mathbf{H}_1$, приписываются требуемые моменты времени их исполнения, которые определяются исходя из следующих соображений.

Допустим, что некоторая вершина q_f^2 графа $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$ инцидентна вершине q_b^1, принадлежащей нити $\mathbf{H}_1$ (рис. 5). Это означает, что для выполнения подзадачи, приписанной вершине q_b^1, необходимы результаты выполнения подзадачи, приписанной вершине q_f^2 графа $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$. Поэтому очевидно, что результаты выполнения подзадачи, приписанной вершине q_f^2, должны быть получены и переданы агенту R_p, выполняющему подзадачи нити $\mathbf{H}_1$, не позже, чем к требуемому моменту времени T_b^1 начала выполнения агентом R_p подзадачи A_b^1, приписанной вершине q_b^1, определяемому согласно выражению (2) как

$$T_b^1 = T_{k+1}^1 - \left(\sum_{i=b}^{k} \frac{V_i^1}{S_i^p} + \frac{D_{k,k+1}^f}{Y_p} \right). \tag{3}$$

В противном случае ресурс R_p не успеет закончить исполнение взятой на себя нити $\mathbf{H}_1$ к требуемому моменту времени T_{k+1}^1. Поэтому вершине q_f^2 графа $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$ приписывается требуемое время ее исполнения $T_{f+1}^2 = T_b^1$, а также идентификатор агента R_p, которому результаты исполнения подзадачи, приписанной вершине q_f^2, должны быть переданы (рис. 5).

Аналогичным образом определяются требуемые моменты T_{m+1}^2 исполнения всех остальных вершин графа $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$, инцидентных вершинам нити $\mathbf{H}_1$. Если после модификации новый граф $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$ задачи Z_l еще не пустой, т. е. $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1) = \varnothing$, то процесс создания сообщества $\mathbf{R}_l$ для выполнения задачи Z_l продолжается.

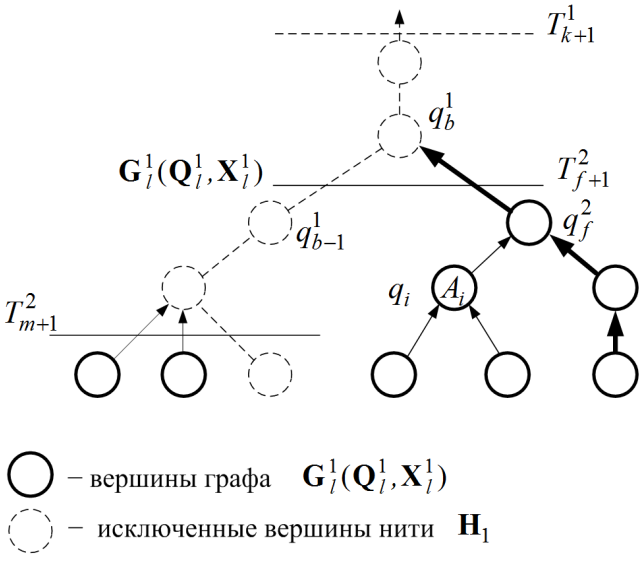

Рис. 5. Граф $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$ задачи Z_l, модифицированный агентом R_p

Допустим, что через какое-то время другой свободный агент R_c обнаруживает на ДО дескриптор задачи Z_l и предпринимает попытку войти в состав сообщества $\mathbf{R}_l$ по ее решению. Для этого агент R_c выделяет в графе $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$ подграф $\mathbf{G}_l^{1c}(\mathbf{Q}_l^{1c}, \mathbf{X}_l^{1c})$, вершинам которого приписаны подзадачи, входящие в исполняемое им множество A_A (рис. 6). Далее в графе $\mathbf{G}_l^{1c}(\mathbf{Q}_l^{1c}, \mathbf{X}_l^{1c})$ агент R_c выделяет наиболее длинную нить $\mathbf{H}_2 =< q_1^2, q_2^2, \ldots, q_0^2 >$, конечной вершине q_f^2 которой приписано требуемое время исполнения T_{f+1}^2, и анализирует возможность ее исполнения к данному моменту времени. Для этого он с помощью выражения (2) определяет время T_1^2 начала исполнения подзадачи A_1^2, приписанной первой вершине q_1^2 данной нити $\mathbf{H}_2$, и сравнивает его с текущим временем $T_{\text{тек}}$. Если $T_{\text{тек}} > T_1^2$, то это означает, что ресурс R_c не может выполнить данную нить к требуемому моменту времени T_{f+1}^2. Поэтому агент R_c продолжает анализировать следующие по убыванию длины нити графа $\mathbf{G}_l^{1c}(\mathbf{Q}_l^{1c}, \mathbf{X}_l^{1c})$ (для которых определено требуемое время исполнения) до тех пор, пока он не найдет нить, которую он может выполнить к требуемому моменту времени, либо не переберет все возможные нити графа $\mathbf{G}_l^{1c}(\mathbf{Q}_l^{1c}, \mathbf{X}_l^{1c})$, после чего он снова переходит в режим опроса досок объявлений в поисках работы для “своего” ресурса.

В случае же, если агент R_c обнаруживает нить $\mathbf{H}_2$, для которой выполняется условие $T_1^2 \geq T_{\text{тек}}$, то он принимает на себя исполнение подзадач, приписанных вершинам данной нити, и осуществляет очередную модификацию дескриптора задания Z_l на ДО:

- идентификатор агента R_c записывается в список членов сообщества $\mathbf{R}_l$ по решению задачи Z_l;
- вершины, входящие в нить $\mathbf{H}_2$, исключаются из графа $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$, в результате чего образуется новый граф $\mathbf{G}_l^2(\mathbf{Q}_l^2, \mathbf{X}_l^2)$ (рис. 7);
- вершинам q_p^3 графа $\mathbf{G}_l^2(\mathbf{Q}_l^2, \mathbf{X}_l^2)$, инцидентным вершинам q_d^2 нити $\mathbf{H}_2$, приписывается идентификатор агента R_c, которому результаты исполнения этих подзадач должны быть направлены, а также требуемое время их исполнения (рис. 7), определяемое как

$$T_{p+1}^3 = T_{f+1}^2 - \left(\sum_{i=d}^{f} \frac{V_i^2}{S_i^c} + \frac{D_{f,f+1}^2}{Y_c} \right).$$

Далее в процесс распределения подзадач задачи Z_l включается следующий свободный агент, обнаруживший ее дескриптор на ДО, и т. д. до тех пор, пока не окажется,

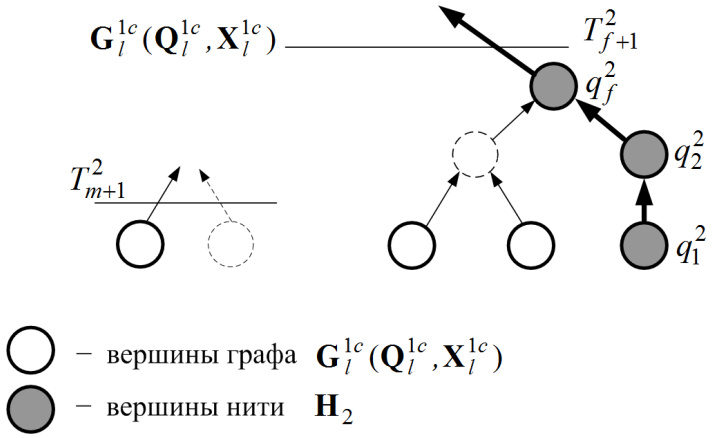

Рис. 6. Выделение подграфа $\mathbf{G}_l^{1c}(\mathbf{Q}_l^{1c}, \mathbf{X}_l^{1c})$ в графе $\mathbf{G}_l^1(\mathbf{Q}_l^1, \mathbf{X}_l^1)$

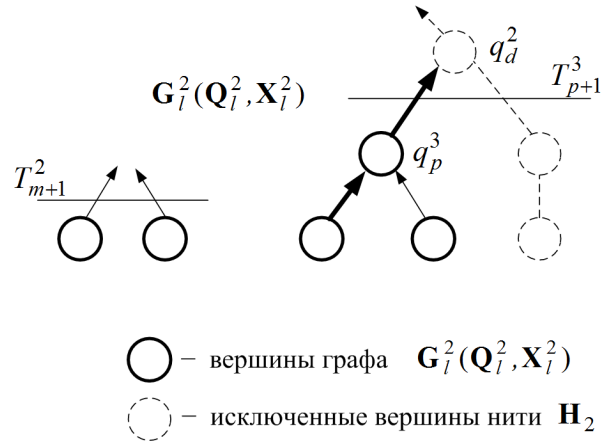

Рис. 7. Граф $\mathbf{G}_l^2(\mathbf{Q}_l^2, \mathbf{X}_l^2)$, модифицированный агентом R_c

что после очередной модификации граф $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ стал пустым. Это означает, что все подзадачи задачи Z_l разобраны агентами, вошедшими в сообщество $\mathbf{R}_l$ по его выполнению.

После того как некоторый агент R_p выбрал для исполнения нить $\mathbf{H}_f = < q_1^f, q_2^f, \ldots, q_k^f >$, он приступает к реализации подзадач, приписанных ее вершинам. Перед началом выполнения очередной подзадачи A_d, приписанной вершине $q_d^f \in \mathbf{H}_f$ $(d = 1, 2, \ldots, k)$, агент R_p должен проверить, во-первых, наличие всех исходных данных, необходимых для ее выполнения, а во-вторых, соблюдение временного графика выполнения всей нити $\mathbf{H}_f$ в целом.

Поскольку для выполнения подзадачи A_d могут требоваться исходные данные, получаемые в результате выполнения смежной нити другим агентом R_c, к моменту начала выполнения агентом R_p подзадачи A_d может оказаться, что эти данные еще не поступили. В этом случае агент R_p должен перейти в режим ожидания поступления необходимых исходных данных. Это ожидание может продолжаться до тех пор, пока $T_d^f \geq T_{\text{тек}}$, где T_d^f — требуемое время начала решения подзадачи A_d, приписанной вершине $q_d^f \in \mathbf{H}_f$ и определяемое согласно выражению (2).

Если $T_d^f < T_{\text{тек}}$, это означает, что ресурс R_p уже не успевает выполнить оставшиеся подзадачи нити $\mathbf{H}_f$ к требуемому моменту времени T_{k+1}^f. Однако, поскольку мы приняли, что производительность всех ресурсов при решении идентичных подзадач разная, в принципе в дальнейшем может появиться свободный ресурс, который успеет выполнить оставшуюся часть нити $\mathbf{H}_f$ к требуемому моменту времени T_{k+1}^f. Поэтому агент R_p должен известить ДО о том, что произошло отставание от графика вычислительного процесса и он не может обеспечить решение принятой на себя нити $\mathbf{H}_f$ к заданному моменту времени. При этом данные, полученные ресурсом R_p в результате решения подзадачи A_{d-1}, приписанной предыдущей вершине q_{d-1}^f нити $\mathbf{H}_f$, отправляются агентом R_p на ДО, а дескриптор задачи модифицируется следующим образом: идентификатор агента R_p исключается из списка членов сообщества $\mathbf{R}_l$, а оставшиеся нереализованными вершины $< q_{d+}^f, q_{d+1}^f, \ldots, q_k^f >$ нити $\mathbf{H}_f$ возвращаются в граф задачи, причем конечной вершине q_k^f этой нити вновь приписываются значение момента времени T_{k+1}^f, к которому необходимо получить результат ее решения, и идентификатор агента, которому этот результат необходимо передать. Модифицированный таким об-

разом дескриптор задачи Z_l вновь выставляется на ДО для поиска новых участников сообщества, которые смогли бы обеспечить решение оставшейся части задачи к установленному пользователем моменту времени.

Если ресурс R_p успешно выполнил всю принятую на себя нить $\mathbf{H}_f$ задачи к установленному моменту времени, то агент R_p информирует об этом доску объявлений. При этом идентификатор агента R_p ресурса исключается из списка участников сообщества $\mathbf{R}_l$ и ему начисляются премиальные баллы за выполненную работу, величина которых определяется как $O_p = OV^p/V$. Здесь O — премиальные баллы (бонусы), гарантируемые пользователем за решение его задачи Z_l к требуемому моменту времени; V^p — суммарная трудоемкость подзадач, выполненных агентом R_p; V — суммарная трудоемкость всей задачи Z_l.

Процесс решения задачи Z_l в облачной вычислительной среде продолжается до тех пор, пока не окажется, что список агентов — участников сообщества $\mathbf{R}_l$ по ее решению пуст, а также пуст граф $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$, хранимый в дескрипторе задачи Z_l на ДО. Это означает, что все нити задачи успешно выполнены. После этого задача снимается с ДО, а пользователю отправляется сообщение об успешном завершении процесса решения его задачи, затем пользователь переводит оплату (премиальные баллы) всем агентам, принявшим участие в решении данной задачи, в зависимости от объема выполненной ими работы (суммарной трудоемкости решенных подзадач).

Описанному выше процессу отвечает следующий укрупненный алгоритм функционирования агента, представляющего ресурс R_p в ОВС.

Алгоритм

1. Агент свободного ресурса R_p опрашивает ДО.
2. При обнаружении на ДО задачи Z_l агент R_p считывает ее дескриптор и анализирует граф задачи $\mathbf{G}_l^j(\mathbf{Q}_l^j, \mathbf{X}_l^j)$. Если $\mathbf{G}_l^j(\mathbf{Q}_l^j, \mathbf{X}_l^j) = \emptyset$, то переход к п. 1.
3. Если $T_{\text{тек}} \geq T_{\max}^l$, где $T_{\text{тек}}$ — текущий момент времени, то переход к п. 22.
4. В графе $\mathbf{G}_l^j(\mathbf{Q}_l^j, \mathbf{X}_l^j)$ выделяется подграф $\mathbf{G}_l^{jp}(\mathbf{Q}_l^{jp}, \mathbf{X}_l^{jp})$, вершинам которого приписаны подзадачи множества $\mathbf{A}_p$, выполняемые ресурсом R_p.
5. Если $\mathbf{G}_l^{jp}(\mathbf{Q}_l^{jp}, \mathbf{X}_l^{jp}) = \emptyset$, то перейти к п. 1.
6. $i = 1$; $\mathbf{G}_i(\mathbf{Q}_i, \mathbf{X}_i) = \mathbf{G}_l^{jp}(\mathbf{Q}_l^{jp}, \mathbf{X}_l^{jp})$.
7. Агент R_p выделяет в графе $\mathbf{G}_i(\mathbf{Q}_i, \mathbf{X}_i)$ наиболее длинную нить $\mathbf{H}_i =< q_1^i, q_2^i, \ldots, q_k^i >$, конечной вершине которой приписано требуемое время исполнения T_{k+1}^i (в момент размещения задания Z_i на ДО требуемое время $T_{k+1}^i = T_{\max}^l$ приписано только конечной вершине q_k графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$). Если таковой нити в графе $\mathbf{G}_i(\mathbf{Q}_i, \mathbf{X}_i)$ нет, то перейти к п. 1.
8. Агент R_p определяет допустимый момент времени T_1^i, когда необходимо начать выполнение нити $\mathbf{H}_i =< q_1^i, q_2^i, \ldots, q_k^i >$, чтобы успеть завершить ее исполнение к моменту T_{k+1}^i, как
$$T_1^i = T_{k+1}^i - \left(\sum_{j=1}^{k} \frac{V_j^i}{S_j^p} + \frac{D_{k,k+1}^i}{Y_p} \right).$$
9. Если $T_1^i \geq T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, то переход к п. 13.
10. Нить $\mathbf{H}_i$ исключается из графа $\mathbf{G}_i(\mathbf{Q}_i, \mathbf{X}_i)$, т. е.
$$\mathbf{G}_{i+1}(\mathbf{Q}_{i+1}, \mathbf{X}_{i+1}) = \mathbf{G}_i(\mathbf{Q}_i, \mathbf{X}_i)/\mathbf{H}_i.$$

11. Если $\mathbf{G}_{i+1}(\mathbf{Q}_{i+1}, \mathbf{X}_{i+1}) = \emptyset$, то перейти к п. 1.
12. $i = i + 1$, перейти к п. 7.
13. Агент R_p принимает на себя исполнение нити $\mathbf{H}_i$ для чего:
    - считывает с ДО последовательность подзадач, приписанных вершинам нити $\mathbf{H}_i$, требуемое время ее исполнения T^i_{k+1} и идентификатор ресурса R_c, которому необходимо передать результаты ее решения;
    - модифицирует дескриптор задач Z_l на ДО. В список участников сообщества $\mathbf{R}_l$ записывается его идентификатор; из графа $\mathbf{G}_l(\mathbf{Q}_l, \mathbf{X}_l)$ исключаются вершины нити $\mathbf{H}_i$, в результате чего формируется новый граф $\mathbf{G}^{j+1}_l(\mathbf{Q}^{j+1}_l, \mathbf{X}^{j+1}_l) = \mathbf{G}^j_l(\mathbf{Q}^j_l, \mathbf{X}^j_l)/\mathbf{H}_i$; вершинам q^{i+1}_f модифицированного графа $\mathbf{G}^{j+1}_l(\mathbf{Q}^{j+1}_l, \mathbf{X}^{j+1}_l)$, инцидентным вершинам q^i_b нити $\mathbf{H}_i =< q^i_1, q^i_2, \ldots, q^i_k >$, приписываются идентификаторы ресурсов R_p, которым необходимо передать результаты исполнения подзадачи, соответствующей вершине q^{i+1}_f, а также требуемое время их исполнения, определяемое как

$$T^i_b = T^i_{k+1} - \left(\sum_{m=b}^{k} \frac{V^i_m}{S^p_m} + \frac{D^i_{k,k+1}}{Y_p} \right).$$

14. Агент R_p переходит к исполнению последовательности подзадач, приписанных вершинам нити $\mathbf{H}_i =< q^i_1, q^i_2, \ldots, q^i_k >$, $d = 1$.
15. Если $T^i_d < T_{\text{тек}}$, где $T^i_d = T^i_{k+1} - \left(\sum_{m=d}^{k} \frac{V^i_m}{S^p_m} + \frac{D^i_{k,k+1}}{Y_p} \right)$ — требуемое время начала выполнения подзадачи A_d, приписанной вершине $q^i_d \in \mathbf{H}_i$, то перейти к п. 17.
16. Агент R_p проверяет наличие исходных данных, необходимых для выполнения подзадачи A_d. Если исходные данные еще не поступили, то перейти к п. 15, иначе перейти к п. 18.
17. Агент R_p отправляет результаты решения подзадачи, приписанной вершине $q^i_{d-1} \in \mathbf{H}_i$, на ДО и осуществляет модификацию дескриптора задачи Z_l: исключает свой идентификатор из списка сообщества $\mathbf{R}_l$; добавляет неисполненные вершины q^i_d, $q^i_{d+1}, \ldots, q^i_k$ нити $\mathbf{H}_i$ в граф задачи; приписывает крайней вершине этой нити момент времени T^i_{k+1}, к которому она должна быть исполнена, а также идентификатор ресурса R_c, которому необходимо передать результаты ее решения. Переход к п. 1.
18. Агент R_p выполняет подзадачу A_d, приписанную вершине $q^i_d \in \mathbf{H}_i$, с помощью "своего" ресурса.
19. Если агенту R_p поступило сообщение о прекращении выполнения задачи Z_l, то переход к п. 1.
20. $d = d + 1$, если $d \le k$, то переход к п. 15.
21. Агент R_p сообщает на ДО об успешном завершении решения нити $\mathbf{H}_i$ задачи Z_l. При этом результаты решения передаются ресурсу R_c, идентификатор которого приписан конечной вершине данной нити, идентификатор агента R_p исключается из списка членов сообщества $\mathbf{R}_l$ по решению данной задачи, а ему начисляются премиальные баллы $O_p = OV^p/V$, где V^p — суммарная вычислительная трудоемкость подзадач, выполненных агентом R_p; V — суммарная трудоемкость всей задачи Z_l. Переход к п. 1.

22. Задача Z_l не может быть решена к установленному пользователем моменту времени. Дескриптор задачи Z_l удаляется с ДО, пользователю направляется сообщение о невозможности решения его задачи к требуемому моменту времени, а всем агентам, идентификаторы которых записаны в списке участников сообщества $\mathbf{R}_l$ по выполнению задачи Z_l, передается сообщение о прекращении ее исполнения. Переход к п. 1.

Результаты экспериментальных исследований

С целью исследования работоспособности и эффективности предложенного мультиагентного диспетчирования ресурсов гетерогенной ОВС при решении потока задач $\mathbf{Z}$ разработана программная модель. Программная модель обеспечивает возможность моделирования работы ОВС при различных значениях таких исходных параметров как:

- количество ресурсов в ОВС (до 1000 шт.);
- количество различных подзадач, выполняемых отдельным ресурсом, входящим в ОВС (до 20);
- производительность ресурсов при решении подзадач различных типов и пропускная способность их каналов связи с облачной инфраструктурой;
- граф пользовательской задачи (количество вершин; вычислительная трудоемкость подзадач, приписанных отдельным вершинам; объем передаваемых данных между подзадачами, приписанный дугам графа);
- требуемые моменты времени получения решения пользовательских задач и количество премиальных баллов, начисленных за их успешное решение.

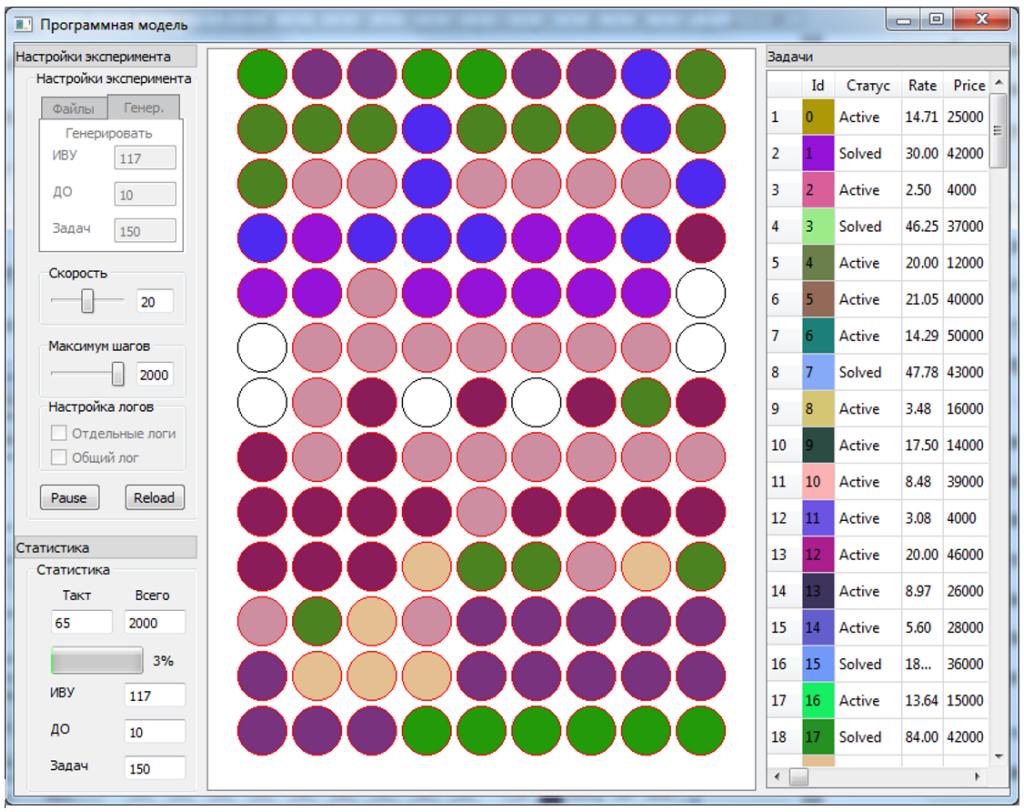

Рис. 8. Интерфейс программной модели

В качестве критериев эффективности работы ОВС при выполнении потока заданий были приняты:

- коэффициент полезного действия (КПД) – отношение времени, затраченного ресурсами ОВС на выполнение пользовательских задач, к общему времени их работы в системе;
- коэффициент гарантированности выполнения (КГВ) пользовательской задачи — отношение количества пользовательских задач, выполненных к требуемому моменту времени, к общему количеству задач, направленных пользователями на ДО.

На рис. 8 показан интерфейс программной модели ОВС, где в правой части отображаются параметры различных задач, размещенных пользователями на ДО, а в центральной части кружками изображены распределенные вычислительные ресурсы ОВС, причем цвет кружка показывает, в решении какой из задач в настоящее время участвует тот или иной ресурс.

Результаты серии экспериментов, проведенных при различных значениях исходных параметров программной модели, показывают, что значение КПД не опускается ниже 70 %, а значение КГВ — ниже 90 %.

Заключение

Описаны обобщенные принципы организации мультиагентного диспетчирования гетерогенной ОВС при выполнении потока потребительских заданий, поступающих в заранее неизвестные моменты времени, обеспечивающие:

- квазиоптимальное адаптивное распределение ресурсов ОВС с учетом их реальной производительности на той или иной подзадаче и пропускной способности каналов связи;
- высокую полезную загрузку ресурсов ОВС;
- возможность неограниченного наращивания (масштабируемости) числа гетерогенных вычислительных ресурсов в ОВС;
- повышенную отказоустойчивость ОВС, поскольку в ней отсутствуют узлы, выход из строя которых приводит к катастрофическим последствиям для системы в целом.

Благодарности. Работа выполнена при поддержке РФФИ (грант № 15-29-07928).

Список литературы / References

[1] **Kepes, B.** Understanding the Cloud Computing Stack: SaaS, PaaS, IaaS. Available at: https://support.rackspace.com/white-paper/understanding-the-cloud-computing-stack-saas-paas-iaas/

[2] Информационно-аналитический журнал Rational Enterprise Management/Рациональное управление предприятием. Круглый стол: Применение облачных технологий при организации ИТ-поддержки бизнеса промышленных компаний. Адрес доступа: http://www.remmag.ru/admin/upload_data/remmag/11-2/RoundTable.pdf
Information-Analytical Rational Enterprise Management magazine / Rational Enterprise Management. Round table: The use of cloud computing in the organization of IT support business for industrial companies. Available at:
http://www.remmag.ru/admin/upload_data/remmag/11-2/RoundTable.pdf (In Russ.)

[3] Журнал “Мир Телекома”. Обзор внедрения технологии облачных вычислений в регионах мира. Адрес доступа: http://mirtelecoma.ru/magazine/elektronnaya-versiya/29/
World Telecom Magazine. Review of the introduction of cloud computing technology in the regions of the world. Available at: http://mirtelecoma.ru/magazine/elektronnaya-versiya/29/ (In Russ.)

[4] “Наука и технологии России — strf.ru”. Облачные технологии накроют мир. Адрес доступа: http://www.strf.ru/material.aspx?CatalogId=223&d_no=31856#.VsMVlDN72UE/
“The science and technology of Russia — strf.ru”. Cloud technologies will cover the world. Available at: http://www.strf.ru/material.aspx?CatalogId=223&d_no=31856#.VsMVlDN72UE/ (In Russ.)

[5] **Каляев А.И., Каляев И.А., Коровин Я.С.** Метод мультиагентного диспетчирования ресурсов в гетерогенной облачной среде при выполнении потока задач // Вест. компьют. и информ. технологий. 2015. № 11. С. 31–40.
Kalyaev, A.I., Kalyaev, I.A., Korovin, Ya.S. Method of multiagent dispatching resources in heterogeneous cloud environments while performing flow of incoming tasks // Vestnik Komp'uternykh i Informatsionnykh Tekhnologii. 2015. No. 11. P. 31–40. (In Russ.)

[6] **Коннов А.Л.** Моделирование облачных технологий в вычислительных системах.
Адрес доступа: http://conference.osu.ru/assets/files/conf_reports/conf9/436.doc
Konnov, A.L. Modelling of cloud computing systems.
Available at: http://conference.osu.ru/assets/files/conf_reports/conf9/436.doc (In Russ.)

[7] IBM developerWorks. Основы облачных вычислений.
Адрес доступа: https://www.ibm.com/developerworks/ru/library/cl-cloudintro/
IBM developerWorks. Fundamentals of cloud computing. Available at:
https://www.ibm.com/developerworks/ru/library/cl-cloudintro/ (In Russ.)

[8] Облако СО РАН. Адрес доступа: http://cloud.sbras.ru/ru/about
About “Corporate cloud SB RAS”. Available at: http://cloud.sbras.ru/ru/about (In Russ.)

[9] **Kalyaev, A.I.** Multiagent approach for building distributed adaptive computing system // Procedia Comput. Sci. 2013. Vol. 18. P. 2193–2202.
Available at: http://dx.doi.org/10.1016/j.procs.2013.05.390

[10] **Каляев А.И.** Децентрализованная организация диспетчера **G**RID на базе сообществ агентов // Изв. ЮФУ. Технические науки. 2011. Т. 121, № 8. С. 230–238. Адрес доступа: http://elibrary.ru/item.asp?id=16562793
Kalyaev, A.I. Decentralised organization of dispatcher of grid based on agents comunities // Izvestiya SFedU. Engineering sciences. 2011. Vol. 121, No. 8. P. 230–238. Available at: http://elibrary.ru/item.asp?id=16562793 (In Russ.)

[11] **Каляев А.И.** Метод и алгоритмы адаптивной организации распределенных вычислений в децентрализованной **G**RID // Вест. компьют. и информ. технологий. 2012. № 4. С. 28–33. Адрес доступа: http://elibrary.ru/item.asp?id=17681405
Kalyaev, A.I. Method and algorithms of the adaptive organization for distributed calculations in decentralized **G**RID // Vestnik Komp'uternykh i Informatsionnykh Tekhnologii. 2012. No. 4. P. 28–33. Available at: http://elibrary.ru/item.asp?id=17681405 (In Russ.)

[12] **Kalyaev, A.I., Korovin, Ya.S.** Adaptive multiagent organization of the distributed computations // AASRI Procedia. 2014. Vol. 6. P. 49–58.
Available at: http://dx.doi.org/10.1016/j.aasri.2014.05.008

[13] **Cormen, T.H., Leiserson, Ch.E., Rivest, R.L., Stein, C.** Introduction to algorithms. Second Edition. The MIT Press, 2009. 1312 p.

[14] **Рейнгольд Э., Нивергельт Ю., Део Н.** Комбинаторные алгоритмы. Теория и практика. М.: Мир, 1980. 476 с.
Reyngold, E., Nivergelt, Yu., Deo, N. Combinatorial algorithms. Theory and practice. Moscow: Mir, 1980. 476 p. (In Russ.)

Поступила в редакцию 25 декабря 2015 г.,
с доработки — 23 марта 2015 г.

Algorithm of multi-agent dispatching resources in heterogeneous cloud environments

Kalyaev, Igor A.[1,2], Kalyaev, Anatoly I.[2,*], Korovin, Yakov S.[2]

[1]Southern Federal University, Rostov-on-Don, 344006, Russia
[2]Scientific Research Institute of Multiprocessing Computing and Control Systems SFU, Taganrog, 347900, Russia
[*]Corresponding author: Kalyaev, Anatoly I., e-mail: anatoly@kalyaev.net

The work is devoted to a solution for the problem of adaptive scheduling (allocation) of resources in a cloud computing environment (CCE), which includes heterogeneous computing resources in its set. The environment is designed for solving large-scale scientific problems on a flow entering at random times, and consisting of a set of interconnected information subtasks. In the present work, the previously proposed by authors method of multi-agent scheduling CCE is extended to general case when all resources of the CCE have different performance for solution of various problems, and various bandwidth communication channel with the cloud infrastructure. Under these conditions, the formal staging for problem scheduling in CCE is given and the authors suggest principles to solve it using a variety of software agents, which are physically implemented in the individual computing resources and representing their "interests" in the process of scheduling. We also propose the algorithm for multi-agent software agent manager, which provides adaptive distribution of all available at the current time computer resources for solution of the problems related to their actual performance and bandwidth of communication channels. The results of experimental studies using this heterogeneous programming model have shown the efficiency of proposed approaches.

Keywords: cloud computing environment, heterogeneous computing resources, a decentralized controller, multi-agent dispatching, flow of large-scale tasks, adaptive resource allocation.

Acknowledgements. This work was supported by RFBR grant No. 15-29-07928.

Received 25 December 2015
Received in revised form 23 March 2016