# *Робастное, адаптивное и сетевое управление*

**© 2022 г. И.А. КАЛЯЕВ, д-р техн. наук, академик РАН (igor@kalyaev.net),**
**А.И. КАЛЯЕВ, канд. техн. наук (anatoly@kalyaev.net)**
**(Южный федеральный университет, Таганрог)**

## МЕТОД И АЛГОРИТМЫ АДАПТИВНОГО МУЛЬТИАГЕНТНОГО ДИСПЕТЧИРОВАНИЯ РЕСУРСОВ В ГЕТЕРОГЕННЫХ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СРЕДАХ[1]

Настоящая статья посвящена созданию новых метода и алгоритмов диспетчирования ресурсов в гетерогенных распределенных вычислительных системах на примере облачных вычислительных сред (ОВС), обеспечивающих сокращение времени выполнения множества поступающих задач за счет использования тех вычислительных ресурсов, которые дают наиболее высокую реальную производительность применительно к конкретной полученной задаче. Для этого предлагается применять мультиагентный подход к организации процесса диспетчирования: в состав каждого элемента ОВС вводится программный агент, располагающий наиболее полной и актуальной информацией об особенностях своего вычислителя, множество таких агентов совместно осуществляют подбор наиболее подходящих задач и подзадач с учетом имеющейся информации. Описаны принципы построения и метод работы адаптивного мультиагентного диспетчера ресурсов ОВС, алгоритмы работы агентов ресурсов и задач, проведены исследования эффективности разработанных алгоритмов с использованием распределенной программной модели.

*Ключевые слова*: распределенная вычислительная среда, облачная вычислительная среда, теория мультиагентных систем, диспетчирование вычислений.

**DOI:** 10.31857/S0005231022080062, **EDN:** AHCWSW

### 1. Введение

Наблюдаемое в последние годы бурное развитие технологий передачи данных привело к появлению новых подходов к организации вычислений, основанных на использовании множества распределенных в пространстве вычислительных ресурсов с применением сервис-ориентированного подхода, обес-

[1] Работа выполнена в рамках Государственного задания Санкт-Петербургского политехнического университета Петра Великого (проект № 075-01429-22-02).

печивающего «вычисления по требованию». Эти возможности породили новый класс распределенных вычислительных сред, базирующихся на парадигме облачных вычислений, предполагающей организацию распределенных вычислений на основе пула виртуализованных вычислительных ресурсов, предоставляемых по запросу внешним пользователям удаленно через сеть Интернет [1]. В настоящее время облачные вычислительные среды (ОВС) находят широкое применение при решении сложных вычислительных задач в различных предметных областях: физика; химия и биология; фармакология и фармацевтика; материаловедение; нефтегазодобыча и т.п. [2, 3]. При этом пользовательские задачи в общем виде являются достаточно трудоемкими, иначе пользователь мог бы решить их локально.

Одним из главных преимуществ ОВС является абстрагирование пользователя от выбора конкретных аппаратных ресурсов: он просто предоставляет свои задачи в облако и ожидает их скорейшего решения, при этом, когда говорим об облачных системах, речь обычно идет о минутах или часах. В общем виде ОВС состоит из множества различных вычислителей, это обусловлено растянутым во времени расширением парка вычислительной техники (когда нет возможности или становится невыгодно приобретать узлы, идентичные уже установленным), более низкой востребованностью определенных ускорителей (GPU, FPGA) и т.д. Такая гетерогенная структура, с одной стороны, позволяет повысить эффективность работы ОВС за счет использования тех вычислительных ресурсов, которые обеспечивают максимальную реальную производительность при решении определенных задач, но, с другой стороны, делает существенно более значимым распределение решаемых задач по разнотипным вычислительным ресурсам, входящим в состав гетерогенной ОВС [4].

Указанная проблема многократно усложняется следующими факторами:

• ОВС должна обеспечивать решение некоторого множества различных пользовательских задач, поступающих в произвольные моменты времени, при этом поступающие задачи в общем виде могут обладать сложной внутренней структурой (состоять из информационно взаимосвязанных подзадач, эффективность решения каждой из которых может в сильной степени зависеть от типа используемого вычислительного ресурса);

• узлы ОВС могут со временем изменять свои характеристики в зависимости от множества факторов, например с связи с изменением температуры процессоров вследствие ухудшения вентиляции или сезонных колебаний, деградацией или отказами оборудования и т.п.

Таким образом, можно сделать вывод о высокой важности учета актуальных параметров элементов ОВС при распределении задач для повышения эффективности их выполнения и об актуальности проблемы эффективного распределения задач в зависимости от параметров вычислителя даже для ОВС, состоящих из идентичных вычислителей.

Современные исследования в области распределения задач в гетерогенных средах с изменяющимися со временем параметрами в основном базируются на предсказании. Например, в [5] авторы описывают мультиагентный алгоритм Deep-Q-network (DQN), базирующийся на централизованном многоагентном обучении с подкреплением (Multiagent Reinforcement Learning — MARL), который предполагает обучение с подкреплением для оптимизации времени и стоимости распределения и выполнения задач, при этом агенты наследуют эвристики для применения в сформированных нейронных сетях. Авторы [6] представляют генетический алгоритм с эффективной настройкой (Genetic Algorithm with Efficient Tune-In — GAETI) для выполнения длительных вычислений на удаленных вычислительных системах с улучшением времени решения. В [7] авторы предлагают метод гибридной оптимизации роя частиц, предполагающий распределение на базе недоминирующей сортировки. Другим направлением публикаций по тематике распределения задач является применение теоретико-игровых моделей и других методик обучения с подкреплением [8–14], где совместно применяются концепция равновесия в теории игр и методы мультиагентного обучения для оптимизации с множеством ограничений и множеством целей. Общим недостатком рассмотренных подходов является необходимость наличия накопленной статистической информации, которая теряет актуальность при изменении состава распределенной системы.

В некоторых из рассмотренных подходов применяются мультиагентные методы поиска оптимальных распределений, показывающие высокую эффективность, однако указанный инструментарий применяется централизованно: виртуальные агенты узлов системы в рамках централизованного диспетчера составляют виртуальное расписание. В настоящей статье предлагается подход, позволяющий существенно расширить функциональность агентов при диспетчировании путем отказа от виртуальных агентов и превращения элементов ОВС в множество физически распределенных агентов, представляющих собой вычислительные узлы, управляемые программными агентами.

В [15, 16] описывается подход к мультиагентной организации диспетчирования ресурсов в гетерогенных ОВС, позволяющий реализовать распределение различных ресурсов в процессе выполнения множества поступающих сложных пользовательских задач. Показано, что такая мультиагентная организация диспетчера позволяет эвристически обеспечить квазиоптимальное распределение ресурсов ОВС, адаптивно учитывающее их актуальные параметры, что обеспечивает возможность применения широкого перечня оборудования. Другими преимуществами указанного подхода являются возможность оперативного масштабирования ОВС, а также увеличенная отказоустойчивость, достигаемая за счет децентрализации процесса диспетчирования.

Однако применение подхода, предложенного в [15, 16], предполагает, что пользователи должны указывать время, к которому необходимо решить зада-

чу. При этом, поскольку пользователи не обладают информацией о загрузке и параметрах ОВС, указание ими неадекватного времени решения приводит к невозможности выполнения задач, что, в свою очередь, приводит к снижению эффективности работы ОВС в целом [17]. Настоящая статья посвящена разработке новых метода и алгоритмов мультиагентного диспетчирования ОВС, позволяющих распределять задачи, требуемое время выполнения которых не задано, при этом целью работы диспетчера ОВС становится минимизация времени решения всех поступивших задач с помощью имеющихся вычислительных ресурсов.

## 2. Формальная постановка задачи

Как было сказано выше, множество пользователей в случайные моменты времени отправляют в ОВС множество различных задач $Z =< Z_1, \ldots, Z_M >$. Под задачей будем понимать некоторое множество информационно зависимых (взаимосвязанных) подзадач, каждая из которых имеет значительную вычислительную трудоемкость. Формально каждая такая задача $Z_l \in Z$ может быть представлена в виде ациклического графа $G_l(Q_l, X_l)$ (рис. 1), вершины $q_j \in Q_l$ которого соответствуют некоторым подзадачам $O_j$, принадлежащим множеству подзадач $O =< O_1, \ldots, O_c >$, а дуги $x(q_j, q_{j+1}) \in X_l$ определяют информационные взаимосвязи между подзадачами (т.е. если две вершины $q_j$ и $q_{j+1}$ соединены дугой $x(q_j, q_{j+1})$, то это означает, что данные, полученные в результате решения подзадачи $O_j$, приписанной вершине $q_j$, являются исходными данными для подзадачи $O_{j+1}$, приписанной вершине $q_{j+1}$). Будем считать, что каждой вершине $q_j \in Q_l$ приписаны тип решаемой подзадачи $O_j \in O$ и ее вычислительная трудоемкость $v_j$, оцениваемая числом эле-

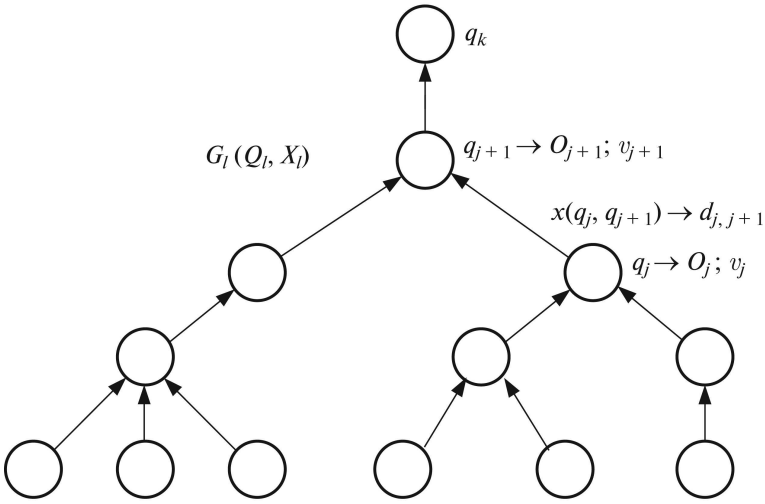


Рис. 1. Граф $G_l(Q_l, X_l)$ задачи $Z_l$.

ментарных вычислительных операций, выполняемых при ее решении; а дуге $x(q_j, q_{j+1}) \in X_l$ приписан объем данных $d_{j,j+1}$, передаваемых от подзадачи $O_j$, приписанной вершине $q_j$, подзадаче $O_{j+1}$, приписанной вершине $q_{j+1}$ (рис. 1).

В состав ОВС входит множество гетерогенных вычислительных ресурсов $R =< R_1, \dots R_N >$. Будем считать, что каждый ресурс $R_i \in R$ может решать некоторый набор (подмножество) типов подзадач $O_i =< O_1, \dots O_L > \subseteq O$ $(i = 1, \dots, N)$, причем реальная производительность ресурса $R_i$ при решении подзадачи $O_j \in O_i$ $(j = 1, \dots, L)$ составляет $S_i(O_j)$ (операций в секунду). В общем случае реальная производительность различных ресурсов ОВС при решении идентичных подзадач $O_j$ может быть различной, т.е. $S_p(O_j) \neq S_c(O_j)$ $(p = 1, \dots, N;\ c = 1, \dots, p-1, p+1, \dots, N)$. Будем считать, что известна пропускная способность $Y_p$ (байт/с) канала связи ресурса $R_i$ $(i = 1, \dots, N)$ с облачной инфраструктурой.

Цель работы диспетчера заключается в минимизации суммарного времени решения задач множества $Z$ $T_Z = \sum_{i=1}^{M} T(Z_i)$, где $T(Z_i)$ — время решения задачи $Z_i$ путем выбора определенного распределения их подзадач по вычислительным ресурсам $R =< R_1, \dots, R_N >$ с учетом оперативной оценки их текущей производительности при решении той или иной подзадачи и пропускной способности канала связи с облачной инфраструктурой.

Решение сформулированной таким образом задачи распределения ресурсов ОВС по поступающим задачам множества $Z$ классическими методами, например методами динамического и линейного программирования [18–20], является сложной задачей из-за того, что вычислительные ресурсы гетерогенны, их число велико, задачи отличаются друг от друга, что приводит к экспоненциальному росту пространства перебора при распределении подзадач различных задач множества $Z$ между ресурсами $R =< R_1, \dots R_N >$ с учетом их функциональных возможностей и производительности на той или иной подзадаче. Поэтому в данном случае предлагается применить мультиагентный подход, так как в этом случае каждый из узлов будет обладать актуальной информацией о своем узле и выполнять часть задачи диспетчирования [21].

## 3. Метод адаптивного мультиагентного диспетчирования ресурсов

Прежде чем приступить к описанию предложенного метода мультиагентного диспетчирования ОВС, расширим понятие «нити», введенное в [15, 16]. Как и раньше, под нитью, будем понимать некоторую последовательность вершин $H_f =< q_1^f, \dots, q_k^f >$ графа $G_l(Q_l, X_l)$ задачи $Z_l \in Z$, в которой вершины $q_j^f$ и $q_{j+1}^f$ $(j = 1, \dots, k-1)$ соединены дугой $x(q_j^f, q_{j+1}^f)$ (рис. 2).

Иными словами, нить определяет некоторую последовательность подзадач задачи $Z_l$, в которой каждая последующая подзадача использует в качестве

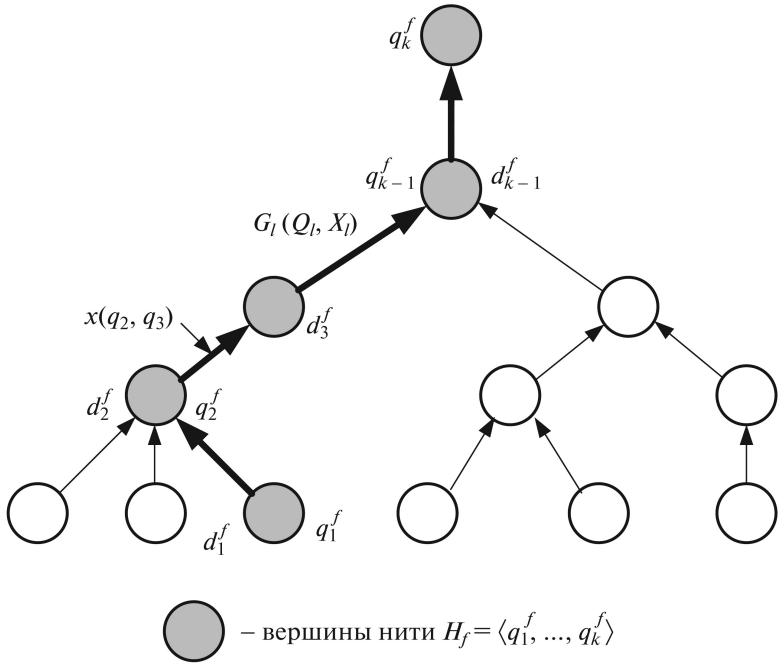


Рис. 2. Нить $H_f$ в графе $G_l(Q_l, X_l)$ задачи $Z_l$.

исходных данных результат выполнения предыдущей подзадачи. При этом подзадачи, приписанные вершинам нити $H_f$, могут выполняться только последовательно. Под длиной $T_f$ нити $H_f$ будем понимать суммарное время, затрачиваемое на ее решение, определяемое как

$$T_f = \sum_{i=1}^{k} (T(O_i) + T_{i,i+1}),$$

где $T(O_i)$ — время решения подзадачи $O_i$, приписанной вершине $q_i^f \in H_f$ $(i = 1, \ldots, k)$; $T_{i,i+1}$ — время передачи данных, полученных в результате решения подзадачи $O_i$, ресурсу, решающему следующую подзадачу $O_{i+1}$, приписанную вершине $q_{i+1}^f \in H_f$, причем

$$T(O_i) = \frac{v_i}{S_p(O_i)},$$

где $v_i$ — трудоемкость подзадачи $O_i$; $S_p(O_i)$ — реальная производительность ресурса $R_p$, решающего подзадачу $O_i$;

$$T_{i,i+1} = \begin{cases} 0, & \text{если подзадачи } O_i \text{ и } O_{i+1} \text{ решаются} \\ & \text{одним и тем же ресурсом } R_p \in \mathbf{R}, \\ \dfrac{d_{r,r+1}^f}{Y_p}, & \text{если подзадачи } O_i \text{ и } O_{i+1} \text{ решаются} \\ & \text{различными ресурсами } R_p \text{ и } R_c, \end{cases}$$

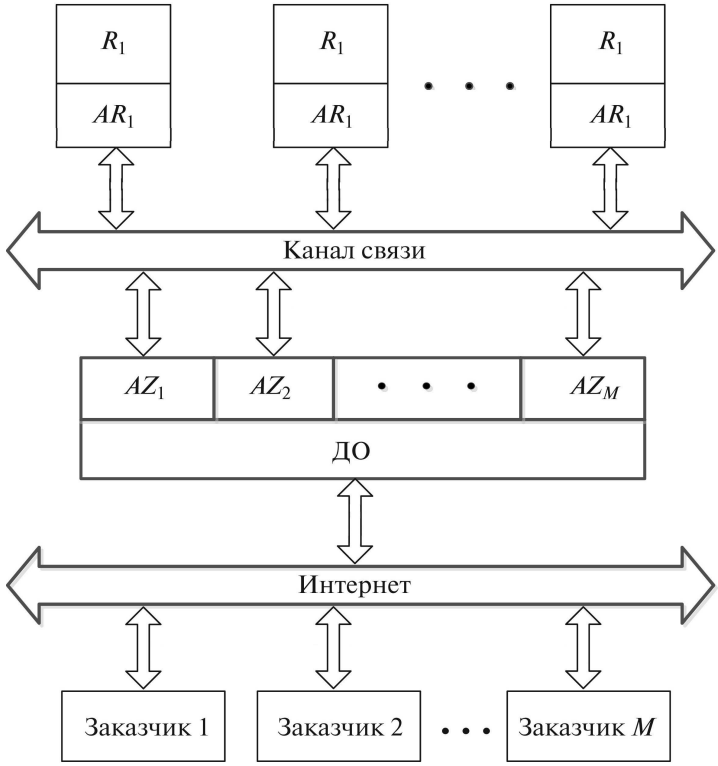


Рис. 3. Структура ОВС.

где $d^f_{i,i+1}$ — объем данных, передаваемых от подзадачи $O_i$ подзадаче $O_{i+1}$; $Y_p$ — пропускная способность канала связи ресурса $R_p$.

Если вся нить $H_f$ выполняется одним и тем же ресурсом $R_p$, то ее длина будет составлять

$$T_f = \sum_{i=1}^{k} \frac{v_i}{S_p(O_i)} + \frac{d^f_{k,k+1}}{Y_p}, \tag{1}$$

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

При этом кратчайшее время выполнения задачи $Z_l \in Z$ будет, в первую очередь, зависеть от длины критического пути в графе $G_l(Q_l, X_l)$, т.е. нити, суммарная трудоемкость вершин которой максимальна [22].

В [15, 16] предложен мультиагентный подход к диспетчированию, предполагающий, что каждый ресурс $R_j$ $(i = 1, \ldots, N)$ обладает программным агентом $AR_i$, осуществляющим поиск работы для «своего» ресурса $R_j$. Для этого агент $AR_i$ периодически опрашивает выделенный ресурс — «доску объявлений» (ДО), на которой пользователи размещают свои задачи. В случае

обнаружения на ДО некоторой задачи $Z_l \in Z$ агент $AR_j$ делает попытку войти в состав сообщества $R_l \subseteq R$ по его решению, принимая на себя исполнение наиболее длинной нити, что он может реализовать с помощью «своего» ресурса $R_j$ к установленному пользователем моменту времени. Предложенный подход не может быть применен в случае, когда время выполнения задач не задано, поэтому, кроме агентов $AR_i$ $(i = 1, \ldots, N)$, представляющих различные ресурсы $R_j \in R$ $(j = 1, \ldots, N)$, предлагается ввести в состав мультиагентного диспетчера дополнительно к ним агентов задач $AZ_j$ $(j = 1, \ldots, M)$, по сути являющихся «представителями пользователей» и ответственных за сокращение времени выполнения своей задачи (рис. 3).

Для решения пользовательской задачи в ОВС выполняются следующие шаги:

1. Пользователь формирует свою задачу $Z_l \in Z$ в виде графа $G_l(Q_l, X_l)$ и размещает на ДО.

2. Каждой задаче $Z_l \in Z$, поступившей на ДО, назначается агент $AZ_l$.

3. Агент $AZ_l$ задачи $Z_l$ с помощью алгоритмов поиска критического пути [23] выделяет в графе $G_l(Q_l, X_l)$ наиболее трудоемкую нить $H_1 = =< q_1^1, \ldots, q_k^1 >$, для которой значение $V_1 = \left( \sum_{i=1}^{k} v_i^1 \right)$ максимально, где $v_i^1$ $(j = 1, \ldots, k)$ — трудоемкость подзадачи $O_i$, приписанной вершине $q_i^1 \in H_1$, и устанавливает возможный (желательный) момент времени начала ее исполнения $t_{\text{н}}^1 = t_{\text{тек}}$, где $t_{\text{тек}}$ — текущий момент времени. После этого нить $H_1$ выставляется на ДО для исполнения.

4. Агенты $AR_j$ различных ресурсов $R_j$ $(j = 1, \ldots, N)$ последовательно опрашивают ДО в поисках работы.

5. При обнаружении нити $H_1$, выставленной агентом $AZ_l$, агент $AR_j$ оценивает эффективность участия в ее решении.

6. Для этого агент $AR_j$ выбирает в нити $H_1 =< q_1^1, \ldots, q_k^1 >$ поднить $H_{1j}^1 = =< q_1^1, \ldots, q_b^1 > \subseteq H_1$ $(b \leq k)$, вершинам которой приписаны подзадачи множества $O_i$, т.е. подзадачи, выполняемые ресурсом $R_j$ (рис. 4).

Агент $AR_j$ имеет на базе информацию о том, когда закончит выполнять запланированные задачи (момент времени $t_{\text{н}j}^1$), определяет момент, когда он сможет начать работу над поднитью $H_{1j}^1$, и время $t_{\text{к}j}^1$ окончания исполнения поднити $H_{1j}^1$, причем

$$t_{\text{к}j}^1 = t_{\text{н}j}^1 + \sum_{i=1}^{b} \frac{v_i^1}{S_j(O_i)} + \frac{d_{b,b+1}^1}{Y_j}, \tag{2}$$

где $v_i^1$ — трудоемкость подзадачи $O_i$, приписанной вершине $q_i^1 \in H_{1j}^1$ $(i = = 1, \ldots, b)$; $S_j(O_i)$ — производительность ресурса $R_j$ при решении подзадачи $O_i$; $d_{b,b+1}^1$ — объем данных, приписанных конечной дуге $x(q_b^1, q_{b+1}^1)$ поднити $H_{1j}^1$; $Y_j$ — пропускная способность канала связи ресурса $R_j$.

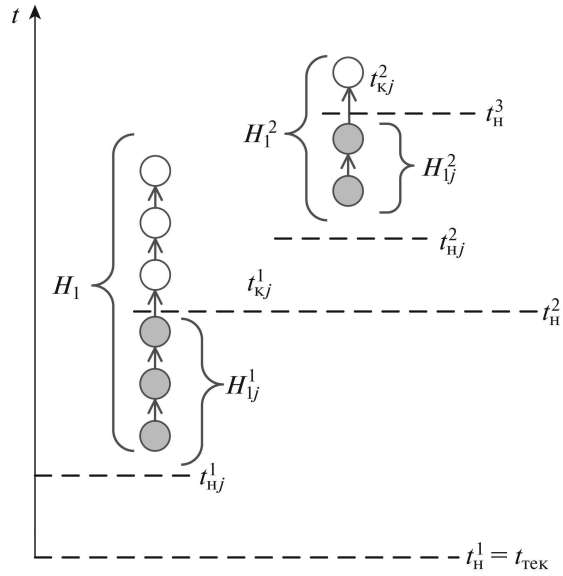


Рис. 4. Распределение операции нити $H_1$ между агентами $AR_j$ $(j = 1, \ldots, N)$.

Выбранная поднить $H^1_{1j}$, с приписанными ей моментами времени начала исполнения $t^1_{нj}$ и окончания исполнения $t^1_{кj}$, передается агентом $AR_p$ агенту $AZ_l$ задачи $Z_l$ в качестве «оферты» по его участию в исполнении нити $H_1$.

7. По прошествии некоторого таймаута агент $AZ_l$ задачи $Z_l$ среди всех поступивших «предложений» выбирает наиболее подходящее — поднить $H^1_{1p}$, «предложенную» агентом $AR_p$, для которой значение величины

$$E_p = \frac{\sum\limits_{i=1}^{b} v^1_i}{t^1_{кp} - t^1_н}$$

максимально, где $v^1_i$ — трудоемкость подзадачи $O_i$ $(i = 1, \ldots, b)$, приписанной вершине $q^1_i \in H^1_{1p}$; $t^1_н$ — желательное время начала исполнения нити $H_1$, установленное агентом задачи $AZ_l$; $t^1_{кp}$ — момент времени завершения исполнения поднити $H^1_{1p}$ агентом $AR_p$.

8. Поднить $H^1_{1p}$, для которой значение $E_p$ максимально, закрепляется за агентом $AR_p$, о чем ему направляется соответствующее сообщение от агента $AZ_l$, и агент $AR_p$ включается в состав сообщества $R_l$ по выполнению задачи $Z_l$. В графе $G_l(Q_l, X_l)$ задачи $Z_l$ каждой вершине $q^1_j$ $(j = 1, \ldots, b)$ поднити $H^1_{1p}$ приписывается номер $p$ ресурса $R_p$, за которым закреплено ее исполнение, а также момент времени ее исполнения $t^1_j$, определяемый как

$$t^1_j = t^1_{нp} + \sum_{i=1}^{j} \frac{v^1_i}{S_p(O_i)}, \tag{3}$$

где $v_i^1$ — трудоемкость подзадачи $O_i$, приписанной вершине $q_i^1$; $S_p(O_i)$ — производительность ресурса $R_p$ при решении подзадачи $O_i$.

9. Агент $AZ_l$ задачи $Z_l$ исключает вершины $q_1^1, \ldots, q_b^1$ поднити $H_{1p}^1$ из нити $H_1$, в результате чего формируется новая (укороченная) нить $H_1^2 =< q_{b+1}^1, \ldots, q_k^1 >$ (рис. 4), а первой вершине $q_{b+1}^1$ этой нити приписывается номер агента $AR_p$, от которого должны поступить исходные данные для ее выполнения, а также возможное (желательное) время начала ее исполнения $t_{\text{н}}^2 = t_{\text{кр}}^1$, где $t_{\text{кр}}^1$ — время окончания исполнения поднити $H_{1p}^1$, определяемое согласно выражению (2).

После этого нить $H_1^2$ выставляется агентом $AZ_l$ на ДО для исполнения.

10. В дальнейшем агенты различных ресурсов $R_j$ $(i = 1, \ldots, N)$ оценивают свои возможности по участию в исполнении нити $H_1^2$. При этом агент $AR_j$ $(j = 1, \ldots, N)$ выделяет поднить $H_{1j}^2 =< q_{b+1}^1, \ldots, q_m^1 > \subseteq H_1^2$ $(m \le k)$ (см. рис. 4), вершинам которой приписаны подзадачи подмножества $O_j \subseteq O$, выполняемого ресурсом $R_j$, а также определяет момент времени начала ее исполнения $t_{\text{н}j}^2$, (т.е. момент, когда он может приступить к ее исполнению, причем $t_{\text{н}j}^2 \ge t_{\text{н}}^2$), а также момент времени окончания ее исполнения $t_{\text{к}j}^2$, определяемый как

$$t_{\text{к}j}^2 = t_{\text{н}j}^2 + \sum_{i=b+1}^{m} \frac{v_i^1}{S_j(O_i)} + \frac{d_{m,m+1}^1}{Y_j}. \tag{4}$$

Выделенная таким образом поднить $H_{1j}^2$ направляется агентом $AR_j$ агенту $AZ_l$ задачи $Z_l$ в качестве «предложения» по участию в выполнении нити $H_1^2$.

11. По прошествии определенного таймаута агент $AZ_l$ задачи $Z_l$ выбирает наилучшее «предложение» того агента $AR_c$, предлагающего к исполнению поднить $H_{1c}^2 =< q_{b+1}^1, \ldots, q_m^1 >$ $(m \le k)$, для которой величина

$$E_c = \frac{\sum\limits_{b+1}^{m} v_i^1}{t_{\text{кс}}^2 - t_{\text{н}}^2} \tag{5}$$

максимальна.

Вершины поднити $H_{1c}^2$ закрепляются за агентом $AR_c$, о чем ему направляется соответствующее сообщение, и агент $AR_p$ включается в сообщество $R_l$ по решению задачи $Z_l$. Кроме того, в каждой вершине $q_j^1$ $(i = b+1, \ldots, m)$ графа $G_l(Q_l, X_l)$ приписывается номер $c$ ресурса $R_c$, за которым закреплено ее исполнение, и момент времени $t_{jc}^1$ ее исполнения, определяемый как

$$t_j^1 = t_{\text{нс}}^2 + \sum_{i=b+1}^{j} \frac{v_i^1}{S_c(O_i)}, \tag{6}$$

где $v_i^1$ — трудоемкость подзадачи $O_i$, приписанной вершине $q_i^1$; $S_c(O_i)$ — производительность ресурса $R_c$ при решении подзадачи $O_i$.

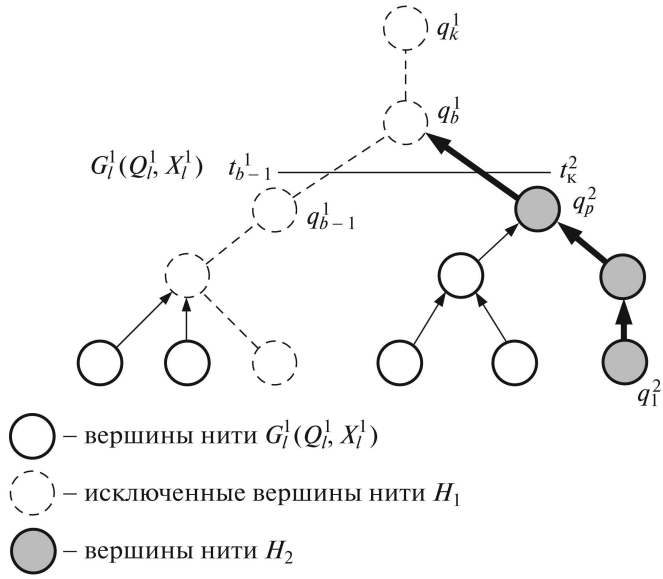


Рис. 5. Граф $G_l^1(Q_l^1, X_l^1)$ задачи $Z_l$, модифицированный агентом $AZ_l$.

12. После этого агент задачи $AZ_l$ формирует новую нить $H_1^3 = =< q_{m+1}^1, \ldots, q_k^1 >$, путем исключения из нити $H_1^2$ вершин поднити $H_{1c}^2$, т.е. $H_1^3 = H_1^2/H_{1c}^2$, и первой вершине $q_{m+1}^1$ этой нити приписывает возможное (желательное) время начала ее исполнения $t_н^3 = t_{кс}^2$, где $t_{кс}^2$ — момент времени завершения поднити $H_{1c}^2$, определяемый согласно (4).

Далее нить $H_1^3$ вновь выставляется агентом задачи $AZ_l$ на исполнение и т.д. до тех пор, пока не окажется, что очередная нить пуста $H_1^k = \varnothing$. Это говорит о том, что все вершины нити $H_1$ закреплены за агентами различных ресурсов $R_j$ $(i = 1, \ldots, N)$.

13. После этого агент $AZ_l$ задачи $Z_l$ исключает нить $H_1$ из графа задачи $G_l(Q_l, X_l)$, в результате чего формируется новый граф $G_l^1(Q_l^1, X_l^1) = = G_l(Q_l, X_l)/H_1$ (рис. 5).

В обновленном графе $G_l^1(Q_l^1, X_l^1)$ агентом $AZ_l$ определяется наиболее трудоемкая нить $H_2 =< q_1^2, \ldots, q_r^2 >$ (рис. 5) для дальнейшего распределения между агентами ресурсов $R_j$ $(i = 1, \ldots, N)$, в результате чего всем вершинам нити $H_2 =< q_1^2, \ldots, q_r^2 >$ в графе $G_l(Q_l, X_l)$ будут поставлены в соответствие номерам ресурсов $R_j$, за которыми закреплено их исполнение, а также моменты времени $t_f^2$ $(f = 1, \ldots, r)$ их исполнения.

14. Поскольку нить $H_2 =< q_1^2, \ldots, q_r^2 >$ является ветвью нити $H_1 = =< q_1^1, \ldots, q_k^1 >$ (т.е. конечная вершина $q_r^2$ нити $H_2$ инцидентна одной из вершин $q_b^1$ нити $H_1$) (см. рис. 5), то требуется оценить погрешность времени завершения нити $H_2$ для определения актуальности времени начала работы над инцидентной ей вершиной $q_b^1$ нити $H_1$. Поэтому агент $AZ_l$ задачи $Z_l$ сравни-

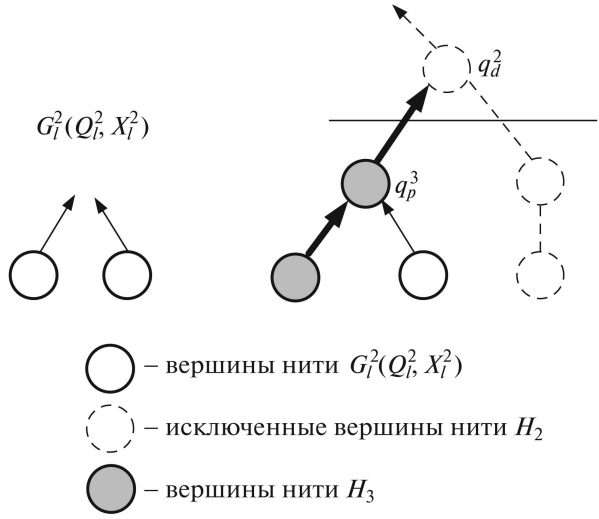


Рис. 6. Граф $G_l^2(Q_l^2, X_l^2)$, модифицированный агентом $AZ_l$.

вает момент времени $t_{\text{к}}^2$ завершения исполнения нити $H_2$, определяемый как

$$t_{\text{к}}^2 = t_r^2 + \frac{d_{r,r+1}^2}{Y_j}, \tag{7}$$

где $t_r^2$ — момент времени, приписанный конечной вершине $q_r^2$ нити $H_2$; $d_{r,r+1}^2$ — объем передаваемых данных, приписанный дуге, исходящей из вершины $q_r^2$; $Y_j$ — пропускная способность канала связи ресурса $R_j$, приписанного вершине $q_r^2$;

с моментом времени $t_{b-1}^1$, приписанным вершине $q_{b-1}^1$ нити $H_1$. Если оказывается, что $t_{\text{к}}^2 > t_{b-1}^1$, то это означает, что данные, получаемые в результате выполнения подзадач нити $H_2$ и необходимые для решения подзадачи, приписанной вершине $q_b^1 \in H_1$, поступят позже, чем данные, также необходимые для решения подзадачи вершины $q_b^1$ и получаемые в результате выполнения подзадачи вершины $q_{b-1}^1$ нити $H_1$. В этом случае агент задачи $AZ_l$ производит корректировку графика исполнения нити $H_1$ путем смещения моментов времени, приписанных ее вершинам $q_b, \ldots q_k$, на величину $\Delta t = t_{\text{к}}^2 - t_{b-1}^1$.

15. Агент $AZ_l$ задачи $Z_l$ формирует обновленный граф задачи

$$G_l^2(Q_l^2, X_l^2) = G_l^1(Q_l^1, X_l^1)/H_2$$

путем исключения из рассмотрения вершин нити $H_2$ (пометив их как решаемые), в этом графе (рис. 6) определяет наиболее трудоемкую нить $H_3$ и отправляет на ДО.

В процессе работы ОВС распределение вершин (подзадач) графа задачи $G_l(Q_l, X_l)$ продолжается далее до тех пор, пока они все не будут распределены, т.е. очередной граф будет пуст $G_l^d(Q_l^d, X_l^d) = \varnothing$. В результате всем вершинам $q_i \in Q_l$ графа задачи $G_l(Q_l, X_l)$ будут приписаны номера ресурсов $R_j$

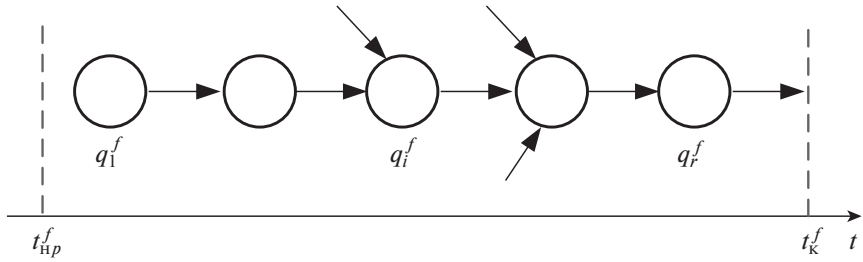


Рис. 7. Исполнение поднити $H_f^m$ агентом $AR_p$.

$(j = 1, \ldots, N)$, отвечающих за их исполнение, а также планируемые моменты времени $t_i$ их исполнения, т.е. будет построен график решения задачи $Z_l$.

16. После этого агент задачи $Z_l$ сообщает пользователю планируемый момент времени $t_l^k$ выполнения его задачи $Z_l$, который определяется как

$$t_l^k = t_k + \frac{d_{k,k+1}}{Y_p}, \tag{8}$$

где $t_k$ — момент времени, приписанный конечной вершине $q_k$ графа $G_l(Q_l, X_l)$; $d_{k,k+1}$ — объем результирующих данных, приписанный дуге, исходящей из вершины $q_k$; $Y_p$ — пропускная способность канала связи ресурса $R_p$, закрепленного за вершиной $q_k$.

В случае согласия пользователя агент $AZ_l$ сообщает всем агентам ресурсов $R_j$ $(i = 1, \ldots, N)$, задействованным в выполнении данной задачи (т.е. входящим в сообщество $R_l$), о необходимости выполнения принятых на себя подзадач согласно установленному плану (временному графику).

17. Когда агент $AR_p$ ресурса $R_p$ получает подтверждение от агента $AZ_l$ задачи $Z_l$ о необходимости выполнения закрепленной за ним поднити $H_f^m = =< q_1^f, \ldots, q_r^f >$ графа задачи $G_l(Q_l, X_l)$, он включает подзадачи, приписанные вершинам поднити $H_f^m$, в свой график работы.

18. При наступлении момента времени $t_{нр}^f$ начала исполнения поднити $H_f^m$ агент $AR_p$ приступает к выполнению последовательности подзадач $O_i^f$ $(i = 1, \ldots, r)$, приписанных вершинам $q_i^f \in H_f^m$ (рис. 7). Если для выполнения очередной подзадачи $O_i^f$ необходимы исходные данные, поступающие от других ресурсов, то агент $AR_j$ первым делом проверяет их наличие. Если требуемые исходные данные еще не поступили, агент $AR_j$ переходит в режим ожидания.

19. По факту получения необходимых данных агент $AR_p$ осуществляет решение подзадачи $O_i^f$ $(i = 1, \ldots, r)$ с помощью «своего» ресурса $R_p$. После выполнения всех подзадач $O_i^f$, приписанных вершинам $q_i^f$ $(i = 1, \ldots, r)$ поднити $H_f^m$, агент $AR_p$ отправляет агенту $AZ_l$ задачи $Z_l$ сообщение о работе над последовательностью подзадач поднити $H_f^m$ и передает результат ее следующему ресурсу $R_p$, исполняющему смежную с ней вершину графа $G_l(Q_l, X_l)$.

20. Агент $AZ_l$ осуществляет проверку соблюдения временного графика выполнения агентом $AR_p$ подзадач нити $H_f^m$ путем сравнения запланированного времени $t_{\text{к}}^f$ завершения поднити $H_f^m$, определяемого как $t_{\text{к}}^f = t_r^f + \frac{d_{r,r+1}^f}{Y_p}$ (рис. 7) (где $t_r^f$ — момент времени, приписанный конечной вершине $q_r^f$ поднити $H_f^m$; $d_{r,r+1}^f$ — объем передаваемых данных, приписанных исходящей из вершины $q_r^f$ дуге; $Y_p$ — пропускная способность канала связи ресурса $R_p$), с текущим временем $t_{\text{тек}}$.

Если $t_{\text{тек}} > t_{\text{к}}$, то это означает, что возникла задержка в выполнении временного графика решения задачи $Z_l$, о чем агент $AZ_l$ сообщает пользователю. Кроме того, агент $AZ_l$ актуализирует временной график исполнения последующих вершин $q_r, q_{r+1}, \ldots, q_k$ графа задачи $G_l(Q_l, X_l)$ путем смещения моментов времени на величину $\Delta t = t_{\text{тек}} - t_{\text{к}}^f$ (см. рис. 5).

21. После того, как все подзадачи задачи $Z_l$ решены (т.е. агент задачи получил подтверждения от всех агентов сообщества $R_l$ об успешном завершении всех принятых на себя поднитей), агент $AZ_l$ сообщает пользователю об успешном решении его задачи $Z_l$, после чего задача $Z_l$ снимается с ДО.

Для применения на практике описанных ранее принципов мультиагентного диспетчирования гетерогенной ОВС были разработаны алгоритмы работы агентов задачи и вычислительного ресурса.

## 4. Алгоритм работы агента задачи при мультиагентном диспетчировании ресурсов

Как только граф $G_l(Q_l, X_l)$ задачи $Z_l$ передается на ДО, ему назначается агент $AZ_l$, который находит в графе задачи $G_l(Q_l, X_l)$ самую трудоемкую нить $H_1 =< q_1^1, \ldots, q_k^1 >$ и устанавливает желательный (возможный) момент времени начала ее исполнения $t_{\text{н}}^1 = t_{\text{тек}}$, после чего выставляет нить $H_1$ на исполнение на ДО (см. рис. 4).

После того, как агенты ресурсов $R_j$ $(i = 1, \ldots, N)$ проанализируют возможность своего участия в выполнении нити $H_1$ и направят свои «предложения» агенту $AZ_l$ задачи $Z_l$, последний выбирает «предложение» того агента $AR_p$, который может обеспечить выполнение поднити $H_{1p}^1 =< q_1^1, \ldots, q_b^1 > \subseteq H_1$, имеющий наибольшее значение

$$E_p = \frac{\sum\limits_{i=1}^{b} v_i^1}{t_{\text{кр}}^1 - t_{\text{н}}^1},$$

где $v_i^1$ $(i = 1, \ldots, b)$ — трудоемкость подзадачи $O_i$, приписанной вершинам $q_i^1 \in H_{1p}^1$; $t_{\text{кр}}^1$ — момент времени завершения исполнения поднити $H_{1p}^1$ агентом $AR_p$, определяемый с помощью выражения (2).

Агент $AR_p$ включается в сообщество $R_l$ по решению задачи $Z_l$, а в графе $G_l(Q_l, X_l)$ всем вершинам поднити $H_{1p}^1$ приписывается номер агента $AR_p$,

отвечающего за их исполнение, а также моменты времени их исполнения, определяемые с помощью выражения (3).

Агент $AZ_l$ исключает поднить $H_1^1$ из нити $H_1$, в результате чего формируется новая нить $H_1^2 \subseteq H_1$, которой агент $AZ_l$ приписывает возможный момент времени начала ее исполнения $t_{\text{н}}^2$, определяемый временем исполнения поднити $H_{1p}^1$, т.е. $t_{\text{н}}^2 = t_{\text{кр}}^1$ (см. рис. 4). После этого нить $H_1^2$ выставляется агентом $AZ_l$ на ДО для исполнения.

После этого агент $AZ_l$ ожидает предложения агентов $AR_j$ $(j = 1, \ldots, N)$ по работе над нитью $H_1^2$ и выбирает «предложение» того агента $AR_c$, который обеспечивает выполнение такой поднити $H_{1c}^2 =< q_{b+1}^1, \ldots, q_m^1 >$ $(m \le k)$, для которой значение $E_c$ (см. выражение (4)) максимально. Выбранный агент $AR_c$ включается в сообщество $R_l$, а в графе $G_l(Q_l, X_l)$ вершинам этой поднити $H_{1c}^2$ приписываются его номер, время их исполнения $t_1^j$, определяемое с помощью выражения (5).

Описанный процесс продолжается до тех пор, пока не будут распределены все вершины (подзадачи) нити $H_1$, т.е. пока не окажется, что очередная поднить пуста $H_1^m = \varnothing$.

Далее агент задачи $AZ_l$ исключает из графа $G_l(Q_l, X_l)$ нить $H_1$, в результате чего формируется новый граф $G_l^1(Q_l^1, X_l^1) = G_l(Q_l, X_l)/H_1$, и выделяет в последнем наиболее трудоемкую нить $H_2$, которая выставляется на ДО на исполнение (см. рис. 5). После того как все вершины (подзадачи) нити $H_2 =< q_1^2, \ldots, q_r^2 >$ будут разобраны агентами $AR_j$ $(j = 1, \ldots, N)$, в графе $G_l(Q_l, X_l)$ им будут приписаны номера соответствующих агентов $AR_p$, а также время их исполнения $t_i^2$ $(i = 1, \ldots, r)$. После этого агент задачи $AZ_l$ должен проверить согласованность времени поступления исходных данных для подзадачи $O_b$, приписанной вершине $q_b^1$ нити $H_1$, инцидентной конечной вершине нити $H_2$. Для этого агент задачи $AZ_l$ сравнивает момент времени $t_{\text{к}}^2$ завершения исполнения нити $H_2$, определяемый с помощью выражения (7), с моментом времени $t_{b-1}^1$ исполнения предыдущей вершины $q_{b-1}^1$ нити $H_1$ (см. рис. 5). Если $t_r^2 > t_{b-1}^1$, то это означает, что данные, получаемые в результате исполнения нити $H_2$ и необходимые для выполнения подзадачи $O_b$, поступят позже, чем данные, получаемые в результате выполнения операции $O_{b-1}$, приписанной вершине $q_{b-1}$ нити $H_1$. В этом случае агент задачи $AZ_l$ должен скорректировать моменты времени исполнения всех последующих вершин $q_b^1, \ldots, q_k^1$ нити $H_1$ путем их увеличения на величину $\Delta t = t_{\text{к}}^2 - t_{b-1}^1$.

Далее нить $H_2$ исключается из графа задачи $G_l^1(Q_l^1, X_l^1)$, в результате чего формируется новый граф $G_l^2(Q_l^2, X_l^2) = G_l^1(Q_l^1, X_l^1)/H_2$, в котором вновь выделяется наиболее трудоемкая нить $H_3$, которая выставляется агентом $AZ_l$ на ДО для исполнения (см. рис. 6). Процесс продолжается до тех пор, пока все вершины графа задачи $G_l(Q_l, X_l)$ не будут закреплены за агентами $AR_j$ $(j = 1, \ldots, N)$ и им будут приписаны планируемые моменты времени их исполнения. По завершению данного процесса агент $AZ_l$ задачи $Z_l$ сообщает

пользователю планируемый момент времени $t_l^k$ решения его задачи $Z_l$, определяемый с помощью выражения (7).

Если предложенное время решения задачи удовлетворяет пользователя, то агент $AZ_l$ отправляет всем агентам, включенным в сообщество $R_l$, команду на выполнение операций.

Формализуем алгоритм, соответствующий описанному выше процессу:

Алгоритм 1

1. $j = 1$; $G_l^j(Q_l^j, X_l^j) = G_l(Q_l, X_l)$; $R_l = \varnothing$.

2. В графе $G_l^j(Q_l^j, X_l^j)$ выделяется наиболее трудоемкая нить $H_j = =< q_1^j, \ldots, q_k^j >$, для которой значение $V_j = \sum\limits_{i=1}^{k} v_i^j$ максимально, где $v_i^j$ — трудоемкость операции $O_i^j$, приписанной вершине $q_i^j$ $(i = 1, \ldots, k)$.

3. $m = 1$; $H_j^m = H_j$; $t_{\text{н}}^m = t_{\text{тек}}$.

4. Нить $H_j^m$ выставляется агентом $AZ_l$ на ДО для исполнения.

5. $r = 1$; $p = 0$; $E_p = \infty$.

6. Агент $AZ_l$ получает «предложение» от агента $AR_r$ об исполнении поднити $H_{jr}^m =< q_1^j, \ldots, q_b^j > \subseteq H_j^m$ $(b \le k)$, а также моменты времени $t_{\text{нг}}^m$ начала и $t_{\text{кг}}^m$ завершения ее исполнения.

7. Если $E_r = \frac{\sum\limits_{i=1}^{b} v_i^j}{t_{\text{кг}}^m - t_{\text{н}}^m} \ge E_p$, где $v_i^j$ — трудоемкость вершины $q_i^j$ $(i = 1, \ldots, b)$, то перейти к 9, иначе

8. $E_p = E_r$; $p = r$.

9. $r = r + 1$; если $r \le N$, то перейти к 6, иначе

10. Агент $AR_p$ включается в сообщество $R_l$ по выполнению задачи $Z_l$ и за ним закрепляется выполнение нити $H_{jp}^m$. В графе $G_l(Q_l, X_l)$ вершинам нити $H_{jp}^m$ приписывается номер агента $AR_p$, а также время их исполнения

$$t_f^j = t_{\text{нр}}^m + \sum_{i=1}^{f} \frac{v_i}{S_p(O_i)} \quad (f = 1, \ldots, b).$$

11. $H_j^{m+1} = H_j^m / H_{jp}^m$; если $H_j^{m+1} = \varnothing$, то перейти к 13, иначе

12. $m = m + 1$; $t_{\text{н}}^m = t_f^b + \frac{d_{b,b+1}}{Y_p}$; перейти к 4.

13. Если $j = 1$ или $t_{\text{к}}^j \le t_{f-1}^{j-1}$, где $t_{\text{к}}^j$ — момент времени завершения нити $H_j$; $t_{b-1}^{j-1}$ — момент времени исполнения вершины $q_{b-1}^{j-1}$ нити $H_{j-1} = =< q_1^{j-1}, \ldots, q_r^{j-1} >$, вершина $q_b^{j-1}$ которой инцидента конечной вершине $q_k^j$ нити $H_j$, перейти к 15, иначе

14. В графе $G_l(Q_l, X_l)$ вершинам $q_b^{j-1}, q_{b+1}^{j-1}, \ldots, q_r^{j-1}$, нити $H_{j-1}$ приписывается новое плановое время их исполнения $t_i^{j-1} = t_i^{j-1} + \Delta T$ $(i = b, b+1, \ldots, r)$, где $\Delta T = t_k^j - t_{b-1}^{j-1}$.

О корректировке времени исполнения вершин $q_b^{j-1}, \ldots, q_r^{j-1}$ нити $H_{j-1}$ сообщается агенту $AR_c$, за которым закреплено их исполнение.

15. $G_l^{j+1}(Q_l^{j+1}, X_l^{j+1}) = G_l^j(Q_l^j, X_l^j)/H_j$, если $G_l^{j+1}(Q_l^{j+1}, X_l^{j+1}) = \varnothing$, то перейти к 17, иначе

16. $j = j + 1$, перейти к 2.

17. Отправить пользователю уведомление о расчетном времени $t_l^k = t_k + \frac{d_{k,k+1}}{Y_p}$ решения, где $t_k$ — момент времени исполнения конечной вершины $q_k$ графа $G_l(Q_l, X_l)$, $d_{k,k+1}$ — объем результирующих данных, приписанных дуге, исходящей из конечной вершины $q_k$; $Y_p$ — пропускная способность канала связи ресурса $R_p$, за которым закреплено выполнение вершины $q_k$.

18. В случае согласия пользователя с расчетным временем решения его задачи $Z_l$ агент $AZ_l$ отправляет всем сообществам $R_l$ команду на исполнение их операций.

19. Если от агента $AR_p \subseteq R_l$ поступило сообщение о завершении исполнения закрепленной за ним нити $H_{jp}^m =< q_1^j, \ldots, q_b^j >$, то агент $AZ_l$ сравнивает планируемый момент времени $t_{\text{к}}^j$ завершения данной нити, определяемый как $t_{\text{к}}^j = t_b^j + \frac{d_{b,b+1}^j}{Y_p}$ (где $t_b^j$ — время исполнения, приписанное конечной вершине $q_b^1 \in H_{ip}^m$; $d_{b,b+1}^j$ — объем данных, приписанный исходящей дуге нити $H_{ip}^m$), с текущим временем $t_{\text{тек}}$. Если $t_{\text{тек}} > t_{\text{к}}^j$, то пользователю сообщается о задержке времени решения его задачи $Z_l$ на величину $\Delta t = t_{\text{тек}} - t_{\text{к}}^j$, а планируемое время исполнения всех последующих вершин графа $G_l(Q_l, X_l)$ увеличивается на величину $\Delta t$.

20. Если конечная вершина $q_b^j$ нити $H_{jp}^m$ является конечной вершиной $q_k$ графа $G_l(Q_l, X_l)$, то пользователю направляется сообщение о завершении решения его задачи и результаты ее решения.

21. Задача $Z_l$ помечается на ДО как решенная.

## 5. Алгоритм работы агента вычислительного ресурса при мультиагентном диспетчировании ресурсов

Основной целью работы агента вычислительного ресурса $AR_p$ является поиск задач для загрузки «своего» ресурса $R_p$ полезной работой. Для этого он периодически опрашивает ДО и в случае обнаружения новой нити $H_j^m$, выставленной на исполнение агентом $AZ_l$ задачи $Z_l$, делает попытку включения в состав сообщества $R_l$ по его выполнению. Для этого агент $AR_p$ выделяет в нити $H_j^m =< q_1^j, \ldots, q_k^j >$ поднить $H_{jp}^m =< q_1^j, \ldots, q_b^j >$ $(b \le k)$ подзадачи, приписанные вершины которой входят в множество $O_p$, т.е. в множество подзадач, выполняемых ресурсом $R_p$ (см. рис. 4).

Агент $AR_p$ определяет момент времени $t_{\text{нр}}^m$, когда он может приступить к исполнению поднити $H_{jp}^m$, т.е. когда он освободится от исполнения ранее закрепленных за ним операций, а также момент времени $t_{\text{кр}}^m$ окончания исполнения поднити $H_{jp}^m$, определяемый согласно выражению (2).

Выделенная таким образом поднить $H_{jp}^m$ направляется агенту $AZ_l$ задачи $Z_l$ в качестве «предложения» агента $AR_p$ по участию в сообществе $R_l$ по выполнению задачи $Z_l$. Если «предложение» агента $AR_p$ наилучшее среди всех поступивших, т.е. значение $E_p$, определяемое с помощью выражения (5), для нее максимально, ему направляется подтверждение о закреплении поднити $H_{jp}^m$ за ним.

Как только наступает запланированный момент $t_{\text{нр}}^m$ начала исполнения нити $H_{jp}^m$, агент $AR_p$ осуществляет ее выполнение (см. рис. 7). Перед тем, как перейти к очередной подзадаче $O_i^j$, приписанной вершине $q_i^j$ $(i = 1, \ldots, b)$ нити $H_{jp}^m$, агент $AR_p$ проверяет наличие требуемых для ее исполнения исходных данных. Если какие-либо исходные данные еще не поступили, то агент $AR_p$ переходит в режим ожидания. Как только все исходные данные, необходимые для решения подзадачи $O_i^j$, поступают, агент $AR_p$ инициирует решение подзадачи $O_i^j$ с помощью «своего» ресурса $R_p$. После выполнения всех подзадач, приписанных вершинам нити $H_{jp}^m$, агент $AR_p$ сообщает агенту задачи $Z_l$ о завершении исполнения нити $H_{jp}^m$.

Формализуем алгоритм, соответствующий описанному выше процессу:

Алгоритм 2

1. Агент $AR_p$ опрашивает ДО в поисках работы для «своего» ресурса $R_p$.

2. Если агент $AR_p$ обнаружил на ДО выставленную на исполнение агентом $AZ_l$ нить $H_j^m =< q_1^j, \ldots, q_k^j >$ задачи $Z_l$, то он выделяет в нити $H_j^m$ поднить $H_{jp}^m =< q_1^j, \ldots, q_b^j >$ $(b \le k)$, вершины которой удовлетворяют условию $O_i^j \subseteq$ $\subseteq O_p$ $(j = 1, \ldots, b)$, где $O_i^j$ — подзадачи, приписанная вершине $q_i^j$ нити $H_{jp}^m$.

3. Агент $AR_p$ рассчитывает моменты времени $t_{\text{нр}}^m$, когда он может приступить к исполнению нити $H_j^m$, и $t_{\text{кр}}^m$ завершения нити как

$$t_{\text{кр}}^m = t_{\text{нр}}^m + \sum_{i=1}^{b} \frac{v_i^j}{S_p(O_i)} + \frac{d_{b,b+1}^j}{Y_p}.$$

4. Полученные значения направляются агенту $AZ_l$.

5. Если агент $AR_p$ получает от агента $AZ_l$ подтверждение о его включении в сообщество $R_l$, то он в момент времени $t_{\text{нр}}^m$ приступает к выполнению вершин нити $H_{jp}^m =< q_1^j, \ldots, q_b^j >$.

6. Введем индекс $i = 1$.

7. Агент $AR_p$ проверяет наличие всех исходных данных, необходимых для решения подзадачи $O_i^j$, приписанных вершине $q_i^j \in H_{jp}^m$. Если исходные данные еще не поступили, агент $AR_p$ переходит в режим ожидания.

8. Как только все необходимые исходные данные поступили, агент $AR_p$ решает подзадачу $O_i^j$ с помощью «своего» ресурса $R_p$.

9. $i = i + 1$, если $i < b$, то перейти к 7, иначе

10. Агент $AR_p$ сообщает агенту задачи $Z_l$ о завершении выполнения подзадач нити $H_{jp}^m$.

11. Переход к 1.

## 6. Исследование эффективности предложенных метода и алгоритмов

Для проведения экспериментальных исследований предложенных в статье метода и алгоритмов была разработана распределенная программная модель ОВС с мультиагентным диспетчером, состоящая из программных агентов ресурса и задачи, доски объявлений, визуальных оболочек пользователя и администратора.

Интерфейс визуальной оболочки пользователя представлен на рис. 8.

С целью развертывания разработанной распределенной программной модели ОВС с мультиагентным диспетчером был собран экспериментальный стенд, состоящий из 10 вычислительных модулей, включающих в себя от 1 до 16 физических процессорных ядер, объединенных общей сетью с пропускной способностью до 1000 мегабит в секунду.

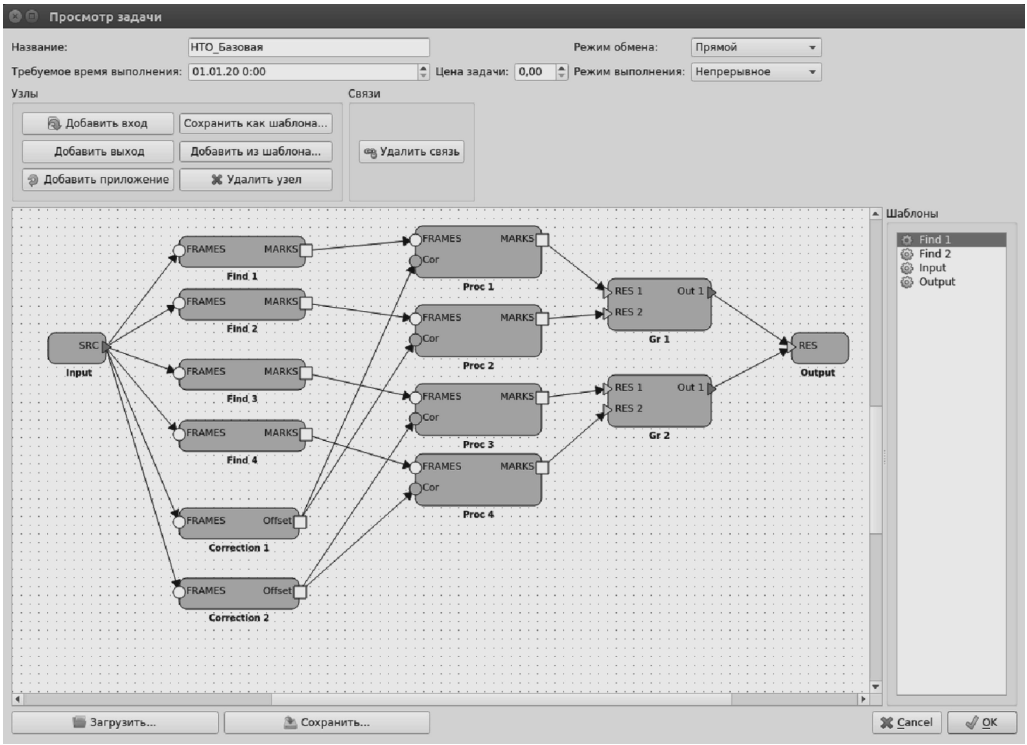


Рис. 8. Визуальная оболочка пользователя.

Для оценки эффективности предложенного метода и алгоритмов работы мультиагентного диспетчера ОВС при решении поступающих задач потребовалось провести исследование работы ОВС в различных режимах работы. В связи с тем, что моделируемая вычислительная среда представляет собой сложную систему, для полноценного исследования такой системы необходима целенаправленная организация эксперимента. При этом эффективность работы ОВС оценивалась с применением критерия Y — отношения суммарного времени решения всех задач множества Z с применением предложенных в настоящей статье алгоритмов к суммарному времени решения всех задач множества Z с применением распределения, полученного с помощью алгоритмов распределения задач, предложенных в [16], выраженного в процентах. Чем меньше будет значение критерия Y, полученного при исследовании, тем эффективнее работают предложенные алгоритмы.

Значение критерия Y будет в большой степени зависеть от таких параметров ОВС как: количество вычислительных ресурсов (ВР) в ОВС; производительность ВР; пропускная способность канала связи ВР с облачной инфраструктурой; частота появления новых пользовательских задач на ДО; количество пользовательских задач, направленных в ОВС; количество подзадач в пользовательской задаче; вычислительная трудоемкость подзадач; объемы данных, передаваемых между подзадачами.

Для того чтобы оценить эффективность разработанных алгоритмов мультиагентного диспетчирования ОВС, необходимо оценить значение критерия Y для всевозможных комбинаций перечисленных выше параметров. Однако сделать это будет крайне трудно, поскольку каждый из этих параметров может существенно измениться в рамках некоторого интервала, а общее число их комбинаций будет ограниченным. Поэтому была разработана методика проведения экспериментов, позволяющая сократить общее количество экспериментов, необходимое для оценки критериев и эффективности работы предложенных алгоритмов, без существенного снижения их достоверности [24]. При этом были предложены следующие интервалы изменения параметров моделирования (чтобы сделать процесс моделирования более близким к реальным условиям, было решено ввести 20%-ное значение возможной погрешности параметров): параметр P1 — количество вычислительных ресурсов в ОВС (10, 100, 500, 1000); параметр P2 — количество решаемых в ОВС пользовательских задач (10 (малое), 50 (среднее), 100 (большое)); параметр S1 — производительность ресурсов ОВС (низкая (−) — 10, высокая (+) — 100); параметр S2 — пропускная способность каналов связи между ресурсами ОВС (низкая (−) — 10, высокая (+) — 100); параметр S3 — трудоемкость пользовательских задач (низкая (−) — 10, высокая (+) — 100); параметр S4 — количество данных, передаваемых между подзадачами (низкая (−) — 10, высокая (+) — 100); параметр T1 – количество подзадач в задаче (от 5 до 25). Разбиение экспериментов было предложено реализовать следующим образом. Комбинации параметров P1 и P2 определяют серию экспериментов. В рамках каждой из серий проводится по 16 экспериментов (один эксперимент для

каждой из комбинаций параметров S1, S2, S3 и S4), каждый из экспериментов проводится по три раза с различными значениями параметра T1.

Наименьшее значение 14% критерия Y было получено при P1 = 50, P2 = = 1000, S1 = 10, S2 = 10, S3 = 100, S4 = 10, что подтверждает высокую эффективность разработанных алгоритмов для небольших ОВС с высокой загрузкой. Наиболее высокое значение 84% критерия Y было получено при P1 = 1000, P2 = 10, S1 = 10, S2 = 100, S3 = 100, S4 = 100, что соответствует слабо загруженной задачами ОВС, содержащей большое количество вычислительных ресурсов, являющейся наименее подходящей для применения предложенных в статье эвристических метода и алгоритмов. Итоги всех проведенных экспериментов показывают, что среднее значение критерия Y составило 53%. Это позволяет сделать вывод, что в общем виде даже при большом количестве ресурсов в ОВС предложенные в настоящей статье алгоритмы позволяют сократить суммарное время решения всех задач множества Z.

## 7. Заключение

Проведенные исследования показали, что основным преимуществом предлагаемого мультиагентного метода диспетчирования ресурсов в ОВС является то, что вычислительный процесс адаптируется к актуальным вычислительным возможностям гетерогенных ресурсов, входящих в ее состав. По сравнению с классической централизованной организацией диспетчера облачной среды в данном случае упрощаются требования к служебным серверам (доскам объявлений), что позволяет существенно снизить стоимость облачных вычислений, а также упростить процесс масштабирования облачной среды.

Предложенные в статье метод и алгоритмы позволяют повысить эффективность и гибкость использования вычислительного оборудования в ОВС за счет возможности адаптивного распределения задач в множестве гетерогенных ресурсов с динамически изменяемыми параметрами, при этом не требуют от пользователей дополнительной информации о необходимом времени решения задач.

## СПИСОК ЛИТЕРАТУРЫ


1. *Alam T.* Cloud Computing and its role in the Information Technology // IAIC Transactions on Sustainable Digital Innovation (ITSDI). 2020. Vol. 1. No. 2. P. 108–115.
2. *Батаев А.В.* Оценка мирового рынка облачных технологий в финансовой сфере // Вектор экономики. 2019. № 6. С. 91–91.
3. *Караев А.В., Емельянов Д.О., Барановская Т.П.* Актуальность и особенности внедрения ИТ-сервисов с применением облачных технологий // Информационное общество: современное состояние и перспективы развития. 2020. С. 387–390.

4. *Fink A., Homberger J.* An ant-based coordination mechanism for resource-constrained project scheduling with multiple agents and cash flow objectives // Flexible Services and Manufacturing Journal. 2013. Vol. 25. No. 1. P. 94–121.
5. *Verma A., Kaushal S.* A hybrid multi-objective Particle Swarm Optimization for scientific workflow scheduling // Parallel Comput. 2017. Vol. 62. P. 1–19.
6. *Yuan X., Liu J., Wimmers M.O.* A multi-agent genetic algorithm with variable neighborhood search for resource investment project scheduling problems // IEEE Congress on Evolutionary Computation (CEC). IEEE, 2015. P. 23–30.
7. *Bertsekas D.P.* Feature-based aggregation and deep reinforcement learning: A survey and some new implementations // IEEE/CAA J. Autom. Sin. 2019. Vol. 6. No. 1. P. 1–31.
8. *Habibi F., Barzinpour F., Sadjadi S.* Resource-constrained project scheduling problem: review of past and recent developments // J. Project Management. 2018. Vol. 3. No. 2. P. 55–88.
9. *Mao H., Alizadeh M., Menache I., Kandula S.* Resource management with deep reinforcement learning // HotNets 2016 – Proceedings of the 15th ACM Workshop on Hot Topics in Networks. 2016. P. 50–56.
10. *Xue L., Sun C., Wunsch D., et al.* An adaptive strategy via reinforcement learning for the prisoner's dilemma game // IEEE/CAA J. Autom. Sin. 2018. V. 5. No. 1. P. 301–310.
11. *Zhan Y., Ammar H.B., Taylor M.E.* Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer // IJCAI International Joint Conference on Artificial Intelligence. 2016. Vol. 2016-Janua. P. 2315–2321.
12. *Wang H., Huang T., Liao X., et al.* Reinforcement Learning for Constrained Energy Trading Games with Incomplete Information // IEEE Trans. Cybern. 2017. Vol. 47. No. 10. P. 3404–3416.
13. *Zheng L., Yang J., Cai H., et al.* Magent: A many-agent reinforcement learning platform for artificial collective intelligence // Proceedings of the AAAI Conference on Artificial Intelligence. 2018. Vol. 32. No. 1. P. 8222–8223.
14. *Lowe R., Wu Y.I., Tamar A., et al.* Multi-agent actor-critic for mixed cooperative-competitive environments // Advances in neural information processing systems. 2017. Vol. 30. P. 1–12.
15. *Каляев И.А., Каляев А.И., Коровин Я.С.* Алгоритм мультиагентного диспетчирования ресурсов в гетерогенной облачной среде // Вычислительные технологии. 2016. Т. 21. № 5. С. 38–53.
16. *Каляев А.И., Каляев И.А.* Метод мультиагентного диспетчирования ресурсов в облачных вычислительных средах // Известия Российской академии наук. Теория и системы управления. 2016. № 2. С. 51–57.
17. *Каляев И.А., Капустян С.Г.* Метод мультиагентного управления «умным» интернет-производством // Робототехника и техническая кибернетика. 2018. № 1. С. 34–48.
18. *Аблялимов О.С.* О решении задачи оптимизации методом динамического программирования // Universum: технические науки. 2020. № 9-1(78). С. 16–18.
19. *Канцедал С.А., Костикова М.В.* Динамическое программирование для задачи коммивояжера //Автоматизированные системы управления и приборы автоматики. 2014. № 166. С. 15–20.

20. *Колемаев В.А.* Математическая экономика. М.: ЮНИТИ – ДАНА, 2002.

21. *Kalyaev A.I., Korovin Y.S.* Adaptive Multiagent Organization of the Distributed Computations / AASRi Procedia. 2014. Vol. 6. P. 49–58.
(URL: http://dx.doi.org/10.1016/j.aasri.2014.05.008)

22. INTRODUCTION TO ALGORITHMS, Second Edition Thomas H. Cormen, Charles E Leiserson, Ronald L. Rivest, Clifford Stein. (URL: http://www.mif.vu.lt/-valdas/ALGORITMAI/LITERATURA/Cormen/Cormen.pdf).

23. *Рейнгольд О., Нвергельт Ю., Део Н.* Комбинаторные алгоритмы. Теория и практика. М.: Мир, 1980.

24. *Каляев А.И., Хисамутдинов М.В.* Программа и методики экспериментальных исследований методов и алгоритмов работы распределенной вычислительной системы с помощью программной модели // Наука и современность: сборник материалов V Международной научно-практической конференции. 2016. С. 44–45.

*Статья представлена к публикации членом редколлегии А.А. Галяевым.*

Поступила в редакцию 15.10.2021

После доработки 14.02.2022

Принята к публикации 28.04.2022