# Формальная модель задания в распределенных вычислительных средах*

А.В. Шамакина, Л.Б. Соколинский

ФГБОУ ВПО «Южно-Уральский государственный университет» (НИУ)

В работе описывается новая формальная модель задания в распределенных вычислительных средах. Разработанная модель задания позволяет описать поток работ в виде размеченного взвешенного ориентированного ациклического графа, предусматривает использование кластеризации вершин графа. Данная модель дает возможность задать масштабируемость отдельных задач в задании и количество процессорных ядер для каждого из вычислительных узлов. Модель может быть использована для описания алгоритмов планирования ресурсов в распределенных вычислительных средах.

## 1. Введение

На сегодняшний день предложено большое количество алгоритмов планирования, ориентированных на их использование в распределенных вычислительных средах [1-7]. Некоторые из алгоритмов производят планирование с учетом потоков работ в сложных приложениях. Однако разработанные алгоритмы часто не учитывают специфику области задачи и не используют дополнительные знания о ней, алгоритмы планирования рассматривают информацию о прикладных сервисах, но не используют знания о производственном процессе, описываемом потоком работ. В связи с этим, актуальной является задача разработки методов управления ресурсами в распределенных вычислительных средах, которые позволят создать эффективную и действенную систему планирования ресурсами. Для достижения этой цели необходимо построить формальную модель задания, которая представляет поток работ в виде ориентированного ациклического графа, предусматривает использование кластеризации вершин графа, а также возможность масштабирования отдельных задач задания.

Разрабатываемая формальная модель задания должна также давать возможность для описания новых алгоритмов планирования ресурсов в распределенных вычислительных средах. В частности, позволять описать задание с масштабируемыми задачами и вычислительные узлы с заданным числом процессорных ядер.

## 2. Формальная модель задания

Для описания формальной модели задания нам понадобятся следующие базовые определения.

*Ориентированным графом* называется четверка $G = \langle V, E, init, fin \rangle$, где $V$ – множество вершин; $E$ – множество дуг; $init: E \to V$ – функция, определяющая *начальную вершину* дуги; $fin: E \to V$ – функция, определяющая *конечную вершину* дуги.

Вершина $v_1, v_2 \in V$ называются *смежными*, если

$$\exists e \in E \left( \left( v_1 = init(e) \,\&\, v_2 = fin(e) \right) \vee \left( v_1 = fin(e) \,\&\, v_2 = init(e) \right) \right), \quad (1)$$

другими словами, существует дуга $e$, соединяющая эти вершины. Если $v_1 = init(e) \,\&\, v_2 = fin(e)$, мы будем обозначать это следующим образом: $(v_1, v_2) = e \in E$ и говорить, что вершины $v_1, v_2$ *инцидентны* дуге $e$.

Пусть $v_1, v_2 \in V$ и $n \geq 1$. Упорядоченная последовательность дуг

$$(e_1, e_2, \dots, e_n) \in E^n$$

называется *путем длины* $n$, от вершины $v_1$ к вершине $v_2$, если $v_1 = init(e_1)$, $v_2 = fin(e_n)$ и $fin(e_i) = init(e_{i+1})$ для всех $i \in \{1, \dots, n-1\}$. Если $(e_1, e_2, \dots, e_n) \in E^n$ – путь от вершины $v_1$ к вершине $v_2$, то *обратным путем* от вершины $v_2$ к вершине $v_1$ называется упорядоченная по-

* Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 14-07-31159.

следовательность дуг $(e_n, e_{n-1}, \dots, e_1) \in E^n$. Путь $(e_1, e_2, \dots, e_n)$ называется *простым*, если $init(e_1), init(e_2), \dots, init(e_n)$ все различны между собой и если $fin(e_1), fin(e_2), \dots, fin(e_n)$ тоже все различны. *Циклом* называется простой путь от некоторой вершины к ней самой. Ориентированный граф называется *ациклическим*, если он не содержит циклов [1].

Вершины $v_1, v_2 \in V$ называются *независимыми*, если не существует прямого или обратного пути от вершины $v_1$ к вершине $v_2$. В противном случае, вершины $v_1, v_2$ *зависимы*.

Пусть задан ориентированный граф $G = \langle V, E, init, fin \rangle$. *Взвешиванием графа G* будем называть функцию $\delta: E \to \mathbb{Z}_{\geq 0}$. *Разметкой графа G* будем называть функцию

$$\gamma : V \to \mathrm{N}^2. \tag{2}$$

Теперь мы готовы дать формальное определение задания в распределенной вычислительной среде.

*Графом задания* называется размеченный взвешенный ориентированный ациклический граф $G = \langle V, E, init, fin, \delta, \gamma \rangle$, где $V$ – множество вершин, соответствующих задачам, $E$ – множество дуг, соответствующих потокам данных.

Вес $\delta(e)$ дуги $e$ определяет объем данных, который необходимо передать по дуге $e$ от задачи, ассоциированной с вершиной $init(e)$ к задаче, ассоциированной с вершиной $fin(e)$. Метка

$$\gamma(v) = (m_v, t_v) \tag{3}$$

определяет максимальное количество процессорных ядер $m_v$, на которых задача $v$ имеет ускорение, близкое к *линейному*, и время $t_v$ выполнения задачи $v$ на одном ядре. Данная модель предполагает, что *вычислительная стоимость* $\chi(v, j_v)$ задачи $v$ на $j_v$ процессорных ядрах определяется следующей формулой:

$$\chi(v, j_v) = \begin{cases} t_v / j_v, & \text{если } 1 \leq j \leq m_v; \\ t_v / m_v, & \text{если } m_v < j_v. \end{cases} \tag{4}$$

Другими словами, при наращивании количества процессорных ядер в диапазоне от 1 до $m_v$, мы будем получать прямо пропорциональное уменьшение времени счета; при увеличении количества ядер в интервале от $m_v$ до $+\infty$ ускорение будет отсутствовать.

На рис. 1 приведен пример графа задания, содержащего 8 вершин. Для каждой из вершин указана метка в виде пары $(m_v, t_v)$. Каждой дуге графа сопоставляется вес – объем данных $\delta(e)$, передаваемых по данной дуге.

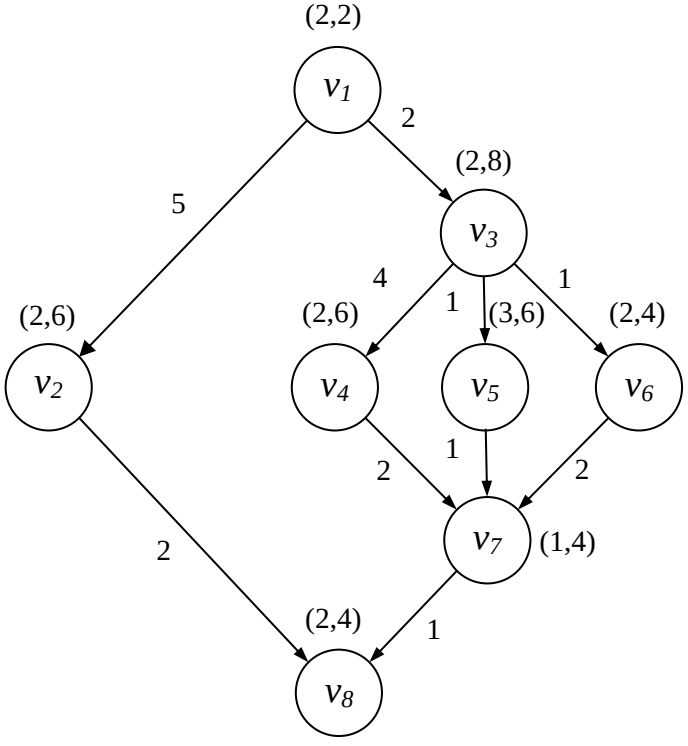



**Рис. 1**. Граф задания

*Вычислительным узлом* $P$ называется упорядоченное множество процессорных ядер $\{c_0, \dots, c_{q-1}\}$.

*Вычислительной системой* называется упорядоченное множество вычислительных узлов $\mathfrak{P} = \{P_0, \dots, P_{k-1}\}$. В реальности вычислительная система может являться распределенной вычислительной системой, объединяющей несколько вычислительных кластеров, каждый из которых является отдельным узлом этой системы.

*Кластеризацией* называется однозначное отображение $\omega: V \to \mathfrak{P}$ множества вершин $V$ графа задания $G$ на множество вычислительных узлов $\mathfrak{P}$.

Пусть задана вычислительная система $\mathfrak{P} = \{P_0, \dots, P_{k-1}\}$, состоящая из $k$ узлов. *Кластер* $W_i$ – это подмножество всех вершин, отображаемых на вычислительный узел $P_i \in \mathfrak{P}$:

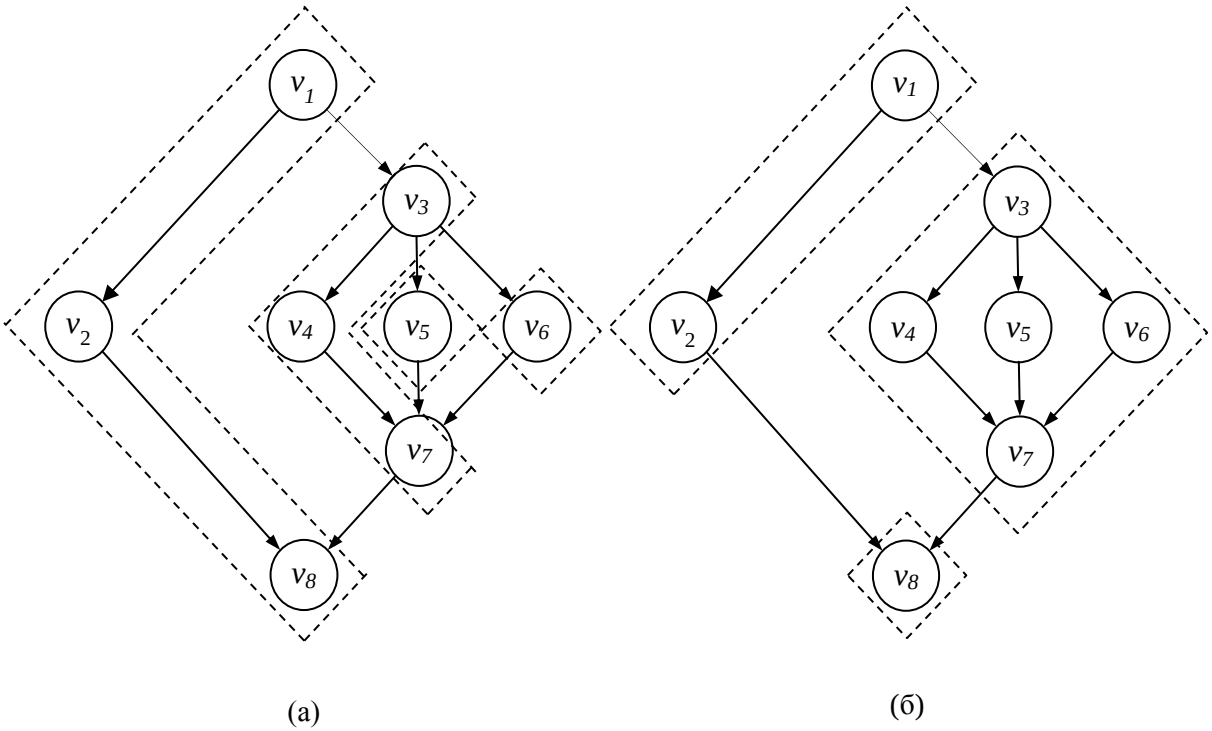
$$W_i = \{v \in V \mid \omega(v) = P_i \in \mathfrak{P}\}. \quad (5)$$

Имеем

$$W_i \cap W_j = \emptyset \text{ для } i \neq j; \quad (6)$$

$$V = \bigcup_{i=0}^{k-1} W_i. \quad (7)$$

Кластеризация называется *нелинейной*, если существуют две независимые вершины, которые отображаются на один вычислительный узел. В противном случае – кластеризация называется *линейной* [4]. На рис. 2 приведены примеры линейной и нелинейной кластеризации для графа задания, изображенного на рис. 1: (а) линейная кластеризация с четырьмя кластерами $\{v_1, v_2, v_8\}$, $\{v_3, v_4, v_7\}$, $\{v_5\}$ и $\{v_6\}$; (б) нелинейная кластеризация с кластерами $\{v_1, v_2\}$, $\{v_3, v_4, v_5, v_6, v_7\}$ и $\{v_8\}$.




(а) (б)

**Рис. 2.** (а) Линейная кластеризация; (б) нелинейная кластеризация

Пусть задан граф задания $G = \langle V, E, init, fin, \delta, \gamma \rangle$, для которого определена функция кластеризации $\omega(v)$. Будем называть такой граф *кластеризованным* и обозначать как $G = \langle V, E, init, fin, \delta, \gamma, \omega \rangle$.

В рамках модели мы предполагаем, что время передачи любого объема данных между узлами, принадлежащими одному кластеру, близко к нулю, а время передачи данных между узлами, принадлежащими разным кластерам, пропорционально объему передаваемых данных с коэффициентом 1. В соответствии с эти мы можем определить функцию $\sigma: E \to \mathbb{Z}_{\geq 0}$, вычисляющую *коммуникационную стоимость* (время) передачи данных по дуге $e \in E$, следующим образом:

$$\sigma(e) = \begin{cases} 0, & \text{если } \omega\big(init(e)\big) = \omega\big(fin(e)\big); \\ \delta(e), & \text{если } \omega\big(init(e)\big) \neq \omega\big(fin(e)\big). \end{cases} \quad (8)$$

Пусть задан кластеризованный граф $G = \langle V, E, init, fin, \delta, \gamma, \omega \rangle$. *Расписанием* называется отображение $\xi: V \to \mathbb{Z}_{\geq 0} \times \mathbb{N}$, которое произвольной вершине $v \in V$ сопоставляет двойку чисел

$$\xi(v) = (\tau_v, j_v), \quad (9)$$

где $\tau_v$ определяет время запуска задачи $v$, $j_v$ – количество процессорных ядер, выделяемых задаче $v$. Обозначим через $s_v$ – время останова задачи $v$. Имеем

$$s_v = \tau_v + \chi(v, j_v), \quad (10)$$

где $\chi$ – функция временной сложности, определенная с помощью формулы (4). Расписание называется *корректным*, если оно удовлетворяет следующим условиям:

$$\forall\, e \in E \ \left(\tau_{fin(e)} \geq \tau_{init(e)} + \chi\big(init(e), j_{init(e)}\big) + \sigma(e)\right); \quad (11)$$

$$\forall\, v \in V \ (j_v \leq m_v); \quad (12)$$

$$\forall t \in \Gamma \left( \forall i \in \big[0, \ldots, k-1\big] \left( \sum_{\substack{v \in W_i \wedge \\ \tau_v < t \leq s_v}} j_v \leq \big|P_i\big| \right) \right). \quad (13)$$

Условие (11) означает, что для любых двух смежных вершин $v_1 = init(e)$ и $v_2 = fin(e)$ время запуска $v_2$ не может быть меньше суммы следующих величин: время запуска $v_1$, время выполнения $v_1$, коммуникационная стоимость дуги $e$. Условие (12) означает, что количество ядер, выделяемое задаче $v_1$, не превышает границы линейной масштабируемости, заданной разметкой $\gamma$ в контексте формулы (3). Условие (13) означает, что в любой момент времени $t$ общее количество процессорных ядер, выделяемых задачам на узле с номером $i$, не может превосходить количества ядер на этом узле. В дальнейшем мы будем рассматривать только корректные расписания, если явно не оговорено противное.

*Планом* выполнения расписания $\xi$ называется множество многозначных отображений

$$\alpha = \{\alpha_i: W_i \to P_i | i = 0, \ldots, k-1\},$$

удовлетворяющих следующим условиям $\forall i = 0, \ldots, k-1$ и $(\tau_v, j_v) := \xi(v)$:

$$\forall\, v \in W_i \ (j_v = |\alpha_i(v)|); \quad (14)$$

$$\forall\, v_1, v_2 \in W_i \ \big(v_1 \neq v_2 \wedge \big[\tau_{v_1}, s_{v_1}\big] \cap \big[\tau_{v_2}, s_{v_2}\big] \neq \emptyset \ \Rightarrow\ \alpha_i(v_1) \cap \alpha_i(v_2) = \emptyset\big). \quad (15)$$

Условие (14) означает, что количество процессорных ядер, выделяемых задаче $v$ в соответствии с планом $\alpha$, равно количеству процессорных ядер, указанных для этой задачи в расписании $\xi$. Условие (15) означает, что, если в соответствии с расписанием $\xi$ задачи $v_1$, и $v_2$ пересекаются по времени своего выполнения, то им должны выделяться различные процессорные ядра (мы предполагаем, что на одном процессорном ядре в каждый момент времени не может выполняться более одной задачи).

Кластеризованный граф $G = \langle V, E, init, fin, \delta, \gamma, \omega \rangle$ с заданными расписанием $\xi$ и соответствующим планом $\alpha$ будем называть *распланированным* и обозначать как $G = \langle V, E, init, fin, \delta, \gamma, \omega, \xi, \alpha \rangle$.

*Ярусно-параллельной формой (ЯПФ)* [8] называется разбиение множества вершин $V$ ориентированного ациклического графа $G = \langle V, E, init, fin \rangle$ на перенумерованные подмножества (ярусы) $L_i$ $(i = 1, \ldots, r)$, удовлетворяющие следующим свойствам:

$$\left.\begin{array}{l} V = \bigcup_{i=1}^{r} L_i; \\ \forall i \neq j \in \{1, \ldots . r\} \big(L_i \cap L_j = \varnothing\big); \\ \forall (v_1, v_2) \in E \Big(\forall i \neq j \in \{1, \ldots . r\} \big(v_1 \in L_i \wedge v_2 \in L_j \Rightarrow i < j\big)\Big). \end{array}\right\} \quad (16)$$

Последнее условие означает, что если из вершины $v_1$ идет дуга в вершину $v_2$, то вершина $v_2$ должна располагаться на ярусе с большим номером по отношению к ярусу, на котором распо-

лагается вершина $v_1$. Количество вершин в ярусе $L_i$ называется его *шириной*. Количество ярусов в ЯПФ называется ее *высотой*, а максимальная ширина ее ярусов – *шириной ЯПФ*. ЯПФ называется *канонической* [10], если все *входные* (не имеющие входных дуг) вершины принадлежат ярусу с номером 1, и максимальная длина пути, оканчивающегося в вершине, принадлежащей ярусу *k*, равна *k-1*.

Пусть в распланированном графе $G = \langle V, E, init, fin, \delta, \gamma, \omega, \xi, \alpha \rangle$ задан простой путь $y = (e_1, e_2, \dots, e_n)$. *Стоимостью* пути $u(y)$ называется величина

$$u(y) = \chi\left( fin(e_n), j_{fin(e_n)} \right) + \sum_{i=1}^{n} \left( \chi\left( init(e_i), j_{init(e_i)} \right) + \max\left( \sigma(e_i), \tau_{fin(e_i)} - s_{init(e_i)} \right) \right), \qquad (17)$$

где $\chi$ – функция, определяемая формулой (4), значением которой является вычислительная стоимость вершины; $\sigma$ – функция, определяемая формулой (8), значением которой является коммуникационная стоимость дуги; $j_v$ и $\tau_v$ определяется по формуле (9); $s_v$ определяется по формуле (10).

Пусть Y – множество всех простых путей в распланированном графе $G = \langle V, E, init, fin, \delta, \gamma, \omega, \xi, \alpha \rangle$. Простой путь $\bar{y} \in$ Y называется *критическим*, если

$$u(\bar{y}) = \max_{y \in Y} u(y), \qquad (18)$$

то есть, критический путь обладает максимальной стоимостью.

**Утверждение 1**. Любой критический путь в распланированном графе $G = \langle V, E, init, fin, \delta, \gamma, \omega, \xi, \alpha \rangle$ начинается с входной вершины и заканчивается в *выходной* (не имеющей выходных дуг) вершине.

Доказательство проведем методом от противного. Предположим, что существует критический путь $\bar{y} = (e_1, \dots, e_n)$, начинающийся не с входной вершины. Тогда существует дуга $e_h$ такая, что $fin(e_h) = init(e_1)$, то есть в $G$ существует путь $\tilde{y} = (e_h, e_1, \dots, e_n)$. В силу (2) – (4) и (17) получаем $u(\bar{y}) < u(\tilde{y})$. Это противоречит (18). Аналогичным образом приходим к противоречию при предположении, что что существует критический путь, заканчивающийся не в выходной вершине. Утверждение доказано.

## 3. Пример

Рассмотрим пример вычислительной системы $\mathfrak{P} = \{P_0, P_1\}$, состоящей из двух вычислительных узлов. Вычислительный узел $P_0$ включает в себя четыре процессорных ядра: $P_0 = \{c_{00}, c_{01}, c_{02}, c_{03}\}$. Вычислительный узел $P_1$ включает в себя шесть процессорных ядер: $P_1 = \{c_{10}, c_{11}, c_{12}, c_{13}, c_{14}, c_{15}\}$. Рассмотрим граф задания $G$, приведенный на рис. 1. Определим функцию разметки $\gamma(v) = (m_v, t_v)$ для графа $G$ с помощью табл. 1.

**Таблица 1.** Функция разметки $\gamma$

| **Вершина $v$** | $\gamma(v) = (m_v, t_v)$ | |
|---|---|---|
| | $m_v$ | $t_v$ |
| $v_1$ | 2 | 2 |
| $v_2$ | 2 | 6 |
| $v_3$ | 2 | 8 |
| $v_4$ | 2 | 6 |
| $v_5$ | 3 | 6 |
| $v_6$ | 2 | 4 |
| $v_7$ | 1 | 4 |
| $v_8$ | 2 | 4 |

В соответствии с (3) величина $m_v$ определяет верхнюю границу линейной масштабируемости, а величина $t_v$ – время выполнения на одном процессорном ядре задачи, ассоциированной с вершиной $v$.

Определим для графа $G$ с помощью табл. 2 весовую функцию $\delta$, задающую объем данных, передаваемых по каждой дуге.

**Таблица 2.** Весовая функция $\delta$

| Дуга $e$ | $\delta(e)$ |
|---|---|
| $(v_1, v_2)$ | 5 |
| $(v_1, v_3)$ | 2 |
| $(v_2, v_8)$ | 2 |
| $(v_3, v_4)$ | 4 |
| $(v_3, v_5)$ | 1 |
| $(v_3, v_6)$ | 1 |
| $(v_4, v_7)$ | 2 |
| $(v_5, v_7)$ | 1 |
| $(v_6, v_7)$ | 2 |
| $(v_7, v_8)$ | 1 |

Зададим для графа $G$ и вычислительной системы $\mathfrak{P}$ функцию кластеризации $\omega$ с помощью табл. 3. Вершины $v_1, v_2, v_8$ отображаются на вычислительный узел $P_0$, а вершины $v_3, v_4, v_5, v_6, v_7$ отображаются на вычислительный узел $P_1$. В соответствии с этим, множество вершин $V$ графа задания $G$ разбивается функцией кластеризации $\omega(v)$ на два кластера: $W_0 = \{v_1, v_2, v_8\}$ и $W_1 = \{v_3, v_4, v_5, v_6, v_7\}$.

**Таблица 3.** Функция кластеризации $\omega$

| Вершина $v$ | $\omega(v)$ |
|---|---|
| $v_1$ | $P_0$ |
| $v_2$ | $P_0$ |
| $v_3$ | $P_1$ |
| $v_4$ | $P_1$ |
| $v_5$ | $P_1$ |
| $v_6$ | $P_1$ |
| $v_7$ | $P_1$ |
| $v_8$ | $P_0$ |

С помощью формулы (8) вычислим коммуникационные стоимости для дуг графа $G$, кластеризованного с использованием функции $\omega$, заданной табл. 3. Полученные значения приведены в табл. 4.

**Таблица 4.** Коммуникационная стоимость σ

| Дуга $e$ | Коммуникационная стоимость $\sigma(e)$ |
|---|---|
| $(v_1, v_2)$ | 0 |
| $(v_1, v_3)$ | 2 |
| $(v_2, v_8)$ | 0 |
| $(v_3, v_4)$ | 0 |
| $(v_3, v_5)$ | 0 |
| $(v_3, v_6)$ | 0 |
| $(v_4, v_7)$ | 0 |
| $(v_5, v_7)$ | 0 |
| $(v_6, v_7)$ | 0 |
| $(v_7, v_8)$ | 1 |

Кластеризованный таким образом граф $G = \langle V, E, init, fin, \delta, \gamma, \omega \rangle$ представлен на рис. 3, где для каждой дуги указана получившаяся коммуникационная стоимость. Мы видим, что коммуникационная стоимость передачи данных внутри кластеров равна нулю. Ненулевая стоимость передачи данных сохранилась для дуг, соединяющих вершины из разных кластеров: $(v_1, v_3)$ и $(v_7, v_8)$.

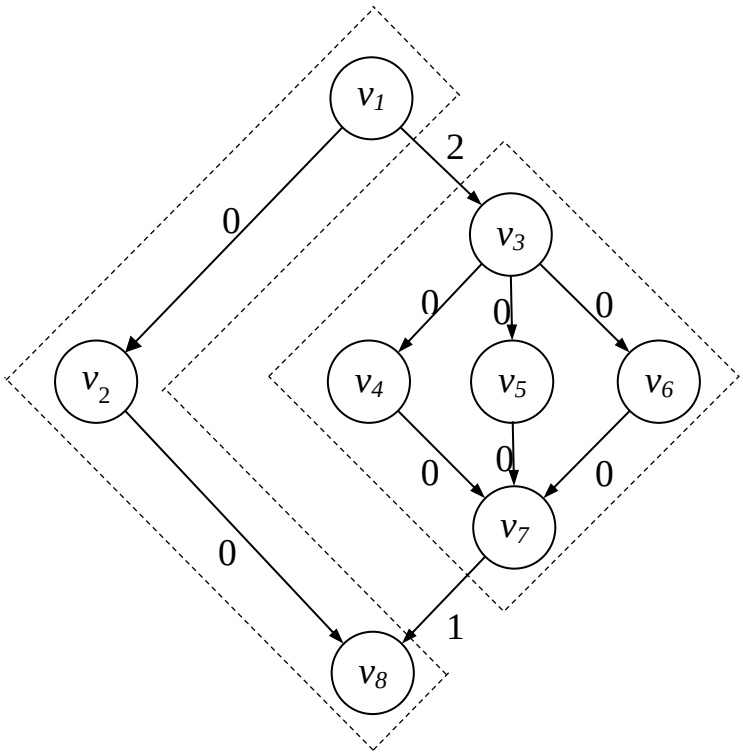



**Рис. 3.** Кластеризованный граф с указанием коммуникационной стоимости для дуг

**Таблица 5.** Расписание

| **Вершина $v$** | $\xi(v) = (\tau_v, j_v)$ | |
|---|---|---|
| | **Время запуска $\tau_v$** | **Количество ядер $j_v$** |
| $v_1$ | 0 | 2 |
| $v_2$ | 1 | 2 |
| $v_3$ | 3 | 2 |
| $v_4$ | 7 | 2 |
| $v_5$ | 7 | 2 |
| $v_6$ | 7 | 2 |
| $v_7$ | 10 | 1 |
| $v_8$ | 14 | 2 |

С помощью табл. 5 зададим для графа $G$, изображенного на рис. 3, функцию расписания $\xi$. Покажем, что данное расписание является корректным. Для этого необходимо и достаточно проверить выполнение условий (11)-(13). Проверим условие (11) означающее, что для любых двух смежных вершин $v_1 = init(e)$ и $v_2 = fin(e)$ время запуска $v_2$ не может быть меньше суммы следующих величин: время запуска $v_1$, вычислительная стоимость $v_1$, коммуникационная стоимость дуги $e$. По формуле (4) рассчитаем вычислительную стоимость $\chi$ для каждой вершины графа $G$ (см. табл. 6).

**Таблица 6.** Вычислительная стоимость

| **Вершина $v$** | $\chi(v, j_v)$ |
|---|---|
| $v_1$ | 1 |
| $v_2$ | 3 |
| $v_3$ | 4 |
| $v_4$ | 3 |
| $v_5$ | 3 |
| $v_6$ | 2 |
| $v_7$ | 4 |
| $v_8$ | 2 |

Вычислим значения, приведенные в табл. 7. Значения $\tau_{v'}$ мы получаем из таблица 5, значения выражения $\tau_v + \chi(v, j_v) + \sigma(e)$ вычисляются с использованием данных из табл. 4, табл. 5 и табл. 6.

**Таблица 7.** Данные для проверки условия (11)

| Дуга $e = (v, v')$ | $\tau_{v'}$ | $\tau_v + \chi(v, j_v) + \sigma(e)$ |
|---|---|---|
| $(v_1, v_2)$ | 1 | 1 |
| $(v_1, v_3)$ | 3 | 3 |
| $(v_2, v_8)$ | 14 | 4 |
| $(v_3, v_4)$ | 7 | 7 |
| $(v_3, v_5)$ | 7 | 7 |
| $(v_3, v_6)$ | 7 | 7 |
| $(v_4, v_7)$ | 10 | 10 |
| $(v_5, v_7)$ | 10 | 10 |
| $(v_6, v_7)$ | 10 | 9 |
| $(v_7, v_8)$ | 14 | 15 |

Из таблицы видно, что для любой дуги $(v, v')$ выполняется условие $\tau_{v'} \geq \tau_v + \chi(v, j_v) + \sigma(e)$, что равносильно

$$\tau_{fin(e)} \geq \tau_{init(e)} + \chi\left(init(e), j_{init(e)}\right) + \sigma(e).$$

Следовательно, условие (11) для расписания $\xi$ выполняется.

Покажем, что выполняется условие (12), означающее, что количество ядер, выделяемое задаче $v_1$, не превышает границы линейной масштабируемости, заданной разметкой $\gamma$ в контексте формулы (3). Для этого заполним табл. 8. Значения $j_v$ взяты из табл. 5, значения $m_v$ – из табл. 1. Мы видим, что для всех вершин $v$ графа $G$ выполняется необходимое условие $j_v \leq m_v$.

**Таблица 8.** Данные для проверки условия (12)

| Вершина $v$ | $j_v$ | $m_v$ |
|---|---|---|
| $v_1$ | 2 | 2 |
| $v_2$ | 2 | 2 |
| $v_3$ | 2 | 2 |
| $v_4$ | 2 | 2 |
| $v_5$ | 2 | 3 |
| $v_6$ | 2 | 2 |
| $v_7$ | 1 | 1 |
| $v_8$ | 2 | 2 |

Проверим условие (13) означающее, что в любой момент времени $t$ общее количество процессорных ядер, выделяемых задачам на узле с номером $i$, не может превосходить количества ядер на этом узле . В нашем примере граф $G$ был разбит на два кластера $W_0$ и $W_1$. Поэтому нам достаточно проверить условие (13) для каждого кластера в отдельности. Сначала проверим условие (13) для кластера $W_0$. Для этого заполним табл. 9, содержащую данные о времени запуска и останова задач, ассоциированных с вершинами кластера $W_0$. Для каждой вершины $v$ время запуска $\tau_v$ получаем из табл. 5, а время останова $s_v$ рассчитывается по формуле (10).

**Таблица 9.** Время запуска и останова для кластера $W_0$

| Вершина $v$ | Время запуска $\tau_v$ | Время останова $s_v$ |
|---|---|---|
| $v_1$ | 0 | 1 |
| $v_2$ | 1 | 4 |
| $v_8$ | 14 | 16 |

Теперь построим табл. 10. Согласно табл. 9 мы должны рассмотреть моменты времени $t$ от 1 до 16 (нижняя граница временного интервала равна <наименьшее время запуска>+1, верхняя граница равна <наибольшее время останова>. Вычислим значения $j_{v_1}, j_{v_2}, j_{v_8}$ для вершин, входящих в кластер $W_0$, для момента времени $t_1$=1 (первая строка таблицы). Из табл. 9 мы видим, что условие $\tau_{v_1} < t_1 \leq s_{v_1}$ – истинно. Следовательно, задача $v_1$ выполняется в момент времени

$t_1$. Из табл. 5 мы находим количество процессорных ядер $j_{v_1} = 2$, выделяемых задаче $v_1$ в соответствии с заданным расписанием $\xi$. Помещаем это число в первую строку таблицы в колонку «$j_{v_1}$». Для задачи $v_1$, в соответствии с табл. 9, мы видим, что условие $\tau_{v_2} < t_2 \le s_{v_2}$ – ложно. Следовательно, задача $v_2$ не выполняется в момент времени $t_1$. Поэтому в столбец «$j_{v_2}$» первой стоки помещаем значение 0. По этой же причине в столбце «$j_{v_8}$» первой стоки также помещается значение 0. Аналогичным образом заполняем остальные строки таблицы. Для каждой строки считаем сумму значений из столбцов «$j_{v_1}, j_{v_2}, j_{v_8}$». Очевидно, что полученное число будет совпадать со значением выражения

$$\sum_{\substack{v \in W_0 \wedge \\ \tau_v < t \le s_v}} j_v.$$

Таким образом мы получаем

$$\forall t \in [1,16] \left( \sum_{\substack{v \in W_0 \wedge \\ \tau_v < t \le s_v}} j_v \le 4 = \left| P_0 \right| \right).$$

Поскольку

$$\forall t \notin [1,16] \left( \sum_{\substack{v \in W_0 \wedge \\ \tau_v < t \le s_v}} j_v = 0 \right),$$

получаем, что в любой момент времени $t$ выполняется неравенство

$$\sum_{\substack{v \in W_0 \wedge \\ \tau_v < t \le s_v}} j_v \le \left| P_0 \right|,$$

что обеспечивает выполнение условие (13) для кластера $W_0$.

**Таблица 10.** Данные для проверки условия (13) для кластера $W_0$

| **Момент времени $t$** | **Количество ядер, выделенных задаче** | | | $\sum_{\substack{v \in W_0 \wedge \\ \tau_v < t \le s_v}} j_v$ |
|---|---|---|---|---|
| | $j_{v_1}$ | $j_{v_2}$ | $j_{v_8}$ | |
| 1 | 2 | 0 | 0 | 2 |
| 2 | 0 | 2 | 0 | 2 |
| 3 | 0 | 2 | 0 | 2 |
| 4 | 0 | 2 | 0 | 2 |
| 5 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 2 | 2 |
| 16 | 0 | 0 | 2 | 2 |

Теперь проверим выполнение условия (13) для кластера $W_1$. Сначала строим табл. 11. Затем строим табл. 12, из которой мы видим, что

$$\forall t \in [4,14]\left( \sum_{\substack{v \in W_1 \wedge \\ \tau_v < t \le s_v}} j_v \le 6 = |P_1| \right).$$

Поскольку

$$\forall t \notin [4,14]\left( \sum_{\substack{v \in W_1 \wedge \\ \tau_v < t \le s_v}} j_v = 0 \right),$$

получаем, что в любой момент времени $t$ выполняется неравенство

$$\sum_{\substack{v \in W_1 \wedge \\ \tau_v < t \le s_v}} j_v \le |P_1|,$$

что обеспечивает выполнение условие (13) для кластера $W_1$, а значит и для всего графа G.

**Таблица 11.** Время запуска и останова для кластера $W_1$

| **Вершина $v$** | **Время запуска $\tau_v$** | **Время останова $s_v$** |
|---|---|---|
| $v_3$ | 3 | 7 |
| $v_4$ | 7 | 10 |
| $v_5$ | 7 | 10 |
| $v_6$ | 7 | 9 |
| $v_7$ | 10 | 14 |

**Таблица 12.** Данные для проверки условия (13) для кластера $W_1$

| **Момент времени $t$** | **Количество ядер, выделенных задаче** | | | | | $\sum_{\substack{v \in W_1 \wedge \\ \tau_v < t \le s_v}} j_v$ |
|---|---|---|---|---|---|---|
| | $j_{v_3}$ | $j_{v_4}$ | $j_{v_5}$ | $j_{v_6}$ | $j_{v_7}$ | |
| 4 | 2 | 0 | 0 | 0 | 0 | 2 |
| 5 | 2 | 0 | 0 | 0 | 0 | 2 |
| 6 | 2 | 0 | 0 | 0 | 0 | 2 |
| 7 | 2 | 0 | 0 | 0 | 0 | 2 |
| 8 | 0 | 2 | 2 | 2 | 0 | 6 |
| 9 | 0 | 2 | 2 | 2 | 0 | 6 |
| 10 | 0 | 2 | 2 | 0 | 0 | 4 |
| 11 | 0 | 0 | 0 | 0 | 1 | 1 |
| 12 | 0 | 0 | 0 | 0 | 1 | 1 |
| 13 | 0 | 0 | 0 | 0 | 1 | 1 |
| 14 | 0 | 0 | 0 | 0 | 1 | 1 |

Таким образом, мы показали, что условия (11)-(13) выполняются для расписания $\xi$, следовательно оно корректно.

С помощью табл. 13 и табл. 14 зададим план $\alpha = \{\alpha_0, \alpha_1\}$ выполнения расписания $\xi$.

**Таблица 13.** Отображение $\alpha_0 : W_0 \to P_0$

| **Задача** | **Процессорные ядра** |
|---|---|
| $v_1$ | $\{c_{00}, c_{01}\}$ |
| $v_2$ | $\{c_{00}, c_{01}\}$ |
| $v_8$ | $\{c_{00}, c_{01}\}$ |

**Таблица 14.** Отображение $\alpha_1: W_1 \to P_1$

| Задача | Процессорные ядра |
|---|---|
| $v_3$ | $\{c_{10}, c_{11}\}$ |
| $v_4$ | $\{c_{10}, c_{11}\}$ |
| $v_5$ | $\{c_{12}, c_{13}\}$ |
| $v_6$ | $\{c_{14}, c_{15}\}$ |
| $v_7$ | $\{c_{10}\}$ |

На рис. 4 представлена диаграмма Ганта [11] для плана $\alpha$ выполнения расписание $\xi$, которая в графическом виде описывает временное и пространственное размещение задач графа $G$ на вычислительных узлах $P_i \in \mathfrak{P}$, $i = \{0,1\}$. По вертикальной оси диаграммы Ганта отмечены процессорные ядра, по горизонтальной оси – время. Каждой задаче сопоставлена прямоугольная область, показывающая на каких ядрах она выполняется и какое время занимает. Данные для построения прямоугольников берутся из табл. 9, табл. 11, табл. 13 и табл. 14.

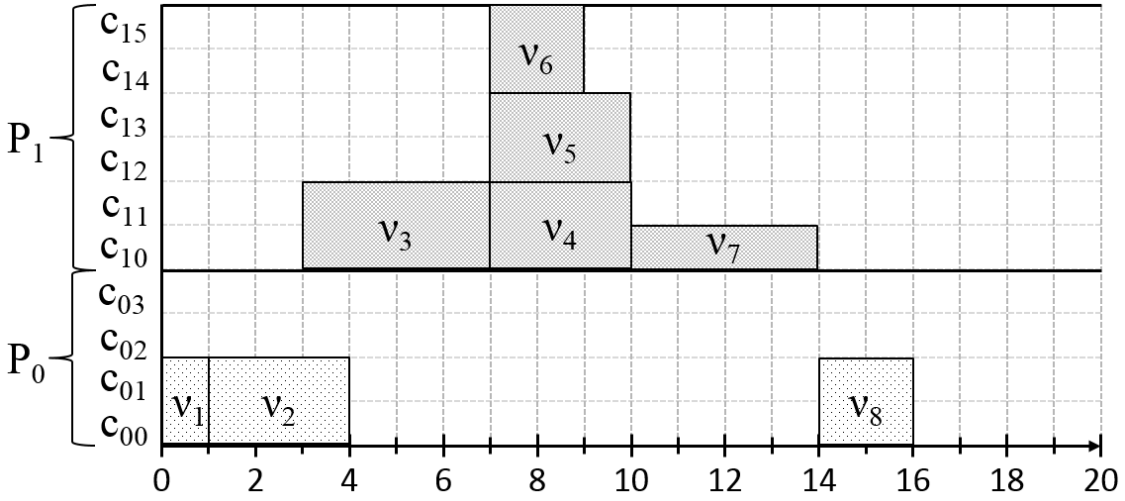



**Рис. 4.** Диаграмма Ганта

Мы видим, что задачи, размещенные на одном вычислительном узле, не требуют времени на передачу данных между ними.

## 4. Заключение

Разработанная формальная модель задания позволяет описать поток работ в виде графа задания, предусматривает использование кластеризации вершин графа, возможность указать время выполнения каждой задачи и ее верхнюю границу линейной масштабируемости, а также возможность задать количество процессорных ядер для каждого из вычислительных узлов. Данная формальная модель задания является универсальной, поскольку предоставляет возможность описать как уже известные алгоритмы кластеризации, так и новые алгоритмы для планирования ресурсов в распределенных вычислительных средах.

## Литература


1. Kim S.J. A general approach to multiprocessor scheduling. Report TR-88-04. Department of Computer Science, University of Texas at Austin, 1988.
2. Kim S.J., Browne J.C. A general approach to mapping of parallel computation upon multiprocessor architectures // Proceedings of the International Conference on Parallel Processing, 1988. Vol. 3. P. 1-8.
3. Sarkar V. Partitioning and Scheduling Parallel Programs for Execution on Multiprocessors. The MIT Press, Cambridge, MA, 1989. P. 215.

4. Gerasoulis A., Yang T. A comparison of clustering heuristics for scheduling directed acyclic graphs on multiprocessors // Journal of Parallel and Distributed Computing, 1992. Vol. 16, No. 4. P. 276-291.
5. Yang T., Gerasoulis A. DSC: Scheduling Parallel Tasks on an Unbounded Number of Processors // IEEE Transactions on Parallel and Distributed Systems. 1994. Vol. 5, No. 9. P. 951-967.
6. Shamakina A. Brokering Service for Supporting Problem-Oriented Grid Environments // UNICORE Summit 2012 Proceedings, Forschungszentrum Julich, 2012. P. 67-75.
7. Шамакина А.В. Брокер ресурсов для поддержки проблемно-ориентированных сред // Вестник ЮУрГУ. Серия "Вычислительная математика и информатика". 2012. № 46(305). Вып. 1. С. 88-98.
8. Кнут Д.Э. Искусство программирования, т. 1. Основные алгоритмы, 3-е изд. М.: Издательский дом «Вильямс», 2000. 720 с.
9. Воеводин В.В. Математические модели и методы в параллельных процессах. Наука, 1986. 296 с.
10. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. Санкт-Петербург: БХВ-Петербург, 2002. 608 с.
11. Wilson J.M. Gantt charts: A centenary appreciation // European Journal of Operational Research. 2003. Vol. 149, No. 2. P. 430-437.