

# МЕТОДЫ И ЭВРИСТИКИ ПЛАНИРОВАНИЯ В РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЯХ С НЕОТЧУЖДАЕМЫМИ РЕСУРСАМИ<sup>1</sup>

*В.В. Топорков, А.В. Бобченков, Д.М. Емельянов, А.С. Целищев*

Статья посвящена вопросам эффективной организации вычислений в виртуальных организациях распределенных сред с неотчуждаемыми ресурсами, включая коммерчески используемый грид. Предлагается и обосновывается комплексное сочетание методов и эвристик планирования и распределения ресурсов на уровне потоков заданий и приложений, реализующих принятую в виртуальной организации политику предоставления и потребления ресурсов с использованием экономических принципов.

*Ключевые слова:* распределенные вычисления, планирование, оптимизация, слот, альтернатива, циклическая схема планирования.

## Введение

Решение больших задач в распределенных вычислительных средах требует привлечения значительных ресурсов, часть из которых может использоваться совместно с их владельцами [1–3]. Известные алгоритмы, их комбинации и эвристики, используемые в программах-планировщиках, как правило, не позволяют получать эффективные или близкие к оптимальным планы в условиях разнородных распределенных сред с динамично изменяющимся составом неотчуждаемых вычислительных ресурсов. В этих условиях весьма эффективными оказываются так называемые экономические модели планирования распределенных вычислений [1–3].

Среди различных подходов к планированию вычислений в распределенных средах можно выявить следующие тенденции. Одна из них основывается на использовании доступных ресурсов и планировании вычислений на уровне приложений. Зачастую при этом не предполагается наличия какого-либо регламента в предоставлении ресурсов [4]. Роль посредников между пользователями и вычислительными узлами выполняют агенты приложений – брокеры ресурсов. В рамках этого направления выполнен ряд проектов - AppLeS, APST, Legion, DRM, Condor-G, Nimrod/G и другие. Другая тенденция связана с образованием виртуальных организаций (ВО) пользователей и предполагает планирование на уровне потоков заданий. В этой модели внешний планировщик (грид-диспетчер, метапланировщик, метаброкер [5, 6]) выполняет роль промежуточного звена между пользователями и множеством систем управления заданиями на удаленных вычислительных узлах.

В рамках первого из направлений системы планирования и управления ресурсами являются хорошо масштабируемыми и адаптируемыми к особенностям приложений. Однако использование независимыми пользователями различных критериев для оптимизации планов выполнения своих заданий (в условиях возможной конкуренции с дру-

---

<sup>1</sup> Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии – 2014».

гими заданиями) может ухудшать такие интегральные характеристики распределенной среды, как время выполнения пакета заданий и загрузка ресурсов. Образование ВО, с одной стороны, естественным образом ограничивает масштабируемость систем управления заданиями. С другой стороны, наличие определенных правил предоставления и потребления ресурсов, основанных, в частности, на экономических моделях [1–3, 5–9], позволяет повысить эффективность планирования и распределения ресурсов на уровне потоков заданий.

Идея «конвергенции» вышеупомянутых подходов была декларирована довольно давно [5, 10, 11]. Тем не менее, успешные попытки ее реализации обладают рядом серьезных ограничений. Так, в одних из известных моделей отыскивается лишь подходящий набор ресурсов [12–14] и не поддерживаются оптимизационные механизмы планирования заданий. В других моделях [5, 7, 8], не представлены аспекты, связанные с динамикой изменения загрузки узлов, конкуренцией независимых пользователей, а также глобальных и локальных потоков заданий. Подход к планированию в ВО, предлагаемый в данной работе, принципиально отличается от известных решений тем, что строится на сочетании методов и эвристик планирования параллельных приложений и управления потоками независимых заданий [10, 11].

## 1. Модель справедливого разделения ресурсов

### 1.1. Циклическая схема и стратегии планирования

Модель для управления выполнением независимых заданий основывается на экономических принципах, а планирование заданий выполняется циклично на альтернативных наборах предварительно отобранных ресурсов [15, 16]. В отличие от известных, рассматриваемая модель предполагает группирование заданий в потоки и пакеты в соответствии с их свойствами и потребностями в ресурсах, а также циклическое планирование системы заданий на основе динамично обновляемых стратегий и ограничений [15]. План выполнения системы заданий оптимизируется по совокупности критериев в соответствии с политикой справедливого предоставления и потребления ресурсов, принятой в виртуальной организации.

Циклический подход к планированию рассматривается в рамках модели, в основу которой заложена иерархическая схема управления потоками заданий [15, 16]. Планирование потока заданий осуществляется по пакетам на основе расписаний выполнения заданий в локальных процессорных узлах – в соответствии с циклической схемой планирования (ЦСП) (рис. 1).

Наборы доступных слотов и стоимость  $C_j$  их использования, определяемая их владельцами, динамично определяются по данным, поступающим от локальных менеджеров ресурсов или систем пакетной обработки заданий. В каждом цикле планирования требуется решение двух задач. Во-первых, необходимо отобрать подходящие (по производительности, времени и стоимости) наборы слотов – альтернативы выполнения для каждого задания пакета на данном цикле планирования. Каждая альтернатива характеризуется суммарной стоимостью, временем выполнения, старта, завершения и другими показателями, например, уровнем энергопотребления. Во-вторых, требуется выбрать комбинацию альтернатив, оптимальную или эффективную с позиций прохождения всего пакета заданий в соответствии с принятой политикой ВО.

Решению вышеупомянутых задач для вектора критериев, формализующих политику ВО и формирующих стратегии планирования, посвящена работа [15]. Здесь, для иллюстрации основной идеи, рассмотрим задачу выбора оптимальной комбинации слотов (с одним критерием).

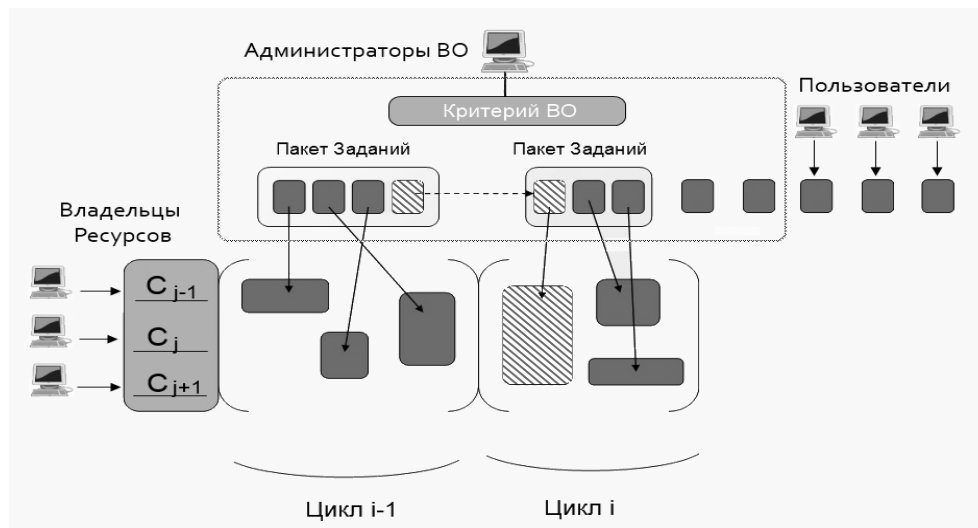


Рис. 1. Циклическое планирование потока заданий

Пусть  $S_i$  – множество наборов слотов, подходящих для выполнения  $i$ -го задания пакета,  $i = 1, \dots, n$ . Набор слотов  $s_j \in S_i$  является подходящим для  $i$ -го задания, если он удовлетворяет требованиям ресурсного запроса, цене  $c_i(s_j)$  и времени  $t_i(s_j)$ ,  $j = 1, \dots, N, N = \left| \bigcup_{i=1}^n S_i \right|$ . При этом  $S_1, \dots, S_n$  не обязательно не пересекаются. Комбинация представляет собой  $n$ -элементную последовательность  $\bar{s}$  не обязательно различных наборов слотов. Любое из заданий может использовать один и только один набор слотов. В то же время некоторые из наборов могут разделяться по времени или ресурсу несколькими заданиями одного и того же пакета. Обозначим через  $S$  множество комбинаций слотов для пакета заданий. Если для какого-либо из заданий подходящего набора слотов не существует, то его планирование переносится в следующий цикл и задание занимает место в очереди цикла планирования в соответствии с некоторым приоритетом (см. рис. 1). После просмотра слотов для всех  $n$  заданий пакета отыскиваются альтернативные наборы слотов, поскольку при отборе слотов для  $i$ -го задания не весь исходный список может быть просмотрен. Эта процедура итеративно реализуется для всех заданий, планирование которых не перенесено в следующий цикл.

Пусть  $f_i(s_j)$  – функция, определяющая эффективность выполнения  $i$ -го задания на наборе  $s_j$  слотов при допустимых затратах  $g_i(s_j)$ . В частности,  $f_i(s_j) = c_i(s_j)$  – стоимость использования набора  $s_j$  в течение времени  $g_i(s_j) = t_i(s_j)$  либо  $f_i(s_j) = t_i(s_j)$  – время выполнения  $i$ -го задания на наборе  $s_j$  слотов стоимостью  $g_i(s_j) = c_i(s_j)$ . Полагаем, что критерий эффективности прохождения пакета заданий представляет собой сепарабельную функцию

$$f(\bar{s}) = \sum_{i=1}^n f_i(s_j), \quad s_j \in S_i, j \in \{1, \dots, N\}, \bar{s} \in S. \quad (1)$$

Задается ограничение  $g^*$  на суммарные затраты для выполнения всего пакета заданий. В частности, для того чтобы не допустить монополизации использования какого-либо ресурса наиболее «богатыми» пользователями, вводится ограничение  $b^*$  на бюджет ВО – максимальное значение  $c(\bar{s}) = \sum_{i=1}^n c_i(s_j), s_j \in S_i, j \in \{1, \dots, N\}, \bar{s} \in S$ . Требование сбалансированности долей глобальных и локальных потоков заданий заставляет собственников вводить ограничение  $t^*$  на суммарное время занятия слотов. Ограничение  $g^*$  формируется динамично к началу текущего цикла планирования таким образом, что

$$g^* = \sum_{i=1}^n g_i^0(s_j), \quad (2)$$

где  $g_i^0(s_j)$  – функция уровня ресурсных затрат, учитывающая характеристики подходящих наборов слотов для каждого задания пакета.

Формальная постановка задачи выбора оптимальной комбинации слотов  $\bar{s}$  заключается в отыскании экстремума (1) при ограничении

$$\sum_{i=1}^n g_i(s_j) \leq g^*, \quad (3)$$

где  $g^*$  определяется в соответствии с (2).

Пример функции  $g_i^0(s_j)$  в (2) уровня временных затрат:

$$t_i^0(s_j) = [\sum_{s_j} t_i(s_j) / l_i], \quad (4)$$

где  $l_i$  – число подходящих (альтернативных) наборов слотов для  $i$ -го задания;  $[ \cdot ]$  – ближайшее не большее целое число.

Тогда ограничение (3) с учетом (4) задается следующим образом:

$$t^* = \sum_{i=1}^n t_i^0(s_j). \quad (5)$$

Пакет состоит из  $n$  независимых заданий, что допускает естественную декомпозицию решения задачи (1)-(3) на  $n$  этапов с использованием методов динамического программирования. Для этого необходимо ввести понятие состояния системы (пакета заданий). Его будем представлять уровнем  $g_i$  допустимых суммарных затрат на выполнение части заданий пакета, включая  $i$ -е задание, например заданий  $i, i+1, \dots, n$  либо заданий  $i, i-1, \dots, 1$ . В частности, в одном случае  $g_i$  может представлять собой суммарное время  $t_i$  занятия слотов заданиями  $i, i+1, \dots, n$  либо заданиями  $i, i-1, \dots, 1$ . В другом случае,  $g_i$  – суммарная стоимость  $c_i$  использования слотов заданиями  $i, i+1, \dots, n$  или  $i, i-1, \dots, 1$ . Тогда рекуррентные соотношения для отыскания экстремума (1) при ограничении (3) и использовании набора слотов  $s_j \in S_i, i = \overline{1, n}, j \in \{1, \dots, N\}$  по схеме обратной прогонки имеют следующий вид:

$$f_i(g_i) = \text{extr}_{s_j \in S_i} \{f_i(s_j) + f_{i+1}(g_i - g_i(s_j))\}, \quad g_i(s_j) \leq g_i \leq g^*, \quad i = \overline{1, n},$$

$$f_{n+1}(g_{n+1}) \equiv 0, \quad g_i = g_{i-1} - g_{i-1}(s_k), \quad 1 < i \leq n, \quad g_1 = g^*, \quad s_k \in S_{i-1}, \quad (6)$$

где  $g_i$  – суммарные затраты при использовании наборов слотов заданиями  $i, i+1, \dots, n$  пакета.

Соответствующие уравнения для схемы прямой прогонки:

$$f_i(g_i) = \text{extr}_{s_j \in S_i} \{f_i(s_j) + f_{i-1}(g_i - g_i(s_j))\}, \quad g_i(s_j) \leq g_i \leq g^*, \quad i = \overline{1, n},$$

$$f_0(g_0) \equiv 0, \quad g_i = g_{i+1} - g_{i+1}(s_k), \quad 1 \leq i < n, \quad g_n = g^*, \quad s_k \in S_{i+1}, \quad (7)$$

где  $g_i$  – суммарные затраты на выполнение заданий  $i, i-1, \dots, 1$  пакета.

Оптимальные затраты в схемах (6), (7) определяются из уравнения

$$g_i^*(s_j) = \arg \text{extr}_{g_i(s_j) \leq g_i} f_i(g_i), \quad i = \overline{1, n}. \quad (8)$$

Оптимальный набор слотов для выполнения  $i$ -го задания будем обозначать через  $s_i^* \in \{1, \dots, N\}$ . В обеих схемах (6) и (7) он задается соотношением

$$s_i^* = \arg \text{extr}_{s_j \in S_i} f_i(g_i^*(s_j)), \quad i = \overline{1, n}. \quad (9)$$

На основе функциональных уравнений (6)-(9) отыскивается решение задачи (1)-(3), позволяющее получить оптимальную комбинацию слотов  $(s_1^*, \dots, s_n^*)$  для пакета заданий.

Рассмотрим примеры стратегий планирования, имеющие важное практическое значение.

1. Максимизация доходов собственников ресурсов при заданном ограничении на суммарное время использования слотов в соответствии с (4), (5)

$$f_i(t_i) = \max_{s_j \in S_i} \{c_i(s_j) + f_{i+1}(t_i - t_i(s_j))\}, \quad i = 1, \dots, n, \quad f_{n+1}(t_{n+1}) \equiv 0. \quad (10)$$

2. Минимизация суммарного времени завершения пакета заданий при ограничении на бюджет  $b^*$  виртуальной организации

$$f_i(c_i) = \min_{s_j \in S_i} \{t_i(s_j) + f_{i+1}(c_i - c_i(s_j))\}, \quad i = 1, \dots, n, \quad f_{n+1}(c_{n+1}) \equiv 0. \quad (11)$$

3. Минимизация суммарной стоимости выполнения пакета заданий при ограничении на суммарное время (5) использования слотов

$$f_i(t_i) = \min_{s_j \in S_i} \{c_i(s_j) + f_{i+1}(t_i - t_i(s_j))\}, \quad i = 1, \dots, n, \quad f_{n+1}(t_{n+1}) \equiv 0. \quad (12)$$

4. Минимизация простоя ресурсов при ограничении на время (5) их использования

$$f_i(t_i) = \max_{s_j \in S_i} \{t_i(s_j) + f_{i+1}(t_i - t_i(s_j))\}, \quad i = 1, \dots, n, \quad f_{n+1}(t_{n+1}) \equiv 0. \quad (13)$$

Вышеприведенные функциональные уравнения (10)–(13) представляют собой конкретизацию (6) и реализуются в виде компонентов симулятора распределенной вычислительной среды [16].

ЦСП присущи следующие ограничения.

Отсутствует возможность влиять на ход выполнения отдельного пользовательского задания: поиск отдельных альтернатив осуществляется по принципу First Fit, а выбор их оптимальной комбинации согласно (9) отражает интересы всей ВО. Таким образом, не учитываются предпочтения отдельных пользователей, что не позволяет говорить о справедливом разделении ресурсов.

Планирование выполнения пакета заданий основывается на пользовательской, зачастую весьма неточной, оценке времени  $t_i(s_j)$  выполнения конкретного приложения. В случае несостоятельности оценки преждевременно высвобожденные ресурсы могут про-

стаивать, тем самым, понижая уровень загрузки среды. Для планирования пакета заданий необходимо выделение нескольких «непересекающихся» (по занятым ресурсам и времени) альтернатив, а для выполнения задания выбирается одна альтернатива. Это может приводить к фрагментации ресурсов.

Известный алгоритм обратного заполнения - бэкфиллинг [17], используемый, как правило, в параллельных системах, в ряде случаев позволяет ее предотвратить. Бэкфиллинг является модификацией метода FCFS и требует от пользователей оценки времени выполнения их заданий, что позволяет заранее, на этапе планирования резервировать ресурсы. С помощью механизма предварительного резервирования метод гарантирует получение ресурсов высокоприоритетными заданиями, и, вместе с тем, допускает нарушение порядка очереди, что способствует повышению коэффициента общей загрузки ресурсов. Нарушение порядка очереди происходит на этапе обратного заполнения, при попытке назначить на доступные ресурсы задания с меньшим приоритетом. При использовании консервативной вариации задание с меньшим приоритетом может быть запущено вне очереди только в том случае, если оно не будет препятствовать запуску всех более приоритетных заданий. Агрессивный бэкфиллинг позволяет запускать задания вне очереди лишь в том случае, если они не будут препятствовать запуску самого приоритетного задания. Бэкфиллинг имеет такие недостатки, как неэффективное использование ресурсов по критериям, отличным от времени старта (особенно при относительно невысоком уровне загрузки доступных ресурсов), а также невозможность принципиально повлиять на качество распределения, изменяя политики и критерии, действующие в рамках ВО. Тем не менее, целесообразно рассмотреть возможность использования бэкфиллинга в сочетании с ЦСП и различными стратегиями для предотвращения фрагментации ресурсов.

## 1.2. Требования к модели справедливого разделения ресурсов и модификация ЦСП

Сформулируем требования к модели планирования и справедливого разделения ресурсов (МПСР) среди участников ВО путем обеспечения следующих дополнительных возможностей.

Для администраторов ВО - управления процессом планирования за счет определения политики прохождения потока заданий. Для пользователей - влияния на план выполнения заданий путем введения критерия эффективности. Для владельцев - управления загрузкой собственных ресурсов установлением удельной стоимости их использования в течение интервала планирования.

МПСР представляет собой развитие и обобщение ЦСП. Для учета интересов пользователей в формат ресурсного запроса дополнительно вводится критерий оптимизации. В отличие от так называемых мягких ограничений [14], представляющих пользовательские предпочтения, критерий оптимизации, заданный в ресурсном запросе, в дальнейшем учитывается на этапе поиска альтернатив выполнения. Для выбора альтернатив, оптимальных по заданному критерию, используется алгоритм АЕР (Algorithm searching for Extreme Performance), подробно описанный в [18]. Таким образом, формирование набора альтернатив выполнения осуществляется с учетом интересов пользователей, согласно заданным критериям оптимизации отдельных заданий, а выбор оптимальной комбинации альтернатив осуществляется по критерию (1), формализующему соответ-

ствующий аспект политики ВО. Владельцы ресурсов получают возможность управлять прибылью и загрузкой вычислительных узлов за счет варьирования расписания доступности и запрашиваемой стоимости использования на интервале планирования.

Другое отличие МПСР от ЦСП состоит в алгоритме формирования системы заданий. МПСР предполагает разделение исходного пакета заданий на множество подпакетов и планирование каждого подпакета в отдельности на заданном интервале планирования. Необходимость процедуры «разрезания» пакета в МПСР особенно ярко проявляется при высоком уровне загрузки ресурсов среды. Согласно реализуемому в ЦСП механизму поиска множества альтернатив [16], при планировании пакета число найденных альтернатив может быть относительно невысоким (вплоть до единственной альтернативы на задание при высоком уровне загрузки доступных ресурсов). Это отражается на этапе выбора лучшей комбинации слотов (9) и на эффективности планирования всего пакета.

«Разрезание» пакета позволяет увеличить количество альтернатив для более приоритетных заданий, разнообразить по параметрам множество доступных альтернатив на этапе выбора оптимальной комбинации и, тем самым, повысить эффективность разделения ресурсов. При разбиении исходного пакета на  $n$  подпакетов, где  $n$  – общее количество заданий в пакете, алгоритм согласно приоритетам позволит найти лучшие наборы слотов для выполнения каждого из заданий в отдельности, не учитывая эффективность планирования пакета в целом. При «выделении» из исходного пакета лишь одного подпакета, результат будет идентичен применению ЦСП (см. рис. 1).

### 1.3. Комбинация циклического планирования и бэкфиллинга

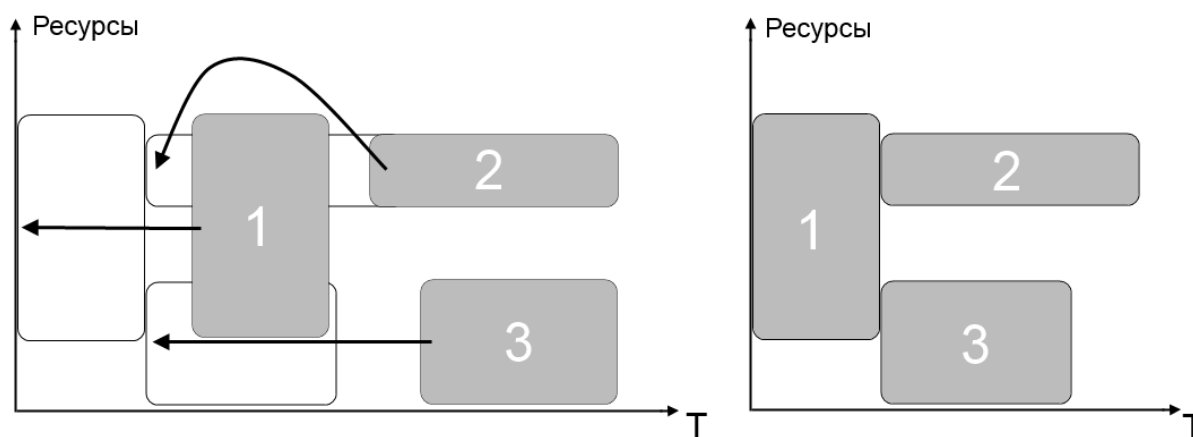
МПСР позволяет оптимизировать выполнение потока заданий согласно интересам участников ВО при условии, что найдены альтернативные наборы слотов для выполнения заданий. Бэкфиллинг реагирует на досрочное освобождение ресурсов и допускает перепланирование «на лету», что очень важно в условиях, когда оценка времени выполнения и действительное время выполнения задания могут существенно отличаться.

Предлагается комбинированный, эвристический подход. В каждом цикле планирования из исходного пакета выделяется подмножество приоритетных заданий, например, самых «дорогих» (по стоимости выполнения), либо самых требовательных к ресурсу (по времени выполнения). Эти задания группируются в отдельный подпакет и планируются, возможно, и без соблюдения дисциплины очереди. Далее планирование этого подпакета производится с помощью МПСР на основе актуального расписания загрузки среды на рассматриваемом интервале планирования. Планирование остальных, менее требовательных к ресурсам заданий осуществляется с помощью бэкфиллинга на основе динамично обновляющейся информации о реальной загрузке узлов. Таким образом, метод планирования с использованием бэкфиллинга (МПБ) совмещает в себе преимущества как МПСР, так и бэкфиллинга, а именно, оптимизируется выполнение наиболее трудоемких заданий, эффективно используются ресурсы, преимущественно соблюдается дисциплина очереди выполнения заданий и достигаются невысокие значения времени старта.

### 1.4. Эвристическая процедура сдвига

Следующий шаг в развитии ЦСП – применение эвристической процедуры сдвига (ЦСПС) альтернатив выполнения задания к началу интервала планирования с сохране-

нием назначения на ресурсы. Суть процедуры сдвига (ПС) состоит в следующем. При поиске множества альтернатив выполнения и последующем выборе их оптимальной комбинации, выбранные альтернативы могут располагаться на рассматриваемом интервале планирования произвольным образом (на рис. 2а выбранные для трех заданий альтернативы схематически представлены серыми прямоугольниками в плоскости «Ресурсы – Время»). Из-за этой особенности не только увеличивается среднее время старта заданий пакета (по сравнению, например, с бэкфилингом), но, также, происходит «дробление» слотов, образование автономных участков, в которых может быть затруднительно выделить необходимое для выполнения очередного задания количество ресурсов.



**Рис. 2.** Демонстрация процедуры сдвига: а) слева – исходное расположение альтернатив выполнения; б) справа – расположение альтернатив после применения процедуры сдвига

Процедура сдвига выполняется итеративно, последовательно для каждого задания. Очередность выборки заданий формируется согласно времени старта выбранных альтернатив: в первую очередь происходит попытка смещения альтернатив, имеющих наименьшее время старта. Такая очередность гарантирует, что при смещении любого задания, все более ранние задания уже будут смещены, и, таким образом, не будут занимать соответствующие узлы. Иначе возможна ситуация, что задание с меньшим временем старта и меньшим приоритетом заблокирует перемещение задания с более высоким приоритетом, а затем, в свою очередь будет смещено, освободив дополнительные слоты.

Попытка смещения заключается в проверке возможности выполнения задания на узлах, на которых уже было запланировано выполнение, в более ранний момент времени. В случае если узлы свободны – план выполнения задания переназначается на более раннее время старта (рис. 2б). При этом сохраняются все основные характеристики альтернативы, такие, например, как стоимость или общее время выполнения.

Применение процедуры сдвига позволит существенно уменьшить среднее время старта пакета заданий, планирование которого осуществлялось ЦСП или ее модификациями в условиях избыточного количества доступных ресурсов, за счет более плотного расположения заданий в начале интервала планирования.



## 2. Исследование подходов к справедливому разделению ресурсов

### 2.1. Постановка эксперимента

Для исследования предложенных подходов, а также сравнения с известными алгоритмами планирования использован симулятор распределенной вычислительной среды [16], дополненный компонентами поддержки МПСР. Для постановки экспериментов используются следующие общие настройки компонентов модели (в относительных единицах симулятора). Типовой размер цикла планирования принят равным 600 единицам времени. Количество узлов в ресурсном домене равно 24. Производительность узлов является случайной равномерно распределенной величиной  $p \in [2, 10]$ . Таким образом, самые производительные ресурсы ( $p = 10$ ) смогут выполнять задания примерно в два раза быстрее средних по производительности ( $p = 6$ ), а наименее производительные ( $p = 2$ ) – в три раза медленнее. При этом обеспечивается достаточный уровень разнообразия ресурсов, отличие наименее и наиболее производительных экземпляров не будет превышать одного порядка внутри конкретного домена. Равномерный закон распределения выбран из предположения, что в рассматриваемой системе планирования состав доменов формируется с помощью искусственного отбора ресурсов (фильтрации) по таким характеристикам, как тип, производительность, расположение и т.п. (жесткие ограничения в [5]). Удельная стоимость ресурса назначается на этапе ценообразования в зависимости от производительности и случайной величины «скидка/наценка», подчиняющейся нормальному распределению. Количество заданий на каждом цикле планирования равно 20. Максимальная стоимость, которую пользователь готов заплатить за выполнение задания, задается таким образом, чтобы «богатейшие» пользователи могли позволить себе использование «дорогих» ресурсов (рыночная стоимость + наценка в 60%), а «беднейшие» были вынуждены рассчитывать на скидки. Эти факторы предотвращают монополизацию наиболее производительных и, соответственно, самых дорогостоящих ресурсов.

### 2.2. Учет интересов пользователей

Учет интересов пользователей и анализ результатов планирования для отдельных заданий происходит на основе сравнения МПСР и исходной ЦСП. Планирование пакета заданий в рамках МПСР, поиск альтернатив в которой осуществляется по алгоритму АЕР [18], выполняется с учетом критерия, указанного в ресурсном запросе задания. Планирование пакета заданий по ЦСП предполагает выбор первой подходящей альтернативы без оптимизации.

Таблица 1 демонстрирует результаты планирования отдельных заданий в зависимости от заданного пользователем критерия оптимизации: времени старта и завершения задания, времени и стоимости выполнения задания (АЕР минимизирует значение заданного критерия). Выбор одного из четырех критериев происходит на этапе генерации пакета заданий случайно по равномерному закону распределения. Равномерный закон распределения выбран из соображения того, что невозможно выделить один критерий оптимизации для независимых пользователей. В последней строке табл. 1 приведены результаты планирования того же пакета заданий с помощью ЦСП. Проведено моделирование 5000 отдельных циклов планирования. Наилучшие значения по критериям вре-

мени старта и завершения заданий, а также по времени и стоимости выполнения имеют задания, для которых был задан соответствующий критерий оптимизации (столбец «Критерий»). Исключение составляет подход с минимизацией времени завершения. Задания, для которых был указан соответствующий критерий оптимизации, в среднем обеспечивают не только наименьшее время старта, но и наименьшее время завершения. В целом, использование критерия оптимизации в МПСР по сравнению с ЦСП при выполнении отдельных заданий позволяет более чем на 23% уменьшить время старта завершения заданий, на 21% уменьшить общее время выполнения и на 12% снизить стоимость выполнения. Среднее количество  $N_A$  альтернатив выполнения в табл. 1, найденных для заданий в одном цикле планирования, практически не зависит от критерия оптимизации. Среднее число заданий на каждую группу, имеющую один и тот же критерий оптимизации, в среднем равно 5, что согласуется с использованием равномерного распределения при выборе одного из четырех рассматриваемых критериев оптимизации для каждого из 20 заданий пакета.

Таблица 1

Учет интересов пользователей ВО.

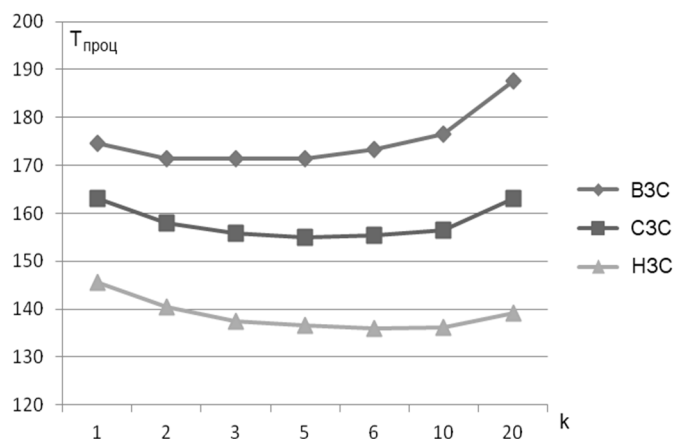
Критерий	$N_A$	Время старта	Время выполнения	Время завершения	Стоимость
Время старта	12,8	171,7	56,1	227,8	1281,1
Время выполнения	10,6	214,5	<b>39,3</b>	253,9	1278,5
Время завершения	12,2	<b>169,6</b>	45	<b>205,5</b>	1283,2
Стоимость	12,9	262,6	55,5	318	<b>1098,3</b>
ЦСП	12,1	222	50,3	272,3	1248,4

Из анализа результатов планирования отдельных заданий следует, что в рамках МПСР пользователи получают возможность влиять на эффективность выполнения собственных приложений путем введения соответствующего критерия оптимизации. Достигается это за счет того, что при наличии различных требований к эффективности выполнения заданий, ресурсы распределяются между заданиями неравномерно: в зависимости от используемого при отборе критерия. Заметим, что в исходной ЦСП на этапе поиска альтернатив все ресурсы распределяются между заданиями пакета равномерно.

### 2.3. Оптимизация выполнения пакета заданий

Следующий эксперимент посвящен сравнительному анализу результатов планирования при делении исходного пакета заданий в МПСР на подпакеты при различных уровнях загрузки среды. При выборе оптимальной комбинации альтернатив выполнения решалась задача минимизации среднего процессорного времени выполнения заданий  $T_{проц}$ . Процессорное время для альтернативы вычисляется как сумма длительностей слотов, входящих в сформированное «окно».

На рис. 3 представлены значения  $T_{проц}$  в зависимости от числа  $k \in \{1, 2, 3, 5, 6, 10, 20\}$  подпакетов, на которое делится исходный пакет заданий и уровня загрузки среды. Уровень загрузки при проведении серии экспериментов определяется средним относительным числом  $Y$  неудач – циклов планирования, в ходе которых не удается найти план выполнения для всех заданий пакета. Эксперименты проводились при высоком ( $Y = 0.3$ ), среднем ( $Y = 0.03$ ) и низком уровнях загрузки ( $Y < 0.0002$ ) (на рис. 3 – ВЗС, СЗС и НЗС соответственно). Таким образом, количество неудач в проводимых сериях экспериментов отличается как минимум на порядок.



**Рис. 3.** Среднее время  $T_{проц}$  выполнения заданий пакета в зависимости от количества  $k$  подпакетов

Анализ результатов показывает, что увеличение числа формируемых подпакетов обуславливает рост числа альтернатив для выполнения отдельного задания, уменьшение суммарной стоимости прохождения пакета заданий и увеличение относительного числа неудач  $Y$ . При увеличении уровня доступных ресурсов увеличивается число альтернатив для выполнения отдельного задания, уменьшается относительное количество неудач  $Y$  и уменьшается стоимость прохождения пакета заданий. Результаты экспериментов подтверждают, что разделение исходного пакета на подпакеты и их последовательное независимое планирование позволяет увеличить число альтернатив выполнения для заданий пакета и позволяет построить более эффективные планы. Так, при высокой загрузке вычислительной среды, наибольшую эффективность и наименьшее количество неудач обеспечивает разделение на меньшее количество подпакетов. С другой стороны, при низкой загрузке доступных ресурсов, целесообразно деление на большее количество подпакетов вплоть до планирования заданий по отдельности (см. рис. 3).

Другое преимущество МПСР, по сравнению с ЦСП, это – уменьшение стоимости прохождения пакета заданий при уменьшении уровня загрузки ресурсов. ЦСП стремится использовать весь доступный для выполнения пакета заданий бюджет  $b^*$ , выбирая соответствующую комбинацию альтернатив. В то же время при планировании подпакетов с небольшим числом заданий, выбор зачастую ограничен несколькими альтернативами, стоимость которых не обязательно будет близка к установленному ограничению на бюджет.

Таким образом, МПСР позволяет не только учитывать интересы администраторов ВО (за счет оптимизации на этапе выбора комбинации альтернатив, как и в ЦСП), но и

за счет разбиения на подпакеты способна обеспечивать лучшее значение целевого критерия по сравнению с исходной схемой.

## 2.4. Учет интересов владельцев

Таблица 2 демонстрирует результаты планирования в МПСР с точки зрения владельцев ресурсов на примере характеристик отдельного вычислительного узла в зависимости от удельной стоимости  $c$ , назначенной на интервале планирования  $T = 600$ . В табл. 2:  $L_c$  – суммарное процессорное время загрузки узла на интервале планирования;  $U$  – средняя относительная величина загрузки ресурса на рассматриваемом интервале планирования,  $P$  – средняя прибыль, полученная владельцем ресурса,  $Y$  – относительное количество неудачных попыток планирования.

Таблица 2

Учет интересов владельцев ресурсов ВО

$c$	$L_c$	$U$	$P$	$Y$
2	256,6	0,44	527,1	0
4	234,9	0,39	939,6	0,001
6	185,4	0,31	<b>1112,3</b>	0,013
8	109,8	0,18	878,7	0,024
10	71	0,12	710,3	0,025

Из табл. 2 видно, что владельцы ресурсов в МПСР могут управлять собственной прибылью  $P$  и уровнем загрузки  $U$  вычислительных узлов на интервале планирования  $T$  путем выставления удельной стоимости  $c$  использования узла. Экстремум прибыли достигается при выставлении стоимости, близкой к «среднерыночной», то есть средней стоимости за экземпляр ресурса аналогичной производительности, выставленной другими владельцами ресурсов.

## 3. Эффективность использования ресурсов

### 3.1. Комбинированный метод планирования

Эффективность планирования с помощью комбинированного подхода МПБ может быть рассмотрена одновременно с позиций: значения критерия оптимизации в конкретной ВО, например, минимизации общего времени выполнения пакета заданий (11), и уровня загрузки среды, а также минимизации времени старта заданий пакета.

На рис. 4 представлены среднее время выполнения  $T_{проц}$  и среднее время старта  $T_{старт}$  заданий пакета в зависимости от соотношения  $N_{цсп}/N$ , в котором происходит разделение  $N$  заданий на подпакеты,  $N_{цсп}$  – число заданий в первом подпакете, планирование которых осуществляется на основе ЦСП. При разделении на подпакеты используется простой приоритет – очередность заданий в пакете, характеристики самих заданий не принимаются во внимание. Представленные на рис. 4 результаты получены на основе моделирования 5000 независимых циклов планирования при среднем уровне загрузки ресурсов согласно настройкам, описанным в разделе 3. Из рис. 4 видно, что если планирование большей части задания осуществлять с помощью МПСР, достигается

лучшее значения целевого критерия ВО – времени выполнения  $T_{проц}$ , при этом среднее время старта  $T_{старт}$  заданий отодвигается. И наоборот, если планирование большей части заданий осуществлять с помощью бэкфиллинга, среднее время старта приближается к началу интервала планирования, но значение целевого критерия оптимизации ухудшается. Особое внимание стоит уделить точке пересечения рассматриваемых графиков  $T_{проц}$  и  $T_{старт}$  на рис. 4. Ее наличие при монотонности графиков позволяет говорить о возможности достижения компромисса между средним временем старта заданий и значением целевого критерия в ВО.

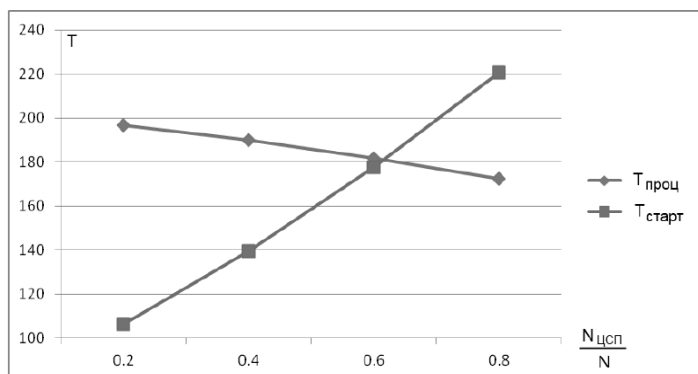


Рис. 4. Среднее время выполнения  $T_{проц}$  и старта  $T_{старт}$  заданий в МПБ

Таким образом, эвристика МПБ демонстрирует «компромиссные» значения рассматриваемых показателей по сравнению с МПСР и бэкфиллингом.

### 3.2. Состоятельность планов

МПСР и ЦСП позволяют строить предварительные планы выполнения пакета заданий на рассматриваемом интервале планирования, но не учитывают ситуаций, когда фактическое время выполнения заданий может оказаться меньше заявленного пользователем. Раннее завершение заданий и преждевременное освобождение и простой ресурсов могут негативно сказаться на эффективности выполнения пакета заданий по критериям, определенным участниками ВО, и сделать построенный план несостоятельным. С другой стороны, бэкфиллинг осуществляет планирование на основе динамически обновляемой информации о статусе выполнения заданий и загрузке вычислительных узлов и за счет этого может обеспечивать более эффективное прохождение потока заданий.

Для исследования и сравнения эффективности планов, построенных ЦСП, МПСР и бэкфиллингом было проведено имитационное моделирование, в котором реальное время выполнения заданий существенно отличается от времени предварительного резервирования ресурсов. Фактическое время выполнения заданий задавалось случайной величиной, подчиняющейся равномерному распределению на интервале  $[0.2 \cdot T_{рез}, T_{рез}]$ , где  $T_{рез}$  – время резервирования ресурсов, отведенное для выполнения задания. Равномерное распределение выбрано потому, что практически невозможно прогнозировать реальное время выполнения задания на выбранных ресурсах.

Таблица 3 содержит значения среднего времени выполнения (критерия оптимизации) и среднего времени старта заданий, полученные на этапе предварительного планирования на основе оценки времени выполнения заданий  $T_{рез}$  (колонка «Запланированное») и в результате моделирования выполнения составленного плана с учетом фактического времени выполнения заданий (колонка «Фактическое»). Из табл. 3 видно, что даже при значительной разнице между временем резервирования ресурсов и фактическим временем выполнения заданий преимущество МПСР перед бэкфиллингом по целевому критерию оптимизации в ВО не только сохраняется, но и увеличивается. Объясняется это тем, что бэкфиллинг не осуществляет оптимизации по критериям, отличным от времени старта, а более плотное расположение заданий (реальное время старта заданий, планирование которых осуществляется с помощью бэкфиллинга, сокращается в среднем на 46%) задействует практически все доступные ресурсы, в том числе менее предпочтительные по целевому критерию. Таким образом, результаты эксперимента показывают, что предварительно составленные планы являются состоятельными по критериям, заданным в ВО, в случае, когда фактическое время выполнения существенно отличается от времени резервирования ресурсов.

Таблица 3

Фактическое и запланированное время выполнения заданий

Алгоритм	Время выполнения		Среднее время старта заданий	
	Запланированное	Фактическое	Запланированное	Фактическое
Бэкфиллинг	187,7	115,1	<b>69</b>	<b>37,3</b>
ЦСП	150,1	90,4	281,2	281,2
МПСР	<b>138,6</b>	<b>83,5</b>	223,8	223,8
Преимущество МПСР перед бэкфиллингом	26,2%	27,5%	-69%	-83%

Следует отметить, что дополнительное преимущество может быть достигнуто за счет учета информации о текущей загрузке вычислительных узлов и перепланирования.

### 3.3. Влияние числа узлов на эффективность планирования

Эффективность планирования в зависимости от числа узлов вычислительной среды оценивается по следующим критериям: общее процессорное время  $T_{проц}$ , среднее время старта  $T_{старт}$  и завершения выполнения  $T_{заб}$  пакета заданий, а также комбинированный показатель  $F = T_{старт} + T_{проц}$ . Проведен сравнительный анализ результатов планирования с применением ЦСП и ЦСП с эвристикой сдвига, МПБ и бэкфиллинга.

На рис. 5 представлено среднее время старта заданий для рассматриваемых подходов в зависимости от количества узлов вычислительной среды. При увеличении количества доступных узлов бэкфиллинг позволяет уменьшить среднее время старта вплоть до нуля (то есть все задания стартуют с начала интервала планирования). При этом время старта заданий, при использовании ЦСП практически не зависит от уровня доступных

ресурсов: оптимальный набор альтернатив находится примерно в середине интервала планирования. МПБ ожидаемо позволяет обеспечить время старта близкое к бэкфиллингу за счет заполнения неиспользованных циклической схемой слотов наименее приоритетными заданиями.

При большом количестве ресурсов время старта заданий в схеме ЦСПС стремится к ненулевому значению, так как при выборе оптимального набора альтернатив наиболее эффективные по критерию оптимизации ресурсы, как правило, используются более, чем одним заданием. Таким образом, в условиях разнородных ресурсов обеспечить старт всех заданий с самого начала интервала планирования практически невозможно.

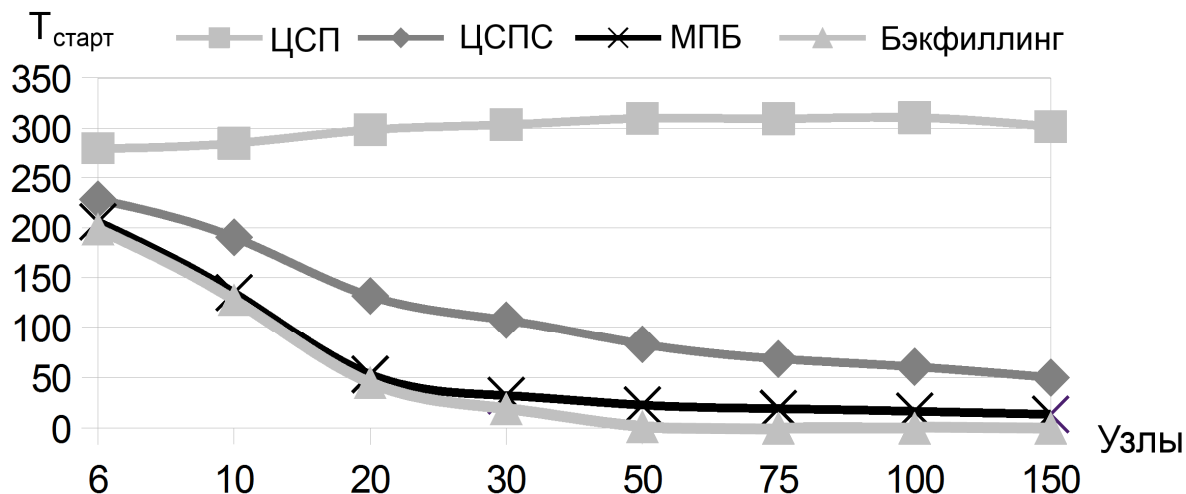


Рис. 5. Среднее время старта заданий пакета

Рис. 6 демонстрирует преимущество ЦСП и МПБ перед бэкфиллингом по целевому критерию оптимизации в ВО – процессорному времени выполнения заданий  $T_{\text{проц}}$ . Следует отметить, что с увеличением числа узлов преимущество ЦСП, ЦСПС и МПБ над бэкфиллингом становится больше. Применение дополнительной эвристики разделения на подпакеты может обеспечить еще большее преимущество циклической схемы по данному критерию.

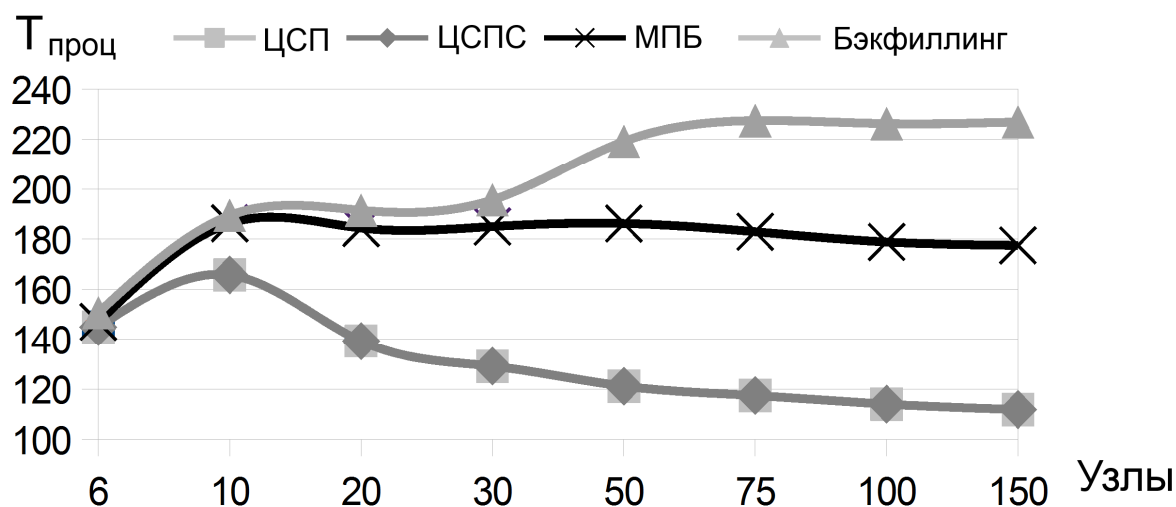


Рис. 6. Среднее процессорное время выполнения заданий пакета

На рис. 7 показана зависимость комбинированного показателя  $F = T_{\text{старт}} + T_{\text{проц}}$  эффективности использования ресурсов от числа узлов.

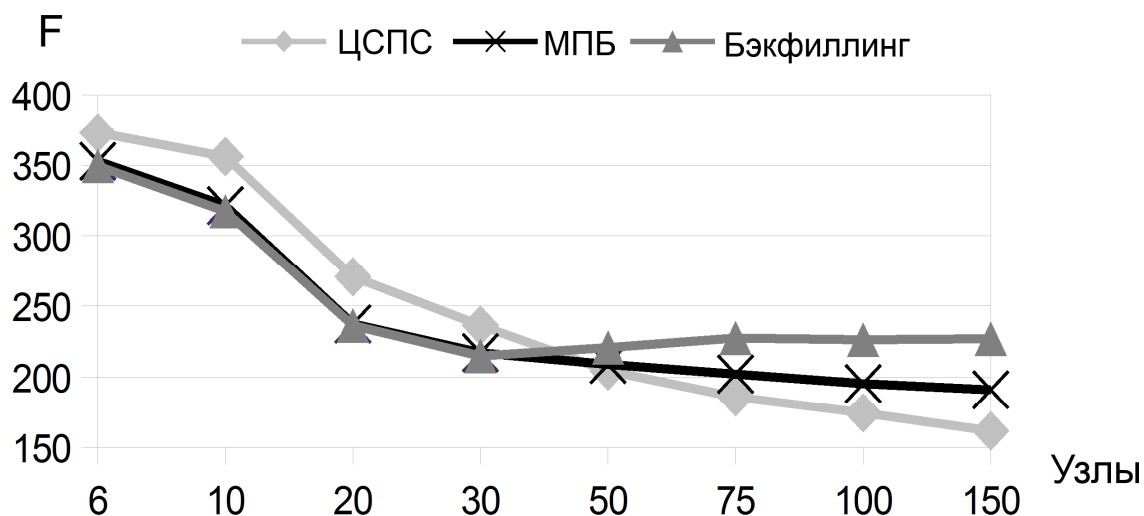


Рис. 7. Среднее значение  $F$  для заданий пакета

Важно отметить, что из пересечения графиков зависимостей для ЦСПС, МПБ и бэкфиллинга следует, что при относительно небольшом количестве доступных ресурсов целесообразнее может быть использование бэкфиллинга или МПБ (они обеспечивают примерно одинаковое значение показателя), однако при увеличении масштабов вычислительной среды, более эффективным становится подход с использованием ЦСП.

Аналогичный вывод можно сделать, если оценивать эффективность использования ресурсов на основе среднего времени завершения заданий (рис. 8).

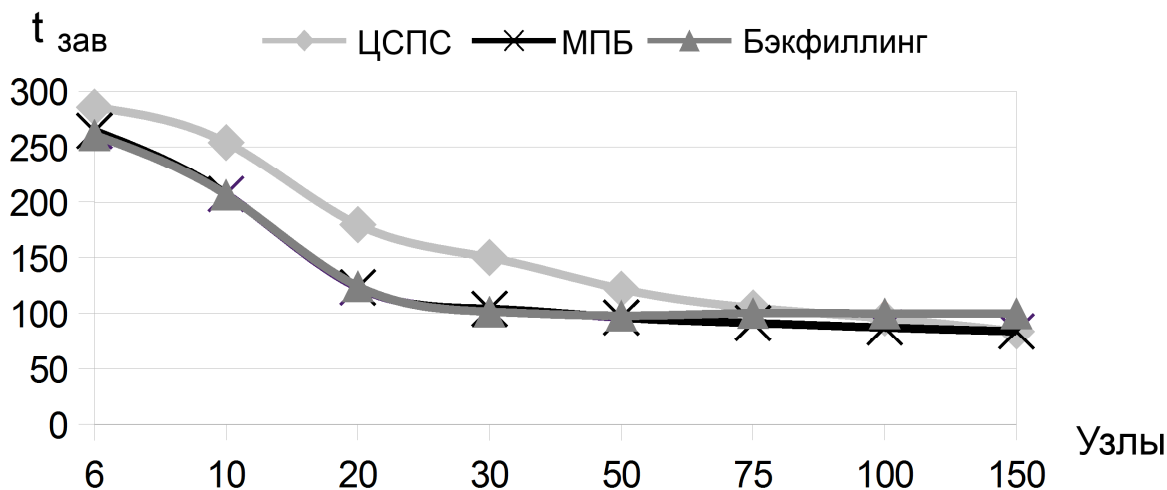


Рис. 8. Среднее время завершения заданий пакета

## Заключение

В работе рассмотрены стратегии и эвристики планирования с различными целевыми критериями, основанные на модели справедливого разделения ресурсов, учитывающей предпочтения всех участников ВО. Предложен подход к решению задачи справедливого разделения ресурсов между участниками ВО. Показано преимущество МПБ по срав-



нению с исходной циклической схемой планирования. Владельцы для получения максимальной прибыли или необходимого уровня загрузки среды могут варьировать удельную стоимость предоставления ресурсов (например, предлагать скидки). На основе сочетания циклической схемы и бэкфиллинга предложен комбинированный подход (эвристика МПБ), который демонстрирует компромиссные результаты по сравнению с МПСР и бэкфиллингом. Удастся эффективно использовать доступные ресурсы (за счет применения бэкфиллинга) при эффективном выполнении части потока заданий в ВО (за счет оптимизации в МПСР). Показана состоятельность планов, составленных МПСР в условиях, когда фактическое время выполнения заданий может существенно отличаться от пользовательской оценки.

Полученные результаты используются в проекте создания прототипа метапланировщика – программного комплекса планирования потоков заданий и управления неотчуждаемыми ресурсами в распределенных вычислительных средах, основные компоненты которого представлены в [19–21].

*Работа выполнена при содействии Совета по грантам Президента Российской Федерации для поддержки ведущих научных школ (шифр НШ-362.2014.9), РФФИ (проект 12-07-00042), Минобрнауки России, задание № 2014/123 на выполнение государственных работ в сфере научной деятельности в рамках базовой части государственного задания (проект 2268).*

## Литература

1. Garg S.K., Buyya R., Siegel H.J. Scheduling Parallel Applications on Utility Grids: Time and Cost Trade-off Management // 32nd Australasian Computer Science Conference, Wellington, New Zealand, Proceedings. 2009. P. 151-159.
2. Degabriele J.P., Pym D. Economic Aspects of a Utility Computing Service, Bristol, July 3, 2007. 23 p. (Technical Report, Trusted Systems Laboratory, HP Laboratories, HPL-2007-101).
3. Garg S.K., Yeo C.S., Anandasivam A., Buyya R. Environment-conscious Scheduling of HPC Applications on Distributed Cloud-oriented Data Centers // J. Parallel and Distributed Computing. 2011. Vol. 71, No. 6. P. 732-749.
4. Воеводин Вл.В., Жолудев Ю.А., Соболев С.И., Стефанов К.С. Эволюция системы метакомпьютинга X-Com // Вестник Нижегородского университета им. Н.И. Лобачевского. 2009. № 4. С. 157–164.
5. Kurowski K., Nabrzyski J., Oleksiak A., Weglarz J. Multicriteria Aspects of Grid Resource Management. In: Nabrzyski, J., Schopf, J.M., Weglarz J. (eds.). Grid resource management. State of the art and future trends. Kluwer Academic Publishers, Boston, 2003. P. 271–293.
6. Garg S.K., Konugurthi P., Buyya R. A Linear Programming-driven Genetic Algorithm for Meta-scheduling on Utility Grids // J. Par., Emergent and Distr. Systems. 2011. Vol. 26. P. 493–517.
7. Buyya R., Abramson D., Giddy J. Economic Models for Resource Management and Scheduling in Grid Computing // J. Concurrency and Computation. 2002. Vol. 14, No. 5. P. 1507–1542.

8. Ernemann C., Hamscher V., Yahyapour R. Economic Scheduling in Grid Computing // Feitelson, D.G., Rudolph, L., Schwiegelshohn, U. (eds.), JSSPP 2002. LNCS, Vol. 2537, Springer, Heidelberg, 2002. P. 128–152.
9. Lee Y.C., Wang C., Zomaya A.Y., Zhou B.B. Profit-driven Scheduling for Cloud Services with Data Access Awareness // J. Par. and Distr. Computing. 2012. Vol. 72, No. 4. P. 591–602.
10. Toporkov V.V. Job and Application-Level Scheduling in Distributed Computing // Ubiquitous Computing and Communication J. Applied Computing. 2009. Vol. 4, No. 3. P. 559 – 570.
11. Toporkov V.V., Toporkova A., Tselishchev A., Yemelyanov D. Job and Application-Level Scheduling: an Integrated Approach for Achieving Quality of Service in Distributed Computing // 4th International Conference on Dependability of Computer Systems, Proceedings. IEEE CS Press, Los Alamitos, 2009. P. 202 – 209.
12. Aida K., Casanova H. Scheduling Mixed-parallel Applications with Advance Reservations // 17th IEEE Int. Symposium on HPDC, Proceedings. IEEE CS Press, New York. 2008. P. 65–74.
13. Ando S., Aida K. Evaluation of Scheduling Algorithms for Advance Reservations // Information Processing Society of Japan SIG Notes. HPC-113, 2007. P. 37-42.
14. Elmroth E., Tordsson J. A Standards-based Grid Resource Brokering Service Supporting Advance Reservations, Coallocation and Cross-Grid Interoperability // J. of Concurrency and Computation. 2009. Vol. 25, No. 18. P. 2298–2335.
15. Toporkov V. Packet Task Processing in Distributed Computational Environments with Inalienable Resources // Automation and Remote Control, 2012. Vol. 73, No. 10. P. 1632–1647.
16. Toporkov V., Tselishchev A., Yemelyanov D., Bobchenkov A. Dependable Strategies for Job-flows Dispatching and Scheduling in Virtual Organizations of Distributed Computing Environments // Complex Systems and Dependability, Proceedings. AISC, vol. 170, Springer, Heidelberg, 2012. P. 240–255.
17. Jackson D., Snell Q., Clement M. Core Algorithms of the Maui Scheduler // LNCS, vol. 2221, Springer, Heidelberg, 2001. P. 87-102.
18. Toporkov V., Toporkova A., Tselishchev A., Yemelyanov D. Slot Selection Algorithms for Economic Scheduling in Distributed Computing with High QoS Rates // New Results in Dependability and Computer Systems. AISC, vol. 224, Springer, Heidelberg, 2013. P. 459-468.
19. Свидетельство о государственной регистрации программы для ЭВМ № 2013613799 «Модуль комбинированного планирования потока заданий «Batch Slice Filling». Правообладатель: НИУ «МЭИ» (RU). Авторы: Топорков В.В., Емельянов Д.М. 16.04.2013 г.

Топорков Виктор Васильевич, д.т.н., заведующий кафедрой вычислительной техники, Национальный исследовательский университет «МЭИ» (Москва, Российская Федерация), ToporkovVV@mpei.ru

Бобченков Александр Викторович, к.т.н., младший научный сотрудник кафедры вычислительной техники, Национальный исследовательский университет «МЭИ» (Москва, Российская Федерация), groddenator@gmail.com

Емельянов Дмитрий Михайлович, к.т.н., ассистент кафедры вычислительной техники, Национальный исследовательский университет «МЭИ» (Москва, Российская Федерация), Yemelyanov.Dmitry@gmail.com

Целищев Алексей Сергеевич, к.т.н., Европейская организация по ядерным исследованиям (CERN) (Женева, Швейцария), Alexey.Tselishchev@cern.ch

*Поступила в редакцию 9 апреля 2014 г.*

---

*Bulletin of the South Ural State University  
Series "Computational Mathematics and Software Engineering"  
2014, vol. 3, no. 2, pp. 43–62*

---

## SCHEDULING METHODS AND HEURISTICS IN DISTRIBUTED COMPUTING WITH NON-DEDICATED RESOURCES

*V. V. Toporkov*, Moscow Power Engineering Institute (Moscow, Russian Federation),  
*A. V. Bobchenkov*, Moscow Power Engineering Institute (Moscow, Russian Federation),  
*D. M. Yemelyanov*, Moscow Power Engineering Institute (Moscow, Russian Federation),  
*A. S. Tselishchev*, European Organization for Nuclear Research (CERN) (Geneva, Switzerland)

The paper is devoted to the questions of effective computations scheduling in virtual organizations with non-dedicated resources, including utility Grid. A complex combination of job-flow and application-level scheduling methods and heuristics, providing virtual organization resources share and allocation policies based on the economic principles is studied.

*Keywords: distributed computing, scheduling, optimization, slot, alternative, cyclic scheduling scheme.*

### References

1. Garg S.K., Buyya R., Siegel H.J. Scheduling Parallel Applications on Utility Grids: Time and Cost Trade-off Management // 32nd Australasian Computer Science Conference, Wellington, New Zealand, Proceedings. 2009. P. 151-159.
2. Degabriele J.P., Pym D. Economic Aspects of a Utility Computing Service, Bristol, July 3, 2007. 23 p. (Technical Report, Trusted Systems Laboratory, HP Laboratories, HPL-2007-101).
3. Garg S.K., Yeo C.S., Anandasivam A., Buyya R. Environment-conscious Scheduling of HPC Applications on Distributed Cloud-oriented Data Centers // J. Parallel and Distributed Computing. 2011. Vol. 71, No. 6. P. 732-749.
4. Voevodin V.I., Zholudev Y.A., Sobolev S.I., Stefanov K.S. Evoluzia sistemy metacomputinga X-Com [X-Com metacomputing system evolution] // Publishing of the N. I. Lobachevsky State University of Nizhny Novgorod 2009. No. 4. P. 157–164.
5. Kurowski K., Nabrzyski J., Oleksiak A., Weglarz J. Multicriteria Aspects of Grid Resource Management. In: Nabrzyski, J., Schopf, J.M., Weglarz J. (eds.). Grid resource management. State of the art and future trends. Kluwer Academic Publishers, Boston, 2003. P. 271–293.

6. Garg S.K., Konugurthi P., Buyya R. A Linear Programming-driven Genetic Algorithm for Meta-scheduling on Utility Grids // J. Par., Emergent and Distr. Systems. 2011. Vol. 26. P. 493–517.
7. Buyya R., Abramson D., Giddy J. Economic Models for Resource Management and Scheduling in Grid Computing // J. Concurrency and Computation. 2002. Vol. 14, No. 5. P. 1507–1542.
8. Ernemann C., Hamscher V., Yahyapour R. Economic Scheduling in Grid Computing // Feitelson, D.G., Rudolph, L., Schwiegelshohn, U. (eds.), JSSPP 2002. LNCS, Vol. 2537, Springer, Heidelberg, 2002. P. 128–152.
9. Lee Y.C., Wang C., Zomaya A.Y., Zhou B.B. Profit-driven Scheduling for Cloud Services with Data Access Awareness // J. Par. and Distr. Computing. 2012. Vol. 72, No. 4. P. 591–602.
10. Toporkov V.V. Job and Application-Level Scheduling in Distributed Computing // Ubiquitous Computing and Communication J. Applied Computing. 2009. Vol. 4, No. 3. P. 559 – 570.
11. Toporkov V.V., Toporkova A., Tselishchev A., Yemelyanov D. Job and Application-Level Scheduling: an Integrated Approach for Achieving Quality of Service in Distributed Computing // 4th International Conference on Dependability of Computer Systems, Proceedings. IEEE CS Press, Los Alamitos, 2009. P. 202 – 209.
12. Aida K., Casanova H. Scheduling Mixed-parallel Applications with Advance Reservations // 17th IEEE Int. Symposium on HPDC, Proceedings. IEEE CS Press, New York. 2008. P. 65–74.
13. Ando S., Aida K. Evaluation of Scheduling Algorithms for Advance Reservations // Information Processing Society of Japan SIG Notes. HPC-113, 2007. P. 37-42.
14. Elmroth E., Tordsson J. A Standards-based Grid Resource Brokering Service Supporting Advance Reservations, Coallocation and Cross-Grid Interoperability // J. of Concurrency and Computation. 2009. Vol. 25, No. 18. P. 2298–2335.
15. Toporkov V. Packet Task Processing in Distributed Computational Environments with Inalienable Resources // Automation and Remote Control, 2012. Vol. 73, No. 10. P. 1632–1647.
16. Toporkov V., Tselishchev A., Yemelyanov D., Bobchenkov A. Dependable Strategies for Job-flows Dispatching and Scheduling in Virtual Organizations of Distributed Computing Environments // Complex Systems and Dependability, Proceedings. AISC, vol. 170, Springer, Heidelberg, 2012. P. 240–255.
17. Jackson D., Snell Q., Clement M. Core Algorithms of the Maui Scheduler // LNCS, vol. 2221, Springer, Heidelberg, 2001. P. 87-102.
18. Toporkov V., Toporkova A., Tselishchev A., Yemelyanov D. Slot Selection Algorithms for Economic Scheduling in Distributed Computing with High QoS Rates // New Results in Dependability and Computer Systems. AISC, vol. 224, Springer, Heidelberg, 2013. P. 459-468.

*Received 9 April 2014.*