УДК 004.75 DOI: 10.17587/mau.16.585-598

А. И. Каляев, канд. техн. наук, ст. науч. сотр., anatoly@ kalyaev.net,
И. А. Каляев, чл.-корр. РАН, д-р техн. наук, проф., директор, kaliaev@mvs.sfedu.ru,
Научно-исследовательский институт многопроцессорных вычислительных систем
имени академика А. В. Каляева Южного федерального университета, г. Таганрог

Метод децентрализованного управления распределенной системой при выполнении потока заданий

Рассматривается задача управления распределенной системой с сетевой архитектурой, состоящей из множества объектов, объединенных каналом связи и совместно участвующих в выполнении потока потребительских заданий. При этом полагается, что каждое потребительское задание состоит из набора связных операций и может поступать в произвольный момент времени. Предлагается метод децентрализованного управления распределенной системой с помощью множества устройств управления отдельных объектов, входящих в ее состав. Приводятся алгоритмы функционирования устройств управления отдельных объектов распределенной системы для четырех вариантов исходной постановки задачи управления. В заключении приводятся результаты экспериментальных исследований предложенных алгоритмов на программной модели распределенной системы.

Ключевые слова: *распределенная система, поток заданий, децентрализованное управление, сетевое планирование операций, автоматическое распределение*

Введение

В настоящее время все большую актуальность приобретает проблема управления распределенными системами, состоящими из некоторого множества объектов (подсистем), совместно участвующих в выполнении общего задания, например, таких как: группа роботов, участвующих в сборочном производстве или выполняющих совместную боевую задачу; распределенные управляющие и вычислительные системы, состоящие из множества процессорных узлов, которые участвуют в совместном решении общей управленческой или вычислительной задачи; системы типа "умный дом" или "умный город", состоящие из множества датчиков и исполнительных устройств, совместно решающих задачи по обслуживанию дома или города [1, 2]. Данная проблема особенно актуальна в такой перспективной сфере, как микробототехника, когда каждый робот группы может осуществлять ограниченный набор простейших действий, и поэтому выполнение сложных заданий возможно только за счет их совместного, группового взаимодействия [3].

Основными преимуществами таких распределенных систем являются:

1. Сокращение времени выполнения задания вследствие возможности распараллеливания выполнения отдельных его операций между объектами, входящими в систему.

2. Повышение надежности и отказоустойчивости системы, поскольку выход из строя отдельного объекта, входящего в систему, не приводит к выходу из строя всей системы в целом, а задания, возложенные на данный объект, могут быть "перераспределены" между остальными объектами системы.

Однако управление такой распределенной системой, особенно в реальном масштабе времени, представляет достаточно сложную проблему. До недавнего времени эта проблема решалась посредством специально выделенного центрального устройства управления (ЦУУ), функции которого заключаются в мониторинге текущего состояния всех объектов системы, распределении операций между ними при выполнении заданий, поступающих от потребителей, и формировании соответствующих команд управления для реализации этих операций (рис. 1).

Однако такая централизованная организация устройства управления распределенной системой имеет целый ряд недостатков. Во-первых, при большом числе объектов в системе мониторинг их текущего состояния, распределение операций и формирование команд управления из одного ЦУУ в реальном времени становится практически невозможным. Во-вторых, существенно затрудняется масштабирование системы (т. е. добавление в ее

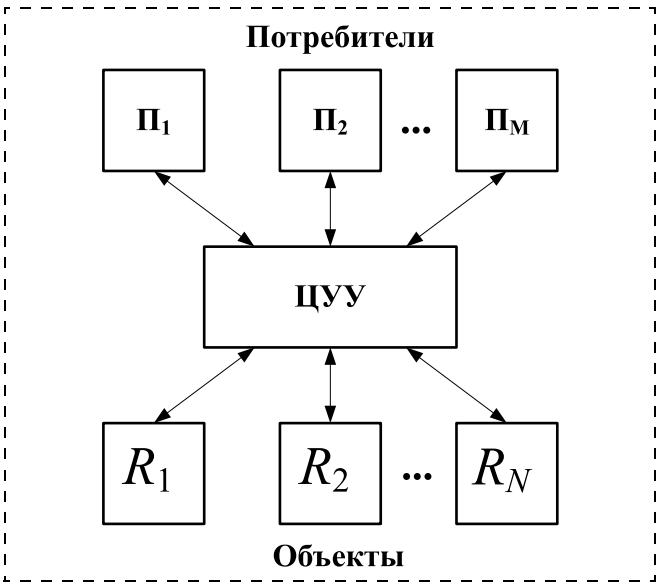

Рис. 1. Централизованное управление распределенной системой

состав новых объектов), поскольку при этом необходимо полностью менять не только программу работы ЦУУ, но и архитектуру его связей. И наконец, в-третьих, распределенная система с ЦУУ становится малонадежной, поскольку выход из строя ЦУУ приводит к выходу из строя всей системы в целом. Все перечисленные проблемы многократно усложняются в случае, когда распределенная система должна выполнять не одиночное задание, а поток потребительских заданий, поступающих в заранее неизвестные моменты времени.

Всех этих недостатков можно избежать, если использовать принципы децентрализованного (коллективного) управления распределенными системами [4, 5]. При этом предполагается, что каждый из объектов распределенной системы обладает своим индивидуальным устройством управления (УУ), а общая координация действий всех объектов, входящих в состав системы, при решении общей задачи осуществляется путем их коллективного взаимодействия посредством некоторого сетевого канала связи (рис. 2).

Использование принципов децентрализованного управления распределенной системой обеспечивает: во-первых, высокую отказоустойчивость системы, поскольку в ней отсутствует "узкое горло" в виде ЦУУ, а выход из строя любого из объектов не приводит к выходу из строя всей системы в целом; во-вторых, возможность практически неограниченного увеличения (масштабирования) числа объектов в системе путем их простого подключения к сетевому каналу связи и, наконец, в-третьих, снижение вычислительной нагрузки на УУ отдельного объекта, поскольку им решается только задача управления данным объектом, а не всей системой в целом, что, в свою очередь, упрощает проблему обеспечения режима реального времени при управлении распределенной системой.

Однако, вместе с тем, использование этих принципов требует создание новых методов и алгоритмов управления распределенной системой. Именно разработке одного из таких методов посвящена данная статья.

1. Постановка задачи

Предположим, что в состав распределенной системы $\boldsymbol{R}$ входит N объектов $R_1, R_2, ..., R_N$, каждый из которых может выполнять некоторый набор операций $\boldsymbol{A}_i = <A_1^i, A_2^i, ..., A_L^i>$ ($i = 1, 2, ..., N$), причем известно, что объект R_i выполняет операцию A_j^i за время $t_i(A_j^i)$ ($j = 1, 2, ..., L$).

Будем считать, что система $\boldsymbol{R}$ должна выполнять некоторое множество (поток) различных заданий $\boldsymbol{Z} = <Z_1, Z_2, ..., Z_M>$, которые могут поступать от различных потребителей в случайные моменты времени. При этом каждое задание $Z_l \in \boldsymbol{Z}$ представляется в виде некоторого ациклического графа операций $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ (рис. 3),вершине $q_j \in \boldsymbol{Q}_l$ которого приписана некоторая операция A_j, принадлежащая множеству $\boldsymbol{A} = \bigcup_{i=1}^{N} \boldsymbol{A}_i$, а дуга $x(q_j, q_{j+1})$ определяет, что результат операции A_j, приписанной вершине q_j, необходим для выполнения операции A_{j+1}, приписанной вершине q_{j+1}. Кроме того, потребитель ус-

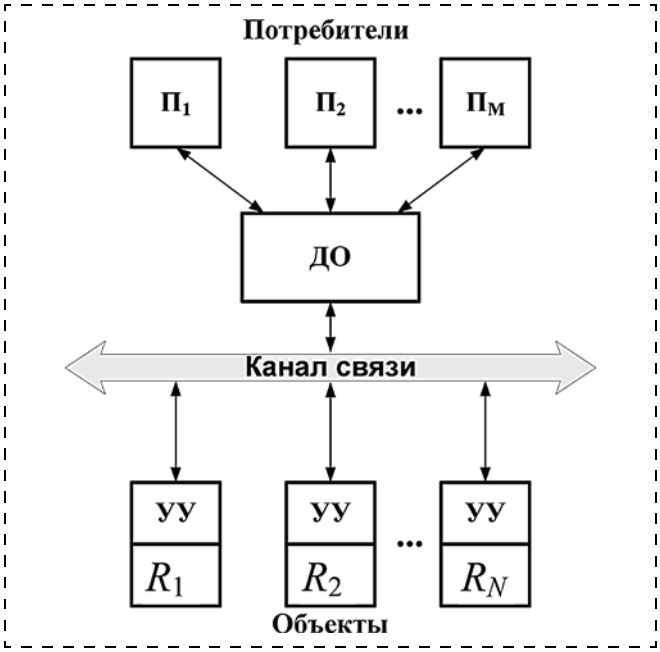

Рис. 2. Децентрализованное управление распределенной системой

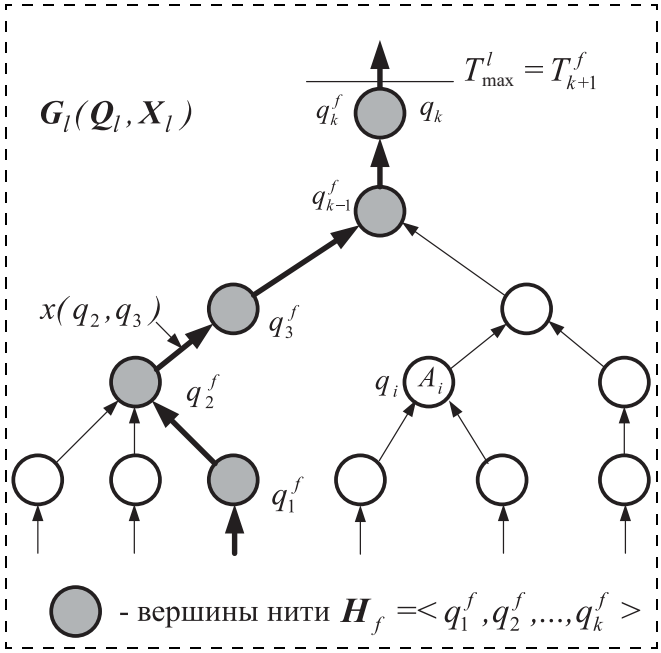

Рис. 3. Граф $G_l(Q_l, X_l)$ задания Z_l

танавливает требуемый момент времени $T^l_{\max}$, к которому он желает получить результат выполнения своего задания Z_l.

При этом возникает вопрос: каким образом и в каком виде распределенная система $\boldsymbol{R}$ должна получать задания от потребителей? При централизованной организации взаимодействие потребителей и объектов множества $\boldsymbol{R}$ осуществляется через ЦУУ (см. рис. 1). При децентрализованном управлении взаимодействие потребителей и объектов множества $\boldsymbol{R}$ может осуществляться посредством некоторого специально выделенного узла, подключенного к общему каналу связи и играющего роль "доски объявлений" (ДО), на которой потребитель может размещать свое задание (см. рис. 2).

Прежде чем разместить задание Z_l на ДО, потребитель должен представить его в некотором формализованном виде — в виде дескриптора. Дескриптор задания Z_l должен включать следующие данные:

- граф $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ задания Z_l, представленный, например, в виде таблицы сложности;
- список вершин множества $\boldsymbol{Q}_l$ и приписанных им операций множества $\boldsymbol{A}$;
- момент времени $T^l_{\max}$, к которому задание должно быть выполнено;
- список объектов множества $\boldsymbol{R}$, участвующих в выполнении данного задания (заполняется в процессе распределения операций).

Сформированный таким образом дескриптор задания Z_l размещается потребителем на ДО.

Цель работы распределенной системы $\boldsymbol{R}$ заключается в выполнении всех заданий потребителей, размещенных на ДО, к заданным моментам времени. При этом для выполнения каждого конкретного задания Z_l среди всего множества объектов $\boldsymbol{R}$ должно быть сформировано некоторое подмножество $\boldsymbol{R}_l \subseteq \boldsymbol{R}$, которое в дальнейшем будем называть сообществом по выполнению задания Z_l.

Очевидно, что проблема выполнения поставленного потребителем задания Z_l в распределенной системе $\boldsymbol{R}$ может быть решена в два этапа:

1. Сначала необходимо создать сообщество $\boldsymbol{R}_l \subseteq \boldsymbol{R}$ по выполнению данного задания Z_l и распределить операции, приписанные вершинам графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, задания Z_l между объектами, входящими в это сообщество, таким образом, чтобы обеспечить выполнение всего задания к требуемому моменту времени $T^l_{\max}$. При этом под распределением (сетевым планированием) операций понимается закрепление операций, приписанных вершинам графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ задания Z_l, за конкретными объектами, входящими в состав сообщества, с привязкой времени их выполнения к определенным моментам времени.

2. После того как операции задания Z_l распределены, каждый объект $R_j \subseteq \boldsymbol{R}_l$ самостоятельно реализует закрепленные за ним операции, контролируя при этом временной график их выполнения.

Основную трудность при этом, очевидно, составляет задача первого этапа, а именно, распределения (сетевого планирования) операций между объектами системы. Проблеме распределения операций (сетевого планирования) между множеством объектов посвящено достаточно большое число исследований [6—8]. Поэтому, если графы $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ ($l = 1, 2, ..., M$) всех заданий множества $\boldsymbol{Z}$ известны и состав распределенной системы $\boldsymbol{R}$ не изменяется, то с помощью известных методов, в принципе, такие расписания могут быть составлены заранее для каждого из заданий множества $Z_l \in \boldsymbol{Z}$ и могут просто храниться в памяти объектов R_i ($i = \overline{1, N}$).

Однако это идеализированный случай. В реальности графы операций выполняемых системой заданий $\boldsymbol{Z} = \langle Z_1, Z_2, ..., Z_M \rangle$ могут быть заранее неизвестны, а состав объектов, входящих в систему $\boldsymbol{R}$, может изменяться непредсказуемым образом (например, некоторые из них могут выходить из строя). В этом случае возникает необходимость разработки метода автоматического распределения операций для всех объектов системы $\boldsymbol{R}$ при выполнении потока заранее неизвестных заданий $\boldsymbol{Z}$ с помощью множества их УУ.

В зависимости от конкретных условий данная задача может иметь четыре различных варианта постановки.

1. В простейшем варианте все объекты R_1, R_2, ..., R_N системы $\boldsymbol{R}$ одинаковы и могут выполнять одинаковые наборы операций, т. е. $\boldsymbol{A}_i = \boldsymbol{A}_j$, ($i = 1, 2, ..., N, j = 1, 2, ..., i - 1, i + 1, ..., N$), причем операция $A_s \in \boldsymbol{A}$ выполняется всеми объектами R_i ($i = 1, 2, ..., N$) за одинаковое время, т. е. $t_i(A_s) = t_j(A_s)$ ($i = 1, 2, ..., N$, $j = 1, 2, ..., i - 1, i + 1, ..., N$).

2. В более сложном варианте наборы операций, выполняемых всеми объектами системы $\boldsymbol{R}$, одинаковы, т. е. $\boldsymbol{A}_i = \boldsymbol{A}_j$ ($i = 1, 2, ..., N, j = 1, 2, ..., i - 1, i + 1, ..., N$), однако время выполнения идентичных операций $A_s \in \boldsymbol{A}$ различными объектами $R_i \in \boldsymbol{R}$ и $R_j \in \boldsymbol{R}$ различно, т. е. $t_i(A_s) \neq t_j(A_s)$ ($i = 1, 2, ..., N$, $j = 1, 2, ..., i - 1, i + 1, ..., N$).

3. В третьем варианте наборы операций, выполняемых различными объектами системы $\boldsymbol{R}$, также различны, т. е. $\boldsymbol{A}_i \neq \boldsymbol{A}_j$ ($i = 1, 2, ..., N, j = 1, 2, ..., i - 1, i + 1, ..., N$), хотя они могут и пересекаться, т. е. $\boldsymbol{A}_i \cap \boldsymbol{A}_j \neq \varnothing$. При этом время выполнения идентичных операций $A_s \in \boldsymbol{A}$ для всех объектов $\boldsymbol{R}_i$ ($i = \overline{1, N}$) одинаковое, т. е. $t_i(A_s) = t_j(A_s)$ ($i = 1, 2, ..., N$, $j = 1, 2, ..., i - 1, i + 1, ..., N$).

4. Наконец, в самом сложном варианте объекты системы $\boldsymbol{R}$ выполняют различные наборы операций, т. е. $\boldsymbol{A}_i \neq \boldsymbol{A}_j$ и $\boldsymbol{A}_i \cap \boldsymbol{A}_j \neq \varnothing$, причем время выполнения идентичных операций $A_s \in \boldsymbol{A}_i$ различными объектами $R_i \in \boldsymbol{R}$ и $R_j \in \boldsymbol{R}$ также различно, т. е. $t_i(A_s) \neq t_j(A_s)$ ($i = 1, 2, ..., N, j = 1, 2, ..., i - 1, i + 1, ..., N$).

Рассмотрим более подробно алгоритмы децентрализованного управления объектами распределенной системы $\boldsymbol{R}$ при выполнении потока зада-

ний $\boldsymbol{Z}$, поступающих от различных потребителей в случайные моменты времени, для всех вариантов исходной постановки.

Вариант 1

Как показано выше, в данном варианте все объекты R_p (p = 1, 2, ..., N) множества $\boldsymbol{R}$ совершенно идентичны, т. е. могут выполнять одинаковый набор операций $\boldsymbol{A} = <A_1, A_2, ..., A_L>$ за одинаковое время $t_p(A_s)$ (s = 1, 2, ..., L).

Прежде чем приступить к разработке алгоритма, введем понятие "нити". Под нитью будем понимать некоторую последовательность вершин $\boldsymbol{H}_f = <q_1^f, q_2^f, ..., q_k^f>$ графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ задания Z_l, в которой вершины q_j^f и q_{j+1}^f (j = 1, 2, ..., k − 1) соединены дугой $x(q_j^f, q_{j+1}^f)$ (рис. 3). Иными словами, нить определяет некоторую последовательность операций задания Z_l, в которой каждая последующая операция использует результат выполнения предыдущей операции. Очевидно, что операции, приписанные вершинам нити $\boldsymbol{H}_f$, могут выполняться только последовательно. При этом под длиной t_f нити $\boldsymbol{H}_f$ будем понимать суммарное время, затрачиваемое на ее выполнение, т. е.

$$t_f = \sum_{i=1}^{k} (t_p(A_i) + t_{\text{п}}(A_j, A_{j+1})),$$

где $t_p(A_i)$ — время, затрачиваемое объектом $R_p \in \boldsymbol{R}$ на выполнение операции A_i, приписанной вершине $q_i^f \in \boldsymbol{H}_f$ (i = 1, 2, ..., k); $t_{\text{п}}(A_i, A_{i+1})$ — время, затрачиваемое на передачу результатов операции A_i, выполняемой объектом $R_p \in \boldsymbol{R}$, объекту $R_c \in \boldsymbol{R}$, выполняющему следующую по очереди операцию A_{i+1} нити $\boldsymbol{H}_f$; $t_{\text{п}}(A_k, A_{k+1})$ — время, затрачиваемое на передачу результата исполнения операции, приписанной последней вершине q_k^f нити $\boldsymbol{H}_f$ (т. е. результата исполнения всей нити $\boldsymbol{H}_f$) либо потребителю, если q_k^f — конечная вершина графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, т. е. $q_k^f = q_k$, либо объекту, исполняющему сопряженную нить (рис. 3).

Для упрощения дальнейших построений будем считать, что $t_{\text{п}}(A_i, A_{i+1}) = 0$, если операции A_i и A_{i+1} выполняет один и тот же объект $R_f \in \boldsymbol{R}$, и $t_{\text{п}}(A_i, A_{i+1}) = t_{\text{п}}$, если операции A_i и A_{i+1} выполняют различные объекты множеств $\boldsymbol{R}$, либо A_{i_f} — операция, приписанная последней вершине q_k^f нити $\boldsymbol{H}_f$.

Очевидно, что если все объекты множества $\boldsymbol{R}$ выполняют идентичные операции множества $\boldsymbol{A}$ за одинаковое время, то выполнение всех операций, приписанных вершинам нити $\boldsymbol{H}_f$, целесообразно осуществлять с помощью одного и того же объекта $R_p \in \boldsymbol{R}$. Действительно, в этом случае время $t_{\text{п}}(A_i, A_{i+1})$, затрачиваемое на передачу результатов операций между вершинами нити $\boldsymbol{H}_f$, будет равно нулю, и, соответственно, длина (время выполнения) нити будет минимальной, т. е.

$$t_f = \sum_{i=1}^{k} t_p(A_i) + t_{\text{п}}, \qquad (1)$$

где $t_{\text{п}}$ — время, затрачиваемое на передачу результата исполнения последней вершины нити $\boldsymbol{H}_f$.

При этом, если требуемый момент времени T_{k+1}^f исполнения последней вершины q_k^f нити $\boldsymbol{H}_f$ известен (рис. 3), то, зная время $t_p(A_i)$ (i = 1, 2, ..., k) выполнения отдельных операций, приписанных вершинам нити, можно определить требуемый момент времени начала выполнения операции A_d, приписанной любой вершине $q_d^f \in \boldsymbol{H}_f$ (d = 1, 2, ..., k), как

$$T_d^f = T_{k+1}^f - \left(\sum_{j=d}^{k} t_p(A_j) + t_{\text{п}} \right). \qquad (2)$$

Отметим, что при размещении задания Z_l на ДО требуемый момент времени исполнения, определяемый установленным потребителем временем $T_{\max}^l$ выполнения всего задания Z_l, приписан только конечной вершине $q_k \in Q_l$ графа (рис. 3).

Исходя из этих соображений можно предложить следующую процедуру распределения (сетевого планирования) операций между объектами системы $\boldsymbol{R}$ при выполнении потока заданий $\boldsymbol{Z}$.

Объекты, не задействованные в выполнении каких-либо заданий, обращаются к ДО в поисках работы. Если объект $R_p \in \boldsymbol{R}$ обнаруживает на ДО дескриптор задания Z_l, то он оценивает возможность своего участия в сообществе по его выполнению. Для этого объект R_p выделяет в графе $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, хранящемся в дескрипторе задания Z_l, наиболее длинную нить $\boldsymbol{H}_f = <q_1^f, q_2^f, ..., q_k^f>$ согласно выражению (1), конечной вершине которой приписан момент времени ее исполнения T_{k+1}^f. Последнее может быть осуществлено с помощью одного из известных алгоритмов поиска максимального пути на графе, например, волнового алгоритма [9]. Отметим, что изначально, в момент размещения дескриптора задания Z_l на ДО, таковой нитью является нить $\boldsymbol{H}_1 = <q_1^1, q_2^1, ..., q_k^1>$, которая начинается на одной из входных вершин графа, а заканчивается на выходной вершине графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, т. е. $q_k^1 = q_k$ (рис. 3). Очевидно, что выполнение данной нити должно быть завершено к заданному потребителем

моменту $T^l_{\max}$ выполнения всего задания Z_l в целом, который также содержится в ее дескрипторе, т. е. $T^1_{k+1} = T^l_{\max}$.

На основании данных о времени $t_p(A_i)$ (i = 1, 2, ..., k) выполнения отдельных операций, входящих в выделенную нить $\boldsymbol{H}_1$, объект R_p определяет согласно выражению (2) моменты времени, когда необходимо приступить к выполнению каждой из операции нити $\boldsymbol{H}_1$, чтобы успеть завершить исполнение всей нити к заданному моменту времени T^l_{k+1}, как

$$T^1_d = T^1_{k+1} - \left(\sum_{i=d}^{k} t_p(A_i) + t_{\text{п}} \right), \ (d = 1, 2, ..., k).$$

Если при этом оказывается, что $T^1_1 < T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, T^1_1 — требуемый момент времени начала выполнения первой операции нити $\boldsymbol{H}_1$, то это означает, что объект R_p не может обеспечить выполнение всей последовательности операций нити $\boldsymbol{H}_1 = <q^1_1, q^1_2, ..., q^1_k>$ к заданному моменту времени T^1_{k+1}. Поскольку в данном варианте постановки задачи мы приняли, что все объекты группы одинаковы и выполняют все идентичные операции за одинаковое время, то это также означает, что никакой другой объект множества $\boldsymbol{R}$ также не сможет выполнить данную нить к заданному моменту времени T^1_{k+1}. Таким образом, задание Z_l не может быть выполнено, и поэтому оно снимается с ДО, а потребителю посылается сообщение о невозможности выполнения его задания за отведенное время. После этого объект R_p вновь переходит к опросу ДО в поисках работы.

Если же условие $T^1_1 \geqslant T_{\text{тек}}$ выполняется, то в этом случае объект R_p принимает на себя исполнение последовательности операций, приписанных вершинам нити $\boldsymbol{H}_1$. При этом объект R_p осуществляет модификацию дескриптора задания Z_l на ДО, а именно:

1) его номер p записывается в список членов сообщества по выполнению задания Z_l;

2) вершины, входящие в нить $\boldsymbol{H}_1$, исключаются из графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ (т. е. в таблице смежности удаляются соответствующие строки и столбцы), в результате чего формируется новый граф $\boldsymbol{G}^1_l(\boldsymbol{Q}^1_l, \boldsymbol{X}^1_l) = \boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)/\boldsymbol{H}_1$ (рис. 4);

3) всем вершинам графа $\boldsymbol{G}^1_l(\boldsymbol{Q}^1_l, \boldsymbol{X}^1_l)$, инцидентным вершинам нити $\boldsymbol{H}_1$, приписываются требуемые моменты времени их исполнения, которые

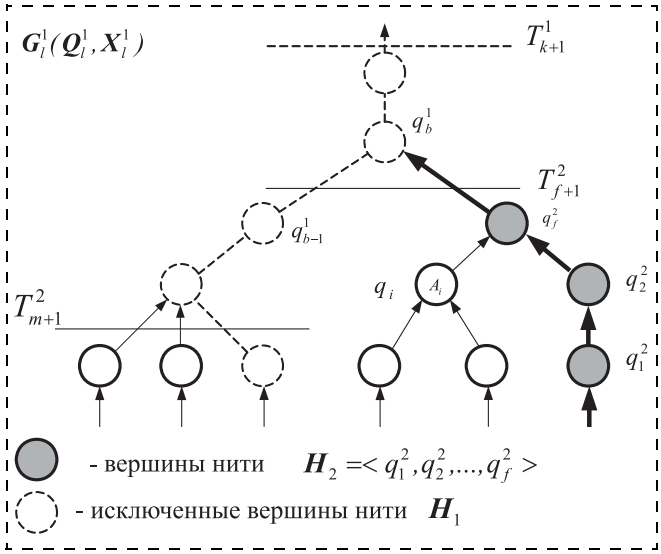

Рис. 4. Граф $\boldsymbol{G}^1_l(\boldsymbol{Q}^1_l, \boldsymbol{X}^1_l)$ задания Z_l, модифицированный объектом R_p

определяются исходя из следующих соображений. Допустим, что некоторая вершина q^2_f графа $\boldsymbol{G}^1_l(\boldsymbol{Q}^1_l, \boldsymbol{X}^1_l)$ инцидентна вершине q^1_b, принадлежащей нити $\boldsymbol{H}_1$ (рис. 4). Это означает, что для выполнения операции, приписанной вершине q^1_b, необходимы результаты выполнения операции, приписанной вершине q^2_f графа $\boldsymbol{G}^1_l(\boldsymbol{Q}^1_l, \boldsymbol{X}^1_l)$. Поэтому очевидно, что результаты выполнения операции, приписанной вершине q^2_f, должны быть получены и переданы объекту R_p, выполняющему операции нити $\boldsymbol{H}_1$, не позже, чем к требуемому моменту времени начала выполнения операции вершины q^1_b, определяемому согласно выражению (2) как

$$T^1_b = T^1_{k+1} - \left(\sum_{i=b}^{k} t_p(A_i) + t_{\text{п}} \right). \qquad (3)$$

В противном случае объект R_p не успеет закончить исполнение взятой на себя нити $\boldsymbol{H}_1$ к требуемому моменту времени $T^1_{k+1} = T^l_{\max}$.

Поэтому вершине q^2_f графа $\boldsymbol{G}^1_l(\boldsymbol{Q}^1_l, \boldsymbol{X}^1_l)$ приписывается требуемое время ее исполнения $T^2_{f+1} = T^1_b$, а также номер объекта R_p, которому результаты исполнения операции q^2_f должны быть переданы.

Аналогичным образом определяются требуемые моменты T^2_{m+1} исполнения всех остальных вер-

шин графа $\boldsymbol{G}_l^1(\boldsymbol{Q}_l^1, \boldsymbol{X}_l^1)$, инцидентных вершинам нити $\boldsymbol{H}_1$ (рис. 4).

Если после модификации новый граф $\boldsymbol{G}_l^1(\boldsymbol{Q}_l^1, \boldsymbol{X}_l^1)$ задания Z_l еще не пустой, т. е. $\boldsymbol{G}_l^1(\boldsymbol{Q}_l^1, \boldsymbol{X}_l^1) \neq \varnothing$, то процесс создания сообщества для выполнения задания Z_l продолжается далее. Допустим, что через какое-то время другой свободный объект R_c обнаруживает на ДО дескриптор задания Z_l. Объект R_p делает попытку войти в состав сообщества по исполнению данного задания. Для этого объект R_c выделяет в графе $\boldsymbol{G}_l^1(\boldsymbol{Q}_l^1, \boldsymbol{X}_l^1)$ (который остался в дескрипторе задания Z_l после модификации, проведенной объектом R_p) наиболее длинную нить $\boldsymbol{H}_2 = <q_1^2, q_2^2, ..., q_f^2>$ (рис. 4) конечной вершине q_f^2, которой приписано требуемое время исполнения T_{f+1}^2, и определяет согласно выражению (2) момент времени, когда необходимо приступить к ее выполнению, как

$$T_1^2 = T_{f+1}^2 - \left(\sum_{i=1}^{f} t_c(A_i) + t_{\text{п}} \right),$$

где T_{f+1}^2 — момент времени, приписанный конечной вершине q_f^2 нити $\boldsymbol{H}_2$.

Если $T_1^2 < T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, то это означает, что данная нить и соответственно все задание Z_l в целом не могут быть выполнены к требуемому моменту времени. Поэтому задание Z_l снимается с ДО, потребителю направляется сообщение о невозможности выполнения

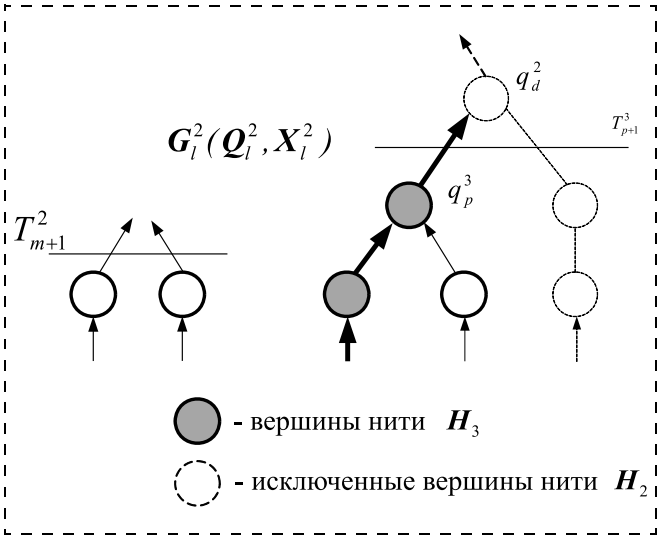

Рис. 5. Граф $\boldsymbol{G}_l^2(\boldsymbol{Q}_l^2, \boldsymbol{X}_l^2)$ задания Z_l, модифицированный объектом R_c

его задания Z_l, а всем объектам, вошедшим ранее в сообщество по его выполнению (номера которых хранятся в дескрипторе задания), сообщается об остановке данного задания, после чего они снова переходят в режим опроса ДО в поисках работы.

В случае же, если $T_1^2 \geqslant T_{\text{тек}}$, объект R_c принимает на себя исполнение операций, приписанных вершинам данной нити $\boldsymbol{H}_2$, и осуществляет очередную модификацию дескриптора задания Z_l на ДО:

- номер объекта R_c записывается в список членов сообщества по выполнению данного задания;
- вершины, входящие в нить $\boldsymbol{H}_2$, исключаются из графа $\boldsymbol{G}_l^1(\boldsymbol{Q}_l^1, \boldsymbol{X}_l^1)$, в результате чего образуется новый граф $\boldsymbol{G}_l^2(\boldsymbol{Q}_l^2, \boldsymbol{X}_l^2)$ (рис. 5);
- вершинам q_p^3 графа $\boldsymbol{G}_l^2(\boldsymbol{Q}_l^2, \boldsymbol{X}_l^2)$, инцидентным вершинам q_d^2 нити $\boldsymbol{H}_2$, приписывается требуемое время их исполнения (рис. 5), определяемое как

$$T_{p+1}^3 = T_{f+1}^2 - \left(\sum_{i=d}^{f} t_c(A_i) + t_{\text{п}} \right),$$

а также номер объекта R_c, которому результаты исполнения этих операций должны быть направлены.

Далее в процесс распределения операций задания Z_l включается следующий свободный объект множества $\boldsymbol{R}$, обнаруживший его дескриптор на ДО, и так до тех пор, пока не окажется, что граф $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ стал пустым, что означает, что все операции задания Z_l разобраны объектами, вошедшими в сообщество по его выполнению.

После того как некоторый объект R_p выбрал для исполнения нить $\boldsymbol{H}_f = <q_1^f, q_2^f, ..., q_k^f>$, он приступает к реализации операций, приписанных ее вершинам. При этом перед началом выполнения очередной операции A_i, приписанной вершине $q_i^f \in \boldsymbol{H}_f$ ($i = 1, 2, ..., k$), он должен проверить соблюдение временного графика выполнения последовательности операций нити $\boldsymbol{H}_f$. Для этого он сравнивает требуемое время начала выполнения операции, приписанной вершине $q_d^f \in \boldsymbol{H}_f$, определяемое как

$$T_d^f = T_{k+1}^f - \left(\sum_{i=d}^{k} t_p(A_i) + t_{\text{п}} \right),$$

где T_{k+1}^f — время, приписанное последней вершине q_k^f нити $\boldsymbol{H}_f$, с текущим временем $T_{\text{тек}}$.

Если $T_j^f < T_{\text{тек}}$, то это означает, что объект R_p уже не успевает выполнить оставшиеся операции

нити $\boldsymbol{H}_f$ к требуемому моменту времени T^f_{k+1}, т. е. он по каким-то причинам не выполнил в запланированное время предыдущие операции нити $\boldsymbol{H}_f$ и, тем самым, "отстал" от графика выполнения нити $\boldsymbol{H}_f$, что, в свою очередь, ведет к "срыву" графика выполнения всего задания Z_l в целом. В этом случае задание Z_l снимается с ДО, потребителю направляется сообщение о невозможности его выполнения к заданному моменту времени, а всем объектам, вошедшим в сообщество по его выполнению, направляется сообщение об остановке данного задания, после чего они переходят в режим поиска задания на ДО.

Описанному выше процессу отвечает следующий укрупненный алгоритм функционирования устройства управления отдельного объекта, входящего в состав распределенной системы $\boldsymbol{R}$.

Алгоритм 1.

1. Свободный объект R_p опрашивает ДО.

2. При обнаружении на ДО задания Z_l объект R_p считывает его дескриптор и анализирует граф задания $\boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l)$. Если $\boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l) = \varnothing$, то переход к п. 1, иначе

3. Объект R_p выделяет в графе $\boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l)$ наиболее длинную нить $\boldsymbol{H}_j = <q^j_1, q^j_2, ..., q^j_k>$ согласно выражению (1), конечной вершине q^j_l которой приписано требуемое время ее исполнения T^j_{k+1} (если $\boldsymbol{H}_j = \boldsymbol{H}_1$, то $T^j_{k+1} = T^l_{\max}$), и определяет допустимый момент времени, когда необходимо начать ее выполнение, как

$$T^j_1 = T^j_{k+1} - \left(\sum_{i=1}^{k} t_p(A_i) + t_{\text{п}} \right).$$

4. Если $T^j_1 < T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, то переход к 12, иначе

5. Объект R_p принимает на себя выполнение нити $\boldsymbol{H}_j$, для чего модифицирует дескриптор задания Z_l на ДО: записывает в список участников сообщества $\boldsymbol{R}_l$ свой номер; модифицирует граф $\boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l)$ задания Z_l путем исключения из него вершин нити $\boldsymbol{H}_j$, т. е. $\boldsymbol{G}^{j+1}_l(\boldsymbol{Q}^{j+1}_l, \boldsymbol{X}^{j+1}_l) = \boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l)/\boldsymbol{H}_j$; приписывает вершинам q^{j+1}_b модифицированного графа $\boldsymbol{G}^{j+1}_l(\boldsymbol{Q}^{j+1}_l, \boldsymbol{X}^{j+1}_l)$, инцидентным вершинам q^j_d нити $\boldsymbol{H}_j$, требуемое время исполнения, определяемое как

$$T^{j+1}_{b+1} = T^j_{k+1} - \left(\sum_{i=d}^{k} t_p(A_i) + t_{\text{п}} \right), \qquad (4)$$

а также номер объекта R_p, которому необходимо передать результаты исполнения вершины q^{i+1}_b.

6. Объект R_p переходит к исполнению последовательности операций, приписанных вершинам нити $\boldsymbol{H}_j = <q^j_1, q^j_2, ..., q^j_k>$; $d = 1$.

7. Если $T^j_d < T_{\text{тек}}$, где $T^j_d = T^j_{k+1} - \left(\sum_{i=d}^{k} t_p(A_i) + t_{\text{п}} \right)$ — требуемое время начала выполнения операции A_d, приписанной вершине $q^j_d \in \boldsymbol{H}_j$, то перейти к 12, иначе

8. Объект R_p выполняет операцию A_d, приписанную вершине $q^f_d \in \boldsymbol{H}_j$.

9. Если объекту R_p поступило сообщение о прекращении выполнения задания Z_l, то переход к 1, иначе

10. $d = d + 1$, если $d \leqslant k$, то переход к 7, иначе

11. После выполнения всех операций нити $\boldsymbol{H}_j$ номер объекта R_p исключается из списка членов сообщества по выполнению задания Z_l в его дескрипторе на ДО, а результат выполнения последней операции нити $\boldsymbol{H}_j$ передается объекту R_c, номер которого приписан последней вершине q^j_k нити $\boldsymbol{H}_j$, или потребителю, если последняя вершина выполненной нити $\boldsymbol{H}_j$ — это конечная вершина графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, т. е. $q^j_k = q_k$. При этом дескриптор задания Z_l удаляется с ДО. Переход к 1.

12. Задание Z_l не может быть выполнено. Дескриптор задания Z_l удаляется с ДО, потребителю направляется сообщение о невозможности выполнения его задания, а всем объектам, номера которых записаны в списке участников сообщества по выполнению задания Z_l, передается сообщение о прекращении его исполнения. Переход к 1.

Вариант 2

В отличие от первого варианта постановки задачи распределения операций потока заданий Z_l между объектами распределенной системы $\boldsymbol{R}$, во втором варианте время выполнения идентичных операций различными объектами системы $\boldsymbol{R}$ также различно, т. е.

$$t_i(A_l) \neq t_j(A_l)$$
$$(i = 1, 2, ..., N, j = 1, 2, ..., i - 1, i + 1, ..., N).$$

При этом очевидно, что время выполнения одной и той же нити $\boldsymbol{H}_f$ различными объектами множества $\boldsymbol{R}$ также будет различным. Поэтому, если

окажется, что объект $R_p \in \boldsymbol{R}$, первым обнаруживший задание Z_l на ДО, не может выполнить $\boldsymbol{H}_f$ к заданному моменту времени $T^l_{\max}$, то это вовсе не означает, что задание невыполнимо. Действительно, через какое-то время может появиться другой свободный объект $R_c \in \boldsymbol{R}$, который сможет выполнить данную нить к требуемому моменту времени.

Поэтому после обнаружения задания Z_l на ДО объект R_p должен сначала согласно выражению (1) выделить наиболее длинную нить, конечной вершине которой приписано требуемое время выполнения, и оценить возможность ее исполнения к заданному моменту времени. Если это условие выполняется, то он принимает исполнение последовательности операций, приписанных данной нити, на себя. В противном случае он должен выделить в графе заданий следующую по длине нить, конечной вершине которой также приписано требуемое время ее исполнения, и проанализировать возможность ее исполнения к требуемому моменту времени и т. д. до тех пор, пока он не найдет нить, которую он может выполнить за отведенное время, либо безуспешно переберет все нити графа задания в порядке убывания их длины.

Следует отметить, что в момент размещения задания Z_l на ДО требуемый момент времени исполнения $T^l_{\max}$ приписан только конечной вершине q_k графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ (см. рис. 3). Поэтому процесс распределения операций этого задания не "сдвинется с места" до тех пор, пока не найдется объект, способный выполнить нить $\boldsymbol{H}_1 = <q^1_1, q^1_2, ..., q^1_k>$, заканчивающуюся на конечной вершине q_k, к моменту времени $T^1_{k+1} = T^l_{\max}$.

После того как нить $\boldsymbol{H}_1$ будет принята некоторым объектом R_p к исполнению и всем вершинам модифицированного графа $\boldsymbol{G}^1_l(\boldsymbol{Q}^1_l, \boldsymbol{X}^1_l)$, инцидентным вершинам нити $\boldsymbol{H}_1$, будут приписаны требуемые моменты времени их исполнения (с помощью описанной выше процедуры), последующие объекты, обнаружившие задание на ДО, будут иметь уже возможность выбора среди некоторого множества тех нитей, для которых требуемое время их исполнения определено.

Процесс распределения задания Z_l должен заканчиваться, когда либо все нити графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ разобраны объектами, вступившими в сообщество $\boldsymbol{R}_l$ по его исполнению, т. е. $\boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l) = \varnothing$, либо текущее время $T_{\text{тек}}$ превысит заданное время $T^l_{\max}$ исполнения всего задания Z_l в целом.

Как только объект R_p принял к исполнению некоторую нить $\boldsymbol{H}_f$, он приступает к выполнению последовательности операций, приписанных ее вершинам. При этом прежде чем приступить к выполнению операции A_d, приписанной очередной вершине $q^f_d \in \boldsymbol{H}_f$, он должен проверить соблюдение временного графика выполнения последовательности операций данной нити, т. е. сравнить текущее время $T_{\text{тек}}$ с требуемым временем начала выполнения операции A_d, определяемым как

$$T^f_d = T^f_{k+1} - \left(\sum_{i=d}^{k} t_p(A_i) + t_{\text{п}} \right).$$

Если $T^f_d < T_{\text{тек}}$, то это означает, что план выполнения задания Z_l "сорван", после чего дальнейшее выполнение задания Z_l прекращается. В противном случае объект R_p приступает к выполнению операции A_d.

Исходя из приведенных выше соображений можно предложить следующий алгоритм функционирования отдельного объекта распределенной системы $\boldsymbol{R}$ в условиях, когда время выполнения идентичных операций различными объектами также различно.

Алгоритм 2.

1. Свободный объект R_p опрашивает ДО.

2. При обнаружении на ДО задания Z_l объект R_p считывает его дескриптор и анализирует граф $\boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l)$. Если $\boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l) = \varnothing$, то переход к 1, иначе

3. Если $T_{\text{тек}} \geqslant T^l_{\max}$, где $T_{\text{тек}}$ — текущий момент времени, то переход к 17, иначе

4. $i = 1$; $\boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i) = \boldsymbol{G}^j_l(\boldsymbol{Q}^j_l, \boldsymbol{X}^j_l)$.

5. Объект R_p выделяет в графе $\boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i)$ наиболее длинную нить $\boldsymbol{H}_i = <q^i_1, q^i_2, ..., q^i_k>$ согласно выражению (1), для конечной вершины q^i_l которой определено требуемое время исполнения T^i_{k+1} (если $\boldsymbol{H}_i = \boldsymbol{H}_1$, то $T^i_{k+1} = T^l_{\max}$), и определяет момент времени, когда он должен приступить к ее выполнению, как

$$T^i_1 = T^i_{k+1} - \left(\sum_{m=d}^{k} t_p(A_m) + t_{\text{п}} \right).$$

6. Если $T^i_1 \geqslant T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, то переход к 10, иначе

7. Нить $\boldsymbol{H}_i$ исключается из графа $\boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i)$, т. е.

$$\boldsymbol{G}_{i+1}(\boldsymbol{Q}_{i+1}, \boldsymbol{X}_{i+1}) = \boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i)/\boldsymbol{H}_i.$$

8. Если $\boldsymbol{G}_{i+1}(\boldsymbol{Q}_{i+1}, \boldsymbol{X}_{i+1}) = \varnothing$, то перейти к 1, иначе

9. $i = i + 1$, перейти к 5.

10. Объект R_p принимает на себя выполнение нити $\boldsymbol{H}_i$, для чего модифицирует дескриптор задания Z_l: записывает в список участников сообщества $\boldsymbol{R}_l$ по

выполнению задания Z_l свой номер; модифицирует граф задания Z_l путем исключения из него вершин нити $\boldsymbol{H}_i$, т. е. $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1}) = \boldsymbol{G}_l^{j}(\boldsymbol{Q}_l^{j}, \boldsymbol{X}_l^{j})/\boldsymbol{H}_i$; приписывает вершинам q_b^{j+1} модифицированного графа $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1})$, инцидентным вершинам q_d^i нити $\boldsymbol{H}_i$, требуемое время их исполнения, определяемое как

$$T_{b+1}^{j+1} = T_{k+1}^{i} - \left(\sum_{m=d}^{k} t_p(A_m) + t_{\text{п}} \right),$$

а также номер объекта R_p, которому необходимо передать результаты исполнения вершины q_b^{j+1}.

11. Объект R_p переходит к исполнению последовательности операций, приписанных вершинам принятой к исполнению нити $\boldsymbol{H}_i = <q_1^i, q_2^i, ..., q_k^i>$; $d = 1$.

12. Если $T_d^i < T_{\text{тек}}$, где $T_d^i = T_{k+1}^i - \left(\sum_{m=d}^{k} t_p(A_m) + t_{\text{п}} \right)$ — требуемое время начала выполнения операции A_d, приписанной вершине $q_d^i \in \boldsymbol{H}_i$, то перейти к 17, иначе

13. Объект R_p выполняет операцию A_d, приписанную вершине $q_d^i \in \boldsymbol{H}_i$.

14. Если объекту R_p поступило сообщение о прекращении дальнейшего выполнения задания Z_l, то переход к 1, иначе

15. $d = d + 1$, если $d \leqslant k$, то перейти к 12, иначе

16. После выполнения всех операций нити $\boldsymbol{H}_i$ номер объекта R_p исключается из списка членов сообщества по выполнению задания Z_l в его дескрипторе на ДО, а результат выполнения последней операции нити $\boldsymbol{H}_i$ передается объекту R_c, номер которого приписан конечной вершине q_k^i нити $\boldsymbol{H}_i$, или потребителю, если последняя вершина выполненной нити $\boldsymbol{H}_i$ — это конечная вершина графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, т. е. $q_k^i = q_k$. В последнем случае дескриптор задания Z_l удаляется с ДО. Переход к 1.

17. Задание Z_l не может быть выполнено. Дескриптор задания Z_l удаляется с ДО, потребителю направляется сообщение о невозможности выполнения его задания, а всем объектам, номера которых записаны в списке участников сообщества по выполнению задания Z_l, сообщается о прекращении его исполнения. Переход к 1.

Вариант 3

В этом варианте исходной постановки задачи предполагается, что объекты R_i ($i = 1, 2, ..., N$) системы $\boldsymbol{R}$ выполняют различные наборы операций $\boldsymbol{A}_i$, причем в общем случае $\boldsymbol{A}_i \cap \boldsymbol{A}_j \neq \varnothing$ ($i = 1, 2, ..., N$, $j = 1, 2, ..., i-1, i+1, ..., N$), однако время выполнения идентичных операций различными объектами одинаково.

Рассмотрим основные принципы организации процесса распределения (сетевого планирования) операций потока заданий $\boldsymbol{Z}$ среди множества объектов $\boldsymbol{R}$ при данных условиях.

Допустим, что некоторый свободный объект $R_p \in \boldsymbol{R}$ обнаружил на ДО задание Z_l. В отличие от предыдущих вариантов исходной постановки в данном случае может оказаться, что объект R_p способен выполнять далеко не все операции, приписанные вершинам графа $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ задания Z_l, хранимого на ДО. Поэтому в графе $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ необходимо предварительно выделить подграф $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$, вершинам которого приписаны операции множества $\boldsymbol{A}_p$, выполняемые объектом R_p (рис. 6). После этого необходимо проанализировать, есть ли среди вершин графа $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ вершины, которым приписано требуемое время их исполнения (при размещении задания Z_l на ДО требуемое время исполнения $T_{\max}^l$ приписано только конечной вершине q_k графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$). Если таковых

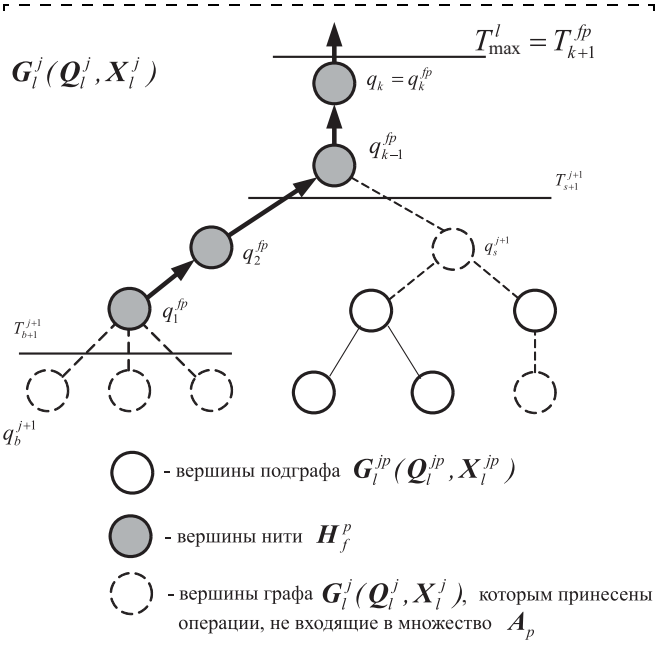

Рис. 6. Выделение подграфа $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ в графе $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ задания Z_l

вершин нет, то это говорит о том, что объект R_p пока что не может вступить в сообщество по выполнению задания Z_l, и поэтому он вновь переходит к режиму опроса ДО в целях поиска других заданий. В противном случае в графе $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ с помощью выражения (1) выделяется наиболее длинная нить $\boldsymbol{H}_f^p = <q_1^{fp}, q_2^{fp}, ..., q_k^{fp}>$, конечной вершине q_k^{fp} которой приписано требуемое время исполнения T_{k+1}^f, и анализируется возможность ее исполнения объектом R_p к данному моменту времени (рис. 6). Для этого определяется требуемое время начала исполнения операции, приписанной первой вершине q_1^{fp} данной нити $\boldsymbol{H}_f^p$, как

$$T_1^{fp} = T_{k+1}^f - \left(\sum_{i=1}^{k} t_p(A_i) + t_{\text{п}} \right),$$

после чего оно сравнивается с текущим временем $T_{\text{тек}}$. Если $T_{\text{тек}} > T_1^{fp}$, то это означает, что объект R_p не может выполнить данную нить к требуемому моменту времени T_{k+1}^f. Поскольку в данном варианте принято, что все объекты множества $\boldsymbol{R}$ выполняют идентичные операции за одинаковое время, то это также означает, что никакой другой объект $R_c \in \boldsymbol{R}$ также не может выполнить данную нить $\boldsymbol{H}_f^p$ к требуемому моменту времени, т. е. задание Z_l невыполнимо, и его дескриптор снимается с ДО. Иначе объект R_p входит в состав сообщества по выполнению задания Z_l, нить $\boldsymbol{H}_f^p$ исключается из графа $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$, в результате чего формируется новый граф $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1}) = \boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)/\boldsymbol{H}_f^p$, а вершинам графа $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1})$, инцидентным вершинам нити $\boldsymbol{H}_f^p$, приписывается требуемое время их исполнения, определяемое согласно выражению (3), а также номер объекта R_p, которому должны быть переданы результаты исполнения приписанных им операций.

После этого объект R_p может приступать к выполнению операций, приписанных вершинам нити $\boldsymbol{H}_f^p$. Однако поскольку поиск нити $\boldsymbol{H}_f^p$ осуществлялся в подграфе $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ графа $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$, то может оказаться, что операция, приписанная первой вершине q_1^{fp} нити $\boldsymbol{H}_f^p$, не может быть выполнена, поскольку она не является начальной вершиной графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$ задания Z_l, а для ее выполнения необходимы результаты операции смежной вершины q_b^{j+1} графа $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1})$ (рис. 6). В этом случае объект R_p должен перейти в режим ожидания получения результатов выполнения операции, приписанной вершине q_b^{j+1}, и только после этого приступать к выполнению операции нити $\boldsymbol{H}_f^p$.

Процесс распределения операций задания Z_l должен заканчиваться в случае, если граф задания $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1})$ становится пустым (что означает, что все его операции распределены между объектами множества $\boldsymbol{R}$) либо если текущее время $T_{\text{тек}}$ превышает установленное потребителем время $T_{\max}^l$ выполнения всего задания Z_l.

Исходя из приведенных выше соображений можно предложить следующий алгоритм работы устройства управления объекта R_p, входящего в состав системы $\boldsymbol{R}$ в условиях, когда наборы операций, выполняемых различными объектами, также различны, однако время выполнения ими идентичных операций одинаково.

Алгоритм 3.

1. Свободный объект R_p опрашивает ДО.

2. При обнаружении на ДО задания Z_i объект R_p считывает его дескриптор и анализирует граф задания $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$. Если $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j) = \varnothing$, то переход к 1, иначе 3.

В графе $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ выделяется подграф $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$, вершинам которого приписаны операции множества $\boldsymbol{A}_p$, выполняемые объектом R_p.

4. Если $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp}) = \varnothing$ или ни одной из вершин графа $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ не приписано требуемое время T_{k+1}^j ее исполнения, то перейти к 1, иначе

5. Объект R_p выделяет в графе $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ наиболее длинную нить $\boldsymbol{H}_j^p = <q_1^{jp}, q_2^{jp}, ..., q_k^{jp}>$, конечной вершине которой приписано требуемое время исполнения T_{k+1}^j (в момент размещения задания Z_i на ДО требуемое время $T_{k+1}^j = T_{\max}^l$ приписано только конечной вершине q_k графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$), и определяет допустимый момент времени, когда необходимо начать ее выполнение, как

$$T_1^{jp} = T_{k+1}^j - \left(\sum_{i=1}^{k} t_p(A_i) + t_{\text{п}} \right),$$

где $t_p(A_i)$ — время, затрачиваемое объектом R_p на выполнение операции A_i, приписанной вершине q_i^{jp} ($i = 1, 2, ..., k$) нити $\boldsymbol{H}_j^p$.

6. Если $T_1^{jp} < T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, то переход к 15, иначе

7. Объект R_p принимает на себя исполнение нити $\boldsymbol{H}_j^p$, для чего модифицирует дескриптор задания Z_i на ДО: в список участников сообщества записывается его номер p; из графа $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ исключаются вершины нити $\boldsymbol{H}_j^p$, в результате чего формируется новый граф $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1}) = \boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)/\boldsymbol{H}_j^p$; вершинам q_b^{j+1} модифицированного графа $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1})$, инцидентным вершинам q_b^{jp} нити $\boldsymbol{H}_j^p$, приписывается требуемое время их исполнения, определяемое как (рис. 6)

$$T_{b+1}^{j+1} = T_{k+1}^j - \left(\sum_{i=d}^{k} t_p(A_i) + t_{\text{п}} \right),$$

а также номер объекта R_p, которому необходимо передать результаты исполнения операции, приписанной вершине q_b^{j+1}.

8. Если первая вершина q_1^{jp} нити $\boldsymbol{H}_j^p = <q_1^{jp}, q_2^{jp}, ..., q_k^{jp}>$ — это начальная вершина графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, то объект R_p приступает к исполнению операций, приписанных вершинам этой нити. В противном случае объект R_p ожидает поступления результатов выполнения операции, приписанной вершине q_b^{j+1}, инцидентной вершине q_1^{jp} (рис. 6).

9. $d = 1$.

10. Если $T_d^{jp} < T_{\text{тек}}$, где $T_d^{jp} = T_{k+1}^{jp} - \left(\sum_{i=d}^{k} t_p(A_i) + t_{\text{п}} \right)$ — требуемое время начала выполнения операции A_d, приписанной вершине $q_b^{jp} \in \boldsymbol{H}_j^p$, то перейти к 15, иначе

11. Объект R_p выполняет операцию A_d, приписанную вершине $q_b^{jp} \in \boldsymbol{H}_j^p$.

12. Если объекту R_p поступило сообщение о прекращении выполнения задания Z_l, то переход к 1, иначе

13. $d = d + 1$, если $d \leqslant k$, то переход к 10, иначе

14. После выполнения всех операций нити $\boldsymbol{H}_j^p$ номер объекта R_p исключается из списка членов сообщества $\boldsymbol{R}_l$ по выполнению задания Z_l в его дескрипторе на ДО, а результат выполнения последней операции нити $\boldsymbol{H}_j^p$ передается объекту R_c, номер которого приписан последней вершине q_k^{jp} нити $\boldsymbol{H}_j^p$, или потребителю, если последняя вершина выполненной нити $\boldsymbol{H}_j^p$ — это конечная вершина графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, т. е. $q_k^{jp} = q_k$. При этом дескриптор задания Z_l удаляется с ДО. Переход к 1.

15. Задание Z не может быть выполнено. Дескриптор задания Z_l удаляется с ДО, потребителю направляется сообщение о невозможности выполнения его задания, а всем объектам, номера которых записаны в списке участников сообщества по выполнению задания Z_l, передается сообщение о прекращении его исполнения. Переход к 1.

Вариант 4

В самом сложном варианте исходной постановки объекты множества $\boldsymbol{R} = <R_1, R_2, ..., R_N>$ выполняют различные наборы операции $\boldsymbol{A}_i$, причем время выполнения идентичных операций различными объектами также различно.

Здесь, как и в предыдущем варианте, объект R_p, обнаруживший на ДО дескриптор задания Z_l, должен первым делом выделить в графе $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ подграф $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$, вершинам которого приписаны операции, входящие в множество $\boldsymbol{A}_p$. Далее в графе $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ объект R_p выделяет наиболее длинную нить $\boldsymbol{H}_f^p$, конечной вершине q_k^{jp} которой приписано требуемое время исполнения T_{k+1}^f, и анализирует возможность ее исполнения к данному моменту времени. Для этого он определяет время начала исполнения операции, приписанной первой вершине данной нити $\boldsymbol{H}_f^p$, как $T_1^{fp} = T_{k+1}^f - \left(\sum_{i=1}^{k} t_p(A_i) + t_{\text{п}} \right)$, и сравнивает его с текущим временем $T_{\text{тек}}$. Если $T_{\text{тек}} > T_1^{fp}$, то это означает, что объект R_p не может выполнить данную нить к требуемому моменту времени T_{k+1}^f. Поскольку в данном варианте принято, что объекты множества $\boldsymbol{R}$ выполняют идентичные операции за различное время, то в дальнейшем может появиться объект R_c,

который сможет выполнить нить $\boldsymbol{H}_f^p$ к моменту времени T_{k+1}^f. Поэтому, в отличие от варианта 3, в данном случае задание Z_l не снимается с ДО, а объект R_p продолжает анализировать следующие по убыванию длины нити графа $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ до тех пор, пока он не найдет нить, которую он может выполнить к требуемому моменту времени, либо пока не переберет все возможные нити графа $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$.

В остальном процедура распределения не отличается от той, которая описана в варианте 3.

Исходя из этих соображений алгоритм работы объекта R_p системы $\boldsymbol{R}$ при выполнении потока заданий в условиях, когда объекты выполняют различные наборы операций за различное время, можно представить в следующем виде.

Алгоритм 4.

1. Свободный объект R_p опрашивает ДО.

2. При обнаружении на ДО задания Z_i объект R_p считывает его дескриптор и анализирует граф задания $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$. Если $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j) = \varnothing$, то переход к 1, иначе

3. Если $T_{\text{тек}} \geqslant T_{\max}^l$, где $T_{\text{тек}}$ — текущий момент времени, то переход к 20, иначе

4. В графе $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ выделяется подграф $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$, вершинам которого приписаны операции множества $\boldsymbol{A}_p$, выполняемые объектом R_p.

5. Если $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp}) = \varnothing$ или ни одной из вершин графа $\boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$ не приписано требуемое время T_{k+1}^j ее исполнения, то перейти к 1, иначе

6. $i = 1$; $\boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i) = \boldsymbol{G}_l^{jp}(\boldsymbol{Q}_l^{jp}, \boldsymbol{X}_l^{jp})$.

7. Объект R_p выделяет в графе $\boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i)$ наиболее длинную нить $\boldsymbol{H}_i^p = <q_1^{ip}, q_2^{ip}, ..., q_k^{ip}>$, для конечной вершины q_k^{ip} которой определено требуемое время исполнения T_{k+1}^i (если $\boldsymbol{H}_i = \boldsymbol{H}_1$, то $T_{k+1}^i = T_{\max}^l$), и определяет момент времени, когда он должен приступить к ее выполнению, как

$$T_1^{ip} = T_{k+1}^i - \left(\sum_{m=1}^{k} t_p(A_m) + t_{\text{п}} \right).$$

8. Если $T_1^{ip} \geqslant T_{\text{тек}}$, где $T_{\text{тек}}$ — текущий момент времени, то переход к 12, иначе

9. Нить $\boldsymbol{H}_i^p$ исключается из графа $\boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i)$, т. е.

$$\boldsymbol{G}_{i+1}(\boldsymbol{Q}_{i+1}, \boldsymbol{X}_{i+1}) = \boldsymbol{G}_i(\boldsymbol{Q}_i, \boldsymbol{X}_i) / \boldsymbol{H}_i^p.$$

10. Если $\boldsymbol{G}_{i+1}(\boldsymbol{Q}_{i+1}, \boldsymbol{X}_{i+1}) = \varnothing$, то перейти к 1, иначе

11. $i = i + 1$, перейти к 7.

12. Объект R_p принимает на себя исполнение нити $\boldsymbol{H}_i^p$, для чего модифицирует дескриптор задания Z_i на ДО: в список участников сообщества записывается его номер p; из графа $\boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j)$ исключаются вершины нити $\boldsymbol{H}_i^p$, в результате чего формируется новый граф $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1}) = \boldsymbol{G}_l^j(\boldsymbol{Q}_l^j, \boldsymbol{X}_l^j) / \boldsymbol{H}_i^p$; вершинам q_b^{j+1} модифицированного графа $\boldsymbol{G}_l^{j+1}(\boldsymbol{Q}_l^{j+1}, \boldsymbol{X}_l^{j+1})$, инцидентным вершинам q_d^{ip} нити $\boldsymbol{H}_i^p$, приписывается требуемое время их исполнения, определяемое как (рис. 6)

$$T_{b+1}^{j+1} = T_{k+1}^i - \left(\sum_{m=d}^{k} t_p(A_m) + t_{\text{п}} \right),$$

а также номер объекта R_p, которому необходимо передать результаты исполнения операции, приписанной вершине q_b^{j+1}.

13. Если первая вершина q_1^{ip} нити $\boldsymbol{H}_i^p = <q_1^{ip}, q_2^{ip}, ..., q_k^{ip}>$ — это начальная вершина графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, то объект R_p приступает к исполнению операций, приписанных вершинам этой нити. В противном случае объект R_p ожидает поступления результатов выполнения операции, приписанной вершине q_b^{j+1}, инцидентной вершине q_1^{ip} (рис. 6).

14. $d = 1$.

15. Если $T_d^{ip} < T_{\text{тек}}$, где $T_d^{ip} = T_{k+1}^i - \left(\sum_{m=d}^{k} t_p(A_m) + t_{\text{п}} \right)$ — требуемое время начала выполнения операций A_d, приписанной вершине $q_d^{ip} \in \boldsymbol{H}_i^p$, то перейти к 15, иначе

16. Объект R_p выполняет операцию A_d, приписанную вершине $q_d^{ip} \in \boldsymbol{H}_i^p$.

17. Если объекту R_p поступило сообщение о прекращении выполнения задания Z_l, то переход к 1, иначе

18. $d = d + 1$, если $d \leqslant k$, то переход к 15, иначе

19. После выполнения всех операций нити $\boldsymbol{H}_i^p$ номер объекта R_p исключается из списка членов сообщества по выполнению задания Z_l в его дескрипторе на ДО, а результат выполнения последней операции нити $\boldsymbol{H}_i^p$ передается объекту R_c, номер которого приписан последней вершине q_k^{ip} нити $\boldsymbol{H}_i^p$, или потребителю, если последняя вершина выполненной нити $\boldsymbol{H}_i^p$ — это конечная вершина графа $\boldsymbol{G}_l(\boldsymbol{Q}_l, \boldsymbol{X}_l)$, т. е. $q_k^{ip} = q_k$. При этом дескриптор задания Z_l удаляется с ДО. Переход к 1.

20. Задание Z не может быть выполнено. Дескриптор задания Z_l удаляется с ДО, потребителю направляется сообщение о невозможности выполнения его задания, а всем объектам, номера которых записаны в списке участников сообщества по выполнению задания Z_l, передается сообщение о прекращении его исполнения. Переход к 1.

Результаты экспериментальных исследований

В целях исследования работоспособности и эффективности предложенного децентрализованного метода управления объектами распределенной системы $\boldsymbol{R}$ при выполнении потока заданий $\boldsymbol{Z}$ была разработана программная модель. Программная модель обеспечивает возможность моделирования работы распределенной системы $\boldsymbol{R}$ при различных значениях таких исходных параметров, как

- число объектов в системе (до 1000 шт.);
- число различных операций, выполняемых объектами, входящими в систему (до 20);
- время выполнения отдельных операций различными объектами системы;
- сложность (число операций в графе) заданий и частота их появления;
- требуемое время выполнения потребительских заданий.

При этом в качестве критериев эффективности работы распределенной системы при выполнении потока заданий были приняты:

- коэффициент полезного действия (КПД) — отношение времени, затраченного объектами системы на выполнение потребительских заданий, к общему времени их работы в системе;
- коэффициент гарантированности выполнения (КГВ) потребительского задания — отношение числа потребительских заданий, выполненных к требуемому моменту времени, к общему числу заданий, направленных потребителями на ДО.

Результаты серии экспериментов, проведенных при различных значениях исходных параметров программной модели, показали, что значение КПД не опускается ниже 75 %, а его среднее значение составляет 84 %; значение КГВ не опускается ниже 91 %, а среднее значение составляет 97 %.

Заключение

В статье описаны общие принципы организации децентрализованного управления распределенной системой, состоящей из множества независимых объектов, объединенных сетевым каналом связи, при выполнении потока потребительских заданий, поступающих в заранее неизвестные моменты времени. Реализация предлагаемого подхода позволяет:

- обеспечить квазиоптимальное автоматическое распределение операций различных заданий между объектами распределенной системы;
- обеспечить высокую полезную загрузку объектов системы при выполнении потока потребительских заданий;
- обеспечить высокую вероятность выполнения потребительских заданий в установленное время;
- обеспечить возможность неограниченного наращивания (масштабируемости) числа объектов в системе;
- обеспечить повышенную отказоустойчивость распределенной системы, поскольку выход из строя любого из них не приводит к катастрофическим последствиям для системы в целом.

Список литературы

1. **Юревич Е. И.** О проблеме группового управления роботами // Мехатроника, автоматизация, управление. 2004. № 2. С. 9—13.
2. **Каляев И. А., Мельник Э. В.** Децентрализованные системы компьютерного управления. Ростов-на-Дону: ЮНЦ РАН, 2011. 196 с.
3. **Интеллектуальные** роботы / Под редакцией Е. И. Юревича. М.: Машиностроение, 2007. 360 с.
4. **Каляев И. А.** Метод коллективного управления группой объектов // Мехатроника, автоматизация, управление. 2004. № 3. С. 9—15.
5. **Каляев И. А., Гайдук А. Р., Капустян С. Г.** Модели и алгоритмы коллективного управления в группах роботов. М.: Физматлит, 2009. 278 с.
6. **Зуховицкий С. И., Радчик И. А.** Математические методы сетевого планирования. М.: Наука, 1965. 360 с.
7. **Кофман А., Дебезей Г.** Сетевые методы планирования. М.: Прогресс, 1968. 182 с.
8. **Новиков Д. А.** Управление проектами. Организационные механизмы. М.: ПМСОФТ, 2007. 140 с.
9. **Рейнгольд Э., Нивергельт Ю., Део Н.** Комбинаторные алгоритмы. Теория и практика. М.: Мир, 1980. 476 с.

Method of Decentralized Control of the Distributed System during Execution of the Task Flow

A. I. Kalyaev, anatoly@kalyaev.net, **I. A. Kalyaev,** kaliaev@mvs.sfedu.ru✉,
Scientific Research Institute of Multiprocessor Computer Systems named after Academician A. V. Kalyaev,
Taganrog, 347928, Russian Federation

Corresponding author: **Kalyaev Igor A.,** D. Sc., Director,
Scientific Research Institute of Multiprocessor Computer Systems
named after Academician A. V. Kalyaev,
Taganrog, 347928, Russian Federation, e-mail: kaliaev@mvs.sfedu.ru

Received on April 16, 2015
Accepted on May 13, 2015

This paper is devoted to the distributed control system with the network architecture consisting of a multitude of objects united by a communication channel and participating in implementation of the flow of the incoming consumer tasks. At that, it is assumed that each consumer task consists of a set of interconnected operations, presented as an acyclic graph, and can appear any moment. In the paper the authors show that a centralized management solution with a single control unit in such a distributed system with a large number of objects is very complicated. Therefore, they propose a method for a decentralized management of the distributed systems using multiple management devices for the individual objects. They propose new algorithms for individual management of the objects of a distributed system for four versions of the original statement: the objects can perform the same set of operations at the same time; the objects perform the same set of operations, but the times of execution of the identical operations in various objects are different; the objects perform different sets of operations, but the times of execution of the identical operations are equal; the objects perform different sets of operations and the times of execution of the identical operations in different objects are different. In conclusion the authors present the results of the experimental researches of the proposed algorithms achieved due to application of the program model of the distributed system.

Keywords: *distributed system, flow of tasks, decentralized management, operations distribution*

For citation:

Kalyaev A. I., Kalyaev I. A. Method of Decentralized Control of the Distributed System during Execution of the Task Flow, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2015, vol. 16, no. 9, pp. 585—598.

DOI: 10.17587/mau.16.585-598

References

1. **Jurevich E. I.** *O problem groupovogo upravlenija robotami* (About the problem of group control robots), *Mehatronika, Avtomatizacija, Upravlenie,* 2004, 2, pp. 9—13.

2. **Kalyaev I. A., Melnik E. V.** *Decentralizovannye sistemy komp'juternogo upravlenija* (Distributed computer control systems), Rostov-na-Donu, JuNC RAN, 2011, 196 p.

3. **Jurevich E. I.** ed. *Intellektual'nye roboty* (Intelligent Robots), Mashinostroenie, 2007, 360 p.

4. **Kalyaev I. A.** *Metod kollektivnogo upravlenija gruppoj ob'ektov* (Method of collective management of a group of objects), *Mehatronika, Avtomatizacija, Upravlenie*, 2004, no. 3, pp. 9—15.

5. **Kalyaev I. A., Gajduk A. R., Kapustjan S. G.** *Modeli i algoritmy kollektivnogo upravlenija v gruppah robotov* (Models and algorithms for collective management in groups of robots), Moscow, Fizmatlit, 2009, 278 p.

6. **Zuhovickij S. I., Radchik I. A.** *Matematicheskie metody setevogo planirovanija* (Mathematical methods of network planning), Nauka, 1965, 360 p.

7. **Kofman A., Debezej G.** *Setevye metody planirovanija* (Network planning methods), Progress, 1968, 182 p.

8. **Novikov D. A.** *Upravlenie proektami. Organizacionnye mehanizmy* (Project management. Institutional arrangements), PMSOFT, 2007, 140 p.

9. **RejngoI'd Je., Nivergel't Ju., Deo N.** *Kombinatornye algoritmy. Teorija i praktika* (Combinatorial algorithms. Theory and practice), Mir, 1980, 476 p.

ИНФОРМАЦИЯ

21—23 октября 2015 г.
в Москве в ИПУ им. В. А. Трапезникова РАН состоится

Седьмая Всероссийская научно-практическая

"ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ. ТЕОРИЯ И ПРАКТИКА" (ИММОД-2015)

Тематические направления конференции:

- Теоретические основы и методология имитационного и комплексного моделирования
- Методы оценивания качества моделей полимодельных комплексов
- Методы и системы распределенного моделирования
- Моделирование глобальных процессов
- Средства автоматизации и визуализации имитационного моделирования
- Системная динамика (с обязательным наличием имитационной составляющей в созданной, либо использованной модельно-алгоритмической разработке)
- Практическое применение моделирования и инструментальных средств автоматизации моделирования
- Имитационное и комплексное моделирование в обучении и образовании

Подробную информацию о конференции см. на сайте:
http://www.simulation.su/