Математическая формализация задач проектного планирования в расширенной постановке

[1]А.С. Аничкин <anton.anichkin@ispras.ru>
[1,2]В.А. Семенов <sem@ispras.ru>
[1]Институт системного программирования РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
[2]Московский физико-технический институт,
141700, Московская область, г. Долгопрудный, Институтский пер., 9

Аннотация. Задачи теории расписаний и проектного планирования находят широкое применение в научных и индустриальных областях. В статье обсуждаются возможности обобщенной математической постановки задач проектного планирования и их эффективного решения эвристическими алгоритмами полиномиальной сложности.

Ключевые слова: теория расписаний; проектное планирование.

DOI: 10.15514/ISPRAS-2017-29(2)-9

Для цитирования: Аничкин А.С., Семенов В.А. Математическая формализация задач проектного планирования в расширенной постановке. Труды ИСП РАН, том 29, вып. 2, 2017 г., стр. 231-256. DOI: 10.15514/ISPRAS-2017-29(2)-9

1. Введение

Теория расписаний и методы проектного планирования находят широкое применение в научных и индустриальных областях, связанных с управлением производством, организацией транспортных потоков, управлением вычислительными ресурсами. Однако многообразие существующих и перманентное появление новых математических моделей приводят к необходимости обобщения классов прикладных задач и применения универсальных подходов к их решению. Необходимость подобных обобщений возникает, в частности, при создании перспективных систем календарно-сетевого планирования индустриальных проектов, в которых задачи составления расписаний решаются в расширенной постановке с учетом разнообразных факторов, влияющих на ход выполнения проектных работ. В

такой постановке учитываются не только типовые отношения предшествования между работами и простые ресурсные ограничения, но и директивные сроки, рабочие календари, условия финансового и логистического обеспечения проектных работ, специфические требования их пространственно-временной согласованности. Примерами подобных требований могут служить особенности монтажа элементов конструкций возводимого сооружения, условия резервирование рабочих зон при организации проектных работ, правила размещения и использования оборудования. Учет всех перечисленных выше факторов крайне важен для масштабных индустриальных программ, в которых риски организационных и технологических ошибок чрезвычайно высоки, а сроки и бюджеты жестко ограничены.

Задачи теории расписаний обычно описываются общепринятой нотацией Грэхема $\alpha|\beta|\gamma$, которая представляет собой комбинацию трёх основных характеристик, определяющих тип условий задачи. Кратко поясним ее, используя привычные для теории расписаний термины «задание», «операция» и «машина». В проектном планировании для этих понятий применяются термины «составная работа», «работа» и «ресурс».

Первая характеристика α описывает модель операций и модель машин (модель исполнения работ и модель ресурсов для проектного планирования). Предопределенные значения задают, например, постановки «рабочего цеха», «потоковой линии», «открытой линии». Дополнительный параметр обычно определяет число машин. В ресурсном планировании он применяется также для уточнения общего количества доступных ресурсов и их типа как возобновимых, невозобновимых, частично возобновимых, логистических или непрерывно разделяемых.

Вторая характеристика β описывает модель исполнения операций. Она обычно задается одним или несколькими значениями, которые определяют допустимость прерываний, наличие отношений предшествования, задание директивных сроков на начало и завершение операций, а также возможность пакетирования операций.

Наконец, третья обязательная характеристика γ определяет целевую функцию, минимум которой должен достигаться на составленном расписании. При поиске допустимого расписания целевая функция может использоваться для оценки качества найденного приближенного решения. Типовые значения γ соответствуют целевой функции для минимизации времени завершения всех операций, наибольшей задержки завершения операции, общего взвешенного времени завершения операций, общего взвешенного запаздывания операций и общих взвешенных опережений и запаздываний операций. Выбор целевых функций в задачах проектного планирования существенно шире, поскольку оптимизационная задача может ставиться как задача выравнивания используемых ресурсов или как задача минимизации общей или приведенной стоимости проекта. Допустим выбор целевых функций, направленных на достижение многокритериальных показателей проекта.

Примечательно, что задачи проектного планирования RCPSP (Resource-Constrained Project Scheduling Problem) в значительной степени обобщают постановки теории расписаний и поэтому алгоритмы проектного планирования часто рассматриваются в качестве универсальных инструментов для их решения. Тем не менее, в зависимости от индивидуальных характеристик задачи вычислительная сложность составления расписания может существенно варьироваться и поэтому для частных постановок обычно применяют специальные алгоритмы. Например, задача проектного планирования RCPSP, являющаяся NP-полной, редуцируется к частным постановкам «открытой линии», «рабочего цеха» или «потоковой линии», имеющим полиномиальную сложность при небольшом числе машин, простых моделях обслуживания и отсутствии директивных сроков.

Вместе с тем, на практике возникает необходимость в обобщении задач проектного планирования, обусловленная особенностями осуществления проектной деятельности разными организациями и индустриальными сообществами, различиями в представлении проектных данных, а также альтернативными способами математической формализации задач проектного планирования. Большое число существующих программных приложений для календарно-сетевого планирования и управления проектами с близкими функциями подтверждает этот факт. В качестве таких приложений следует указать популярные программные системы Oracle Primavera, MS Project, Synchro, Spider Project, Gemini, Merlin, Zoho Projects, ManagePro, обеспечивающие автономную работу планировщиков и управленческого персонала на изолированных компьютерах или групповую работу в корпоративной сети. Ряд систем конфигурируется в виде универсальных Интернет-сервисов и web-клиентов к ним. К подобным решениям относятся сервисы Smartsheet, GanttPro, Asana, Acunote, Teamweek, Bitrix24, Jira, ISETIA. Примечательно, что почти все приложения реализуют алгоритмы для поиска критического пути или вычисления критических работ, а большая часть приложений — алгоритмы решения задач RCPSP в расширенных постановках. В типовых приложениях речь идет о проектных планах, содержащих десятки тысяч работ, поэтому для решения задач RCPSP применяются приближенные, основанные на эвристиках алгоритмы полиномиальной сложности. Применение точных методов при составлении оптимальных расписаний для проектных планов с числом работ выше сотни не представляется возможным из-за экстремально высокой вычислительной сложности задач RCPSP.

В представленной работе предпринимается попытка математического обобщения и формализации задач проектного планирования с учетом особенностей расширенных постановок RCPSP, встречаемых в программных приложениях. Ожидается, что полученные результаты позволят разработать программно-инструментальную среду общего назначения, обеспечивающую задание условий и эффективное решение разнообразных классов задач проектного планирования. Развитые инструментальные возможности среды

позволят относительно просто адаптировать ее к новым постановкам прикладных задач.

В разделе 2 приводится классическая постановка задач RCPSP, и кратко упоминаются алгоритмы их приближенного решения. Недостатки и ограничения математической постановки детально обсуждаются в разделе 3, в котором особое внимание уделяется вопросам структуризации проектного плана и моделям исполнения работ. Кратко обсуждаются вопросы математической формализации, связанные с заданием альтернативных целевых функций оптимизационной задачи и возможным переопределенным характером системы наложенных алгебраических ограничений. Раздел 4 посвящен математической формализации обобщенной задачи проектного планирования GCPSP (Generally Constrained Project Scheduling Problem), а также доказательству некоторых утверждений о достаточных условиях существовании решения. В разделе 5 описывается алгоритм приближенного решения задач GCPSP, который можно рассматривать в качестве развития известного алгоритма последовательной диспетчеризации. Формулируется утверждение об эквивалентности алгоритмов в случаях, когда обобщенная задача редуцируется к классической постановке RCPSP. В заключении кратко резюмируются результаты и определяются возможные направления для дальнейших исследований.

2. Классическая постановка RCPSP

Рассмотрим классическую постановку задач RCPSP, следуя работам [[1], [2], [3]].

Пусть проект состоит из N работ, R возобновимых ресурсов и L связей с соответствующими индексами $i = 1,2,\dots,N$, $r = 1,2,\dots,R$ и $l = 1,2,\dots,L$ соответственно. Пусть задано время старта проекта t_0, а для каждой работы i определено ее время выполнения $d_i \geq 0$ и количества потребляемых ей ресурсов $q_{i,r} \geq 0$. Требуется найти расписание проекта (время старта каждой работы x_i, $i = 1,2,\dots,N$), при котором минимизируется время выполнения всего проекта.

Уточним условия задачи. Потребление работой i ресурса r в количестве $q_{i,r}$ означает, что с началом выполнения работы данное количество ресурса становится недоступным для использования в других работах, а с ее окончанием — высвобождается. Одна работа может потреблять несколько разных ресурсов. Один ресурс может одновременно использоваться несколькими работами при условии, что его суммарное потребление не превышает доступное количество $Q_r > 0$. Пусть $I(t,r) = \{i = 1,..,n \mid x_i \leq t < x_i + d_i \;\&\; q_{i,r} \neq 0\}$ — множество индексов работ, выполняемых в момент времени $t \geq t_0$ и использующих ресурс r. Тогда условие корректного потребления ресурса r и необходимое условие существования решения задачи может быть выражено следующим образом:

$$\sum_{i \in I(t,r)} q_{i,r} \le Q_r$$

Наконец, связи определяют бинарные отношения предшествования между работами таким образом, что работа-последователь не может стартовать до того, как работа-предшественник завершится. Пусть $i_{(l)}$ — индекс работы-предшественника, участвующего в связи l, а $i^{(l)}$ — индекс работы-последователя, участвующего в связи l. Тогда данное отношение имеет вид $x_{i_{(l)}} + d_{i_{(l)}} \le x_{i^{(l)}}$. В дальнейшем будем записывать $i \to j$, если существует связь l, такая что $i = i_{(l)}$ и $j = i^{(l)}$.

Тогда задача RCPSP формализуется следующим образом:

$$\min\left(\max_{i=1,\dots N}(x_i + d_i)\right)$$

$$\text{при } x_i \ge t_0, i = 1, \dots, N$$

$$x_{i_{(l)}} + d_{i_{(l)}} \le x_{i^{(l)}}, l = 1,2, \dots, L$$

$$\forall\, t \ge t_0 \text{ имеет место } \sum_{i \in I(t,r)} q_{i,r} \le Q_r, r = 1,2, \dots, R$$

Неизвестные переменные в постановке задачи обычно целочисленные, поскольку на практике обычно используется дискретное представление временных параметров. Задача считается корректно поставленной, если заданные связи не образуют циклов и уровни потребления ресурсов индивидуальными работами не превышают их общедоступное количество. В противном случае система алгебраических ограничений может оказаться переопределенной, а задача — не иметь решений. В случае отсутствия ресурсных ограничений задача RCPSP естественным образом редуцируется к задаче поиска критического пути или вычисления критических работ.

На сегодняшний день существует несколько популярных алгоритмов приближенного решения задач RCPSP. Наиболее известным является последовательный алгоритм составления расписания, иногда называемый алгоритмом последовательной диспетчеризации [[3], [4]]. Он позволяет за фиксированное количество шагов построить согласованное расписание или убедиться в его отсутствии, если в проектном плане есть неразрешимые ограничения. На каждом шаге алгоритма выбирается одна из работ, предшественники которой уже обработаны и для них определены времена старта и завершения. Для выбранной на основе эвристических правил работы вычисляется наиболее ранее допустимое время старта и осуществляется переход к следующей работе. В предположении отсутствии циклов последовательность работ с обработанными предшественниками всегда существует, а для работ всегда может быть определено время старта, согласованное со всеми наложенными ограничениями.

В классическом варианте алгоритм предполагает использование множества обработанных работ S и множества необработанных работ D, все предшественники которых входят в S [[1]]. На каждом шаге алгоритма множества корректируются в соответствии с приведенным ниже псевдокодом:

Begin
  $S \coloneqq \emptyset$;
  While $|S| < N$ **do**
    $D \coloneqq \{i = 1, .., N | i \notin S \,\&\, j \in S\ \forall\, j \to i\}$;
    For $r \coloneqq 1$ **to** R **do**
      $K_r \coloneqq \left\{Q_r - \sum_{i \in I(\tau,r)} q_{i,r} \,\middle|\, \tau = t_0, t_0 + 1, \dots, T\right\}$;
    End for;
    $i^* \coloneqq \underset{i \in D}{\operatorname{argmax}}\{v(i)\}$;
    If $\nexists\, j \to i^*$ **then**
      $t_{min} \coloneqq t_0$;
    Else
      $t_{min} \coloneqq \max_{j \to i^*}(x_j + d_j)$;
    End if;
    $x_{i^*} \coloneqq \min\{t | t \ge t_{min} \,\&\, q_{i,r} \le K_r(\tau), \tau = t, t + 1, \dots, t + d_{i^*}, r = 1,2, \dots, R\}$;
    $S \coloneqq S \cup \{i^*\}$;
  End while;
End;

Здесь T — время, на протяжении которого составляется расписание, K_r — вспомогательное множество значений доступного ресурса в дискретные моменты времени, $v(i)$ — эвристическое правило, определяющее приоритет и очередность планирования работы i.

Алгоритм относительно прост и допускает многочисленные варианты, что делает его привлекательным для программной реализации. Один из вариантов заключается в попытке составить расписание в предположении отсутствия ресурсных ограничений и определить ранние и поздние даты выполнения работ, как это делается при поиске критического пути [[4]]. Затем проводится анализ ресурсных ограничений с учетом допустимых задержек индивидуальных работ. Можно ожидать, что для большей части работ ресурсные ограничения разрешатся таким способом. При этом следует предусмотреть возможность перезапустить или продолжить процесс составления расписания для оставшихся конфликтных работ и их последователей. Иначе применение алгоритма на практике может оказаться довольно рискованным. Другой известный вариант алгоритма основан на выделении групп несвязанных между собой работ и на их одновременном анализе [[1]]. Было установлено, что в ряде случаев он приводит к более качественным приближенным решениям. Однако реализация данного варианта сопряжена с дополнительными расходами на перераспределение работ по группам и согласование частных расписаний, что может оказаться критичным для больших проектных планов.

Все упомянутые выше алгоритмы имеют полиномиальную сложность и обеспечивают поиск приближенных решений для индустриально значимых задач, размерность которых может составлять десятки и сотни тысяч

неизвестных переменных. На практике часто требуется удостовериться в качестве получаемых решений, для чего желательно получить оптимальные решения, хотя бы для подобных задач низкой размерности. В этом случае можно применить точные комбинаторные алгоритмы с известными оптимизационными приемами типа «альфа-бета-отсечения» или «метода ветвей и границ» [[5]]. Например, лучший из известных точных алгоритмов Брукера за приемлемое время может решать задачи размерности не больше 60 [[6]]. По результатам сравнения приближенных и точных решений можно скорректировать параметры применяемых алгоритмов и настроить их эвристики с тем, чтобы обеспечить надлежащее качество результатов и при решении задач высокой размерности. К сожалению, теоретические исследования дают слишком грубые оценки, не позволяющие судить о качестве приближенных решений, полученных известных алгоритмами.

3. Ограничения классической постановки

Хотя приведенная математическая постановка RCPSP является классической для теории расписаний и прикладных дисциплин, связанных с календарно-сетевым планированием и управлением проектной деятельностью, она имеет ряд принципиальных ограничений для широкого практического использования. На необходимость решения задач проектного планирования в расширенной постановке указывают авторы работы [[3]]. В нашей обзорной статье [[7]] достаточно детально рассмотрены особенности прикладных задач, не учитываемые классической постановкой. В настоящем разделе мы неформально определяем обобщенный класс задач проектного планирования, который мог бы допускать альтернативные критерии и сложные модели исполнения работ, наложения связей, привлечения ресурсов, календарей, финансов при составлении расписаний. В конечном итоге подобные расширения приводят к новым видам целевых функций и функциональных ограничений для решаемых оптимизационных задач.

Опишем главные отличия, характеризующие обсуждаемый класс задач, более систематическим образом.

Во-первых, в нем допускается задание работ различных видов, таких как простые активности, вехи начала и завершения работ, «короткие гамаки», «длинные гамаки», а также составные работы, обеспечивающие иерархическую многоуровневую структуризацию проектного плана. При этом предполагается, что работы могут исполняться в обычном режиме, в режиме с прерываниями, с профильным использованием ресурсов, с учетом накладных временных затрат, в альтернативных мультимодальных режимах, а также с учетом компромиссов. При задании условий задачи может указываться также статус работ, как планируемых, стартованных, прерванных, возобновленных или завершенных, а также процент выполнения для незавершенных работ.

Во-вторых, взаимосвязи работ могут устанавливать отношения предшествования не только между окончанием предшествующей работы и

началом последующей работы, но также и между любыми событиями, связанными с их началом и завершением. Для формируемых взаимосвязей «начало-начало», «начало-окончание», «окончание-начало» и «окончание-окончание» могут быть установлены минимальные и максимальные величины задержки, пересчитанные в календарные даты с использованием рабочих календарей проекта. Примечательно, что величины задержек могут отрицательными, а взаимосвязи могут устанавливаться и для составных работ, обеспечивая тем самым возможность составления расписаний при крупноблочной структуризации проектного плана и при его последующей детализации. Важным аспектом составления расписания для актуализируемого проектного плана является учет нарушенных взаимосвязей и возможность восстановить отношения предшествования, хотя бы для оставшихся частей незавершенных работ. Подобные ситуации приводят к неоднозначной интерпретации ограничений и нуждаются в более строгой формализации.

В-третьих, исполнение работ и привлечение ресурсов в рамках проектной деятельности обычно осуществляется на основе календарей, определяющие графики работы. В обсуждаемых задачах предполагается использование модели календарей, в которой рабочие интервалы могут задаваться на основе регулярных и исключительных правил с требуемой точностью и дискретизацией представления временных параметров. При фиксировании трудоемкости работы и доминирующего ресурса ее продолжительность может определяться доступным количеством ресурса, а не только рабочим календарем.

В-четвертых, для работ могут быть заданы явные временные ограничения, определяющие алгебраические условия их начала и завершения. Ограничения могут задаваться условиями типа «начать не раньше», «завершить не позже», «начать в фиксированную дату» и т.п., так и спецификаторами «выполнить как можно раньше» или «выполнить как можно позже». Кроме того, каждой работе могут быть приписаны правила выравнивания, устанавливающие условия начала или завершения работы строго в начале или конце рабочего интервала (часа, дня, недели, месяца, года и т.п.) независимо от возможности исполнить работу несколько раньше или позже. Важной особенностью обсуждаемых задач является возможность задания временных ограничений для составных работ, что, как и в случае с взаимосвязями, обеспечивает возможность составления расписаний при крупноблочной структуризации проектного плана. Наконец, система наложенных алгебраических ограничений может оказаться переопределенной и не иметь решений. Это означает, что должен быть задан непротиворечивый способ разрешения подобных ситуаций, например, с использованием приоритетов, приписанных индивидуальным ограничениям.

В-пятых, ресурсная модель должна допускать использование невозобновимых, ограничено-возобновимых, частично возобновимых, логистических, непрерывно разделяемых, исключительных ресурсов, а также ресурсов с переменной доступностью. Возобновимые ресурсы обычно моделируют

использование персонала и техники, которые могут объединяться в «бригады». Ресурсы, связанные с персоналом определенной специализации и квалификации, могут формировать соответствующие «компетенции». Невозобновимые ресурсы обычно ассоциируют с расходными материалами, для которых могут определяться цепочки поставок. Финансовое обеспечение проектной деятельности также может моделироваться невозобновимыми ресурсами, ассоциируемыми, например, с банковскими счетами участвующих в проекте организаций. Количество доступных ресурсов может быть нефиксированным и зависеть от времени, например, как в случае поставок материалов или привлечения дополнительных инвестиций в ходе реализации проекта.

В-шестых, допускается выбор альтернативных целевых функций для минимизации временных показателей проекта, обеспечения консервативности и устойчивости расписания к задержкам, минимизации затрат на возобновимые ресурсы, минимизации невозобновимых ресурсов, минимизации общей стоимости проекта, максимизации чистой приведенной стоимости, а также достижения многокритериальных показателей проекта.

В-седьмых, задача проектного планирования может решаться в инкрементальной постановке, когда требуется скорректировать расписание с учетом измененных условий исходной задачи или при актуализации данных непосредственно на проектной площадке в режиме реального времени.

4. Математическая формализация задач GCPSP

Обсудим предлагаемый способ математической формализации задач проектного планирования. Следуя ему, оптимизационная задача ставится на множестве допустимых решений, связанных с частной задачей удовлетворения ограничений с приоритетами (или предпочтениями). В отличие от RCPSP мы не уточняем семантику неизвестных переменных, например, как времен начала, завершения, прерывания и возобновления работ или их продолжительностей. Также не конкретизируем вид целевой функции и функциональных ограничений, например, в виде явных временных или ресурсных условий. Вместо этого определяем обобщенный класс задач GCPSP в математически нейтральной форме в рамках общих предположений относительно свойств целевой функции, вида алгебраических ограничений и способа их разрешения относительно тех или иных переменных. Таким способом определяемый класс задач GCPSP расширяет условия классической постановки RCPSP и удовлетворяет основным требованиям к прикладным постановкам, перечисленным в предыдущем разделе.

Определение 1. Четверку множеств $\langle D, X, C, P\rangle$ назовем задачей в ограничениях с приоритетами, где $D = \{D_1 \times D_2 \times \ldots \times D_n\}$ — домен, определяющий область допустимых значений множества переменных задачи $X = \{x_1, x_2, \ldots, x_n\} \in D$, $x_i \in D_i$, $i = 1,2,\ldots,n$; $C = \{c_1(X_1), c_2(X_2), \ldots, c_m(X_m)\}$ — множество предикатов $c_j(X_j)$, $j = 1,2,\ldots,m$, определенных на подмножествах переменных $X_j = \left\{x_{i_1^{(j)}}, x_{i_2^{(j)}}, \ldots, x_{i_{k_j}^{(j)}}\right\} \subseteq X$ и называемых ограничениями задачи; $P = \{p_1, p_2, \ldots, p_m\}$ — множество натуральных чисел $p_j \in N$, называемых приоритетами ограничений. Пусть $I(X_j) = \{i_1^{(j)}, i_2^{(j)}, \ldots, i_{k_j}^{(j)}\} \subseteq \{1,2,\ldots,n\}$ — множество индексов подмножества переменных X_j. Тогда будем говорить, что переменная задачи x_i, $1 \le i \le n$ участвует в ограничении $c_j(X_j)$, $1 \le j \le m$ или ограничение ассоциировано с переменной, если $i \in I(X_j)$. Ограничение $c_j(X_j)$ будем называть удовлетворённым при любом множестве значений переменных $X_j^* \subseteq X^* \in D$, при котором соответствующий предикат принимает значение «верно».

Определение 2. Пусть $\langle D, X, C, P\rangle$ — задача в ограничениях с приоритетами. Значения переменных $X^* \in D$, при которых все ограничения задачи удовлетворены, будем называть решением задачи. Пусть $C^*(X)$ — множество ограничений, которые удовлетворены при значениях переменных X. Тогда X^* является решением задачи, только если $C^*(X^*) = C$.

Определение 3. Пусть $\langle D, X, C, P\rangle$ — задача в ограничениях с приоритетами. Значения переменных $X^* \in D$ будем называть согласованным решением задачи в ограничениях с приоритетами, если, по крайней мере, хотя бы одно ограничение удовлетворено и не существует других значений переменных Y, повышающих максимальный приоритет разрешенных ограничений: $C^*(X^*) \neq \emptyset$ и для любого неразрешенного ограничения $c_j \in C \backslash C^*(X^*)$, $p_j \in P$ не существует значений $Y = \{y_1, y_2, \ldots, y_n\} \in D$, $Y \neq X^*$, таких что удовлетворяется каждое ограничение $c_k \in C^*(Y)$, $p_k \in P$ с приоритетом $p_k \ge p_j$.

Определение 4. Пусть $\langle D, X, C, P\rangle$ — задача в ограничениях с приоритетами. Значения переменных $X^* \in D$ будем называть локально согласованным решением задачи в ограничениях с приоритетами, если, по крайней мере, хотя бы одно ограничение удовлетворено и согласованы приоритеты разрешенных ограничений относительно каждой переменной: $C^*(X^*) \neq \emptyset$, а также для любой переменной $x_i \in X$ и для любого ассоциированного с ней неудовлетворенного ограничения $c_j(X_j) \in C \backslash C^*(X^*)$, $p_j \in P$, $i \in I(X_j)$ не существует значения переменной $y_i \in D_i$, $y_i \neq x_i$ такого, что удовлетворяется каждое ассоциированное с переменной ограничение $c_k \in C^*(Y)$, $p_k \in P$, $i \in I(X_k)$ с приоритетом $p_k \ge p_j$, где множество значений переменных $Y = \{x_1^*, x_2^*, \ldots, y_i, \ldots, x_n^*\}$.

Определение 5. Обобщенной задачей проектного планирования GCPSP будем называть задачу оптимизации целевой функции проекта на множестве локально согласованных расписаний. Для определенности ограничимся следующей математической постановкой

$$\min f(X)$$

$$X \in D^*$$

где $f(X)\colon Z^n \to R$ — целевая функция, $D^* \subseteq Z^n$ — множество локально согласованных решений системы ограничений $c_j(X_j)$, $X_j = \left\{x_{i_1^{(j)}}, x_{i_2^{(j)}}, \dots, x_{i_{k_j}^{(j)}}\right\} \subseteq X$ с приоритетами $p_j \in N$, $j = 1,2,\dots,m$.

Предполагается, что ограничения разрешимы относительно переменных одним из нижеприведенных способов:

(1) $c_j(X_j) \Leftrightarrow x_{i_1^{(j)}} = g_j\left(x_{i_2^{(j)}}, x_{i_3^{(j)}}, \dots, x_{i_{k_j}^{(j)}}\right)$ (ограничения A)

(2) $c_j(X_j) \Leftrightarrow x_{i_1^{(j)}} \in D_j^+\left(x_{i_2^{(j)}}, x_{i_3^{(j)}}, \dots, x_{i_{k_j}^{(j)}}\right)$ (ограничения B)

(3) $c_j(X_j) \Leftrightarrow x_{i_2^{(j)}} \in D_j^+\left(x_{i_1^{(j)}}\right) \parallel x_{i_3^{(j)}} \in D_j^+\left(x_{i_1^{(j)}}\right) \parallel \dots \parallel x_{i_{k_j}^{(j)}} \in D_j^+\left(x_{i_1^{(j)}}\right)$ (ограничения C)

(4) $c_j(X_j) \Leftrightarrow$ для любого упорядочивания переменных ограничения X_j, представимого биекцией множеств индексов $\pi^{(j)}\colon \{1,2,\dots,k_j\} \to \left\{i_1^{(j)}, i_2^{(j)}, \dots, i_{k_j}^{(j)}\right\}$, имеет место $x_{\pi_1^{(j)}} \in D^+_{\pi^{(j)},1}$, $x_{\pi_2^{(j)}} \in D^+_{\pi^{(j)},2}\left(x_{\pi_1^{(j)}}\right)$, $x_{\pi_3^{(j)}} \in D^+_{\pi^{(j)},3}\left(x_{\pi_1^{(j)}}, x_{\pi_2^{(j)}}\right)$, … , $x_{\pi_{k_j}^{(j)}} \in D^+_{\pi^{(j)},k_j}\left(x_{\pi_1^{(j)}}, x_{\pi_2^{(j)}}, \dots, x_{\pi_{k_j-1}^{(j)}}\right)$ (ограничения D)

(5) $c_j(X_j) \Leftrightarrow x_{i_1^{(j)}} \in D_j^-\left(x_{i_2^{(j)}}, \dots, x_{i_{k_j}^{(j)}}\right)$ (ограничения E)

где $D_j^+ \in Z$, $D^+_{\pi^{(j)},k} \in Z$ — подмножества целых чисел, содержащие открытые положительные интервалы, а $D_j^- \in Z$ — подмножества целых чисел, содержащие открытые отрицательные интервалы.

Естественным визуальным представлением задачи GCPSP является двудольный направленный граф $G\langle X, C, R\rangle$, где X — множество вершин, ассоциированных с переменными задачи, C – множество вершин, ассоциированных с ограничениями задачи, и R — множество ребер графа, соединяющих вершины ограничений с соответствующими вершинами переменных следующим образом. Если переменная задачи x_i участвует в ограничении c_j в качестве независимой переменной, то ребро является входящим в вершину ограничения (в применяемой нотации $x_i \to c_j$). Если переменная задачи x_i является зависимой переменной ограничения c_j, то ребро входит в вершину переменной ($c_j \to x_i$). Если ограничение c_j предусматривает правила разрешения относительно каждой своей переменной x_i, то ребра между соответствующими вершинами будем представлять как двунаправленные и

записывать $x_i \leftrightarrow c_j$. Будем также говорить, что переменная x_{i_1} прямо предшествует переменной x_{i_2} или $x_{i_1} \to x_{i_2}$, если существует такое ограничение задачи c_j, что $x_{i_1} \to c_j \to x_{i_2}$.

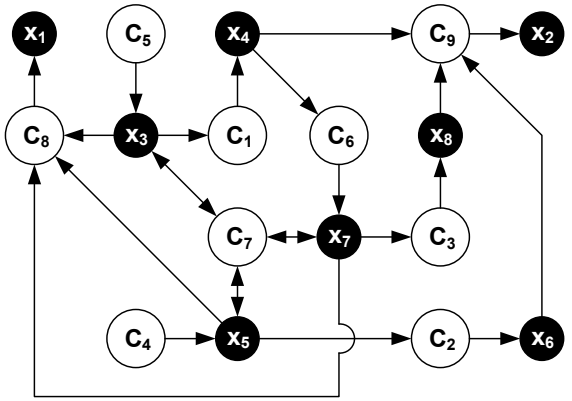

Рис. 1. Пример графа задачи GCPSP, отображающего взаимосвязи между переменными и ограничениями
Fig. 1. An example of the graph of the GCPSP task representing relationships between variables and constraints

На Рис. 1 представлен пример графа задачи GCPSP, на котором темными кружками отображены переменные задачи, а большими светлыми кругами — ограничения. Однонаправленные и двунаправленные ребра указывают характер зависимостей между переменными, который связан со способами разрешения ограничений.

В предположении, что граф задачи ацикличен, план разрешения системы ограничений представляется в виде перестановки переменных $\pi\colon \{1,2,\dots,n\} \to \{1,2,\dots,n\}$, в соответствии с которой они могут быть последовательно выражены друг через друга или уточнена область их допустимых значений в соответствии с наложенными ограничениями. Циклом в данном случае называется упорядоченный набор пар вершин $x_{i_1} \to c_{j_1} \to x_{i_2} \to c_{j_2}, \dots, x_{i_k} \to c_{j_k} \to x_{i_1} \subseteq G\langle X, C, R\rangle$, такой, что вершины-переменные и вершины-ограничения соединены последовательно направленными ребрами от вершины x_{i_l} к вершине c_{j_l} для всех $l = 1,..,k$, от вершины c_{j_l} к вершине $x_{i_{l+1}}$ для всех $l = 1,..,k-1$, и от вершины c_{j_k} к вершине x_{i_1}. Заметим, что двунаправленные ребра не включаются в цикл. План разрешения системы ограничений визуально будем отображать на графе задачи, приписывая индексы упорядочивания переменных соответствующим вершинам.

Сформулируем достаточные условия разрешимости системы ограничений с приоритетами и существования решения для связанной с ней обобщенной задачи проектного планирования GCPSP.

Утверждение 1. Пусть $\langle D, X, C, P\rangle$ — обобщенная задача GCPSP. Решение задачи всегда существует, если

(1) целевая функция задачи $f(X)$ монотонно неубывающая по каждой переменной;

(2) граф задачи $G\langle X, C, R\rangle$ ацикличен;

(3) ограничения (A) имеют наивысший приоритет, а приоритеты ограничений (B), (C) и (D) выше приоритета ограничений (E);

(4) ни одна переменная задачи не ассоциирована с более чем одним ограничением (A) в качестве зависимой переменной;

(5) каждая переменная задачи участвует, по крайней мере, в одном ограничении (A), (B), (C) или (D) в качестве зависимой переменной.

Доказательство.

Прежде всего, покажем, что система ограничений задачи разрешима и множество допустимых значений для переменных задачи не пусто. Допустимость значений будем интерпретировать в терминах локальной согласованности переменных в соответствии с назначенными приоритетами ограничений. В предположении ацикличности графа задачи существует упорядочивание переменных, при котором допустимые значения или область допустимых значений для переменных с большими индексами выражаются через соответствующие значения переменных с меньшими индексами. Покажем, что при обходе переменных по возрастанию индексов всегда существует непустое множество допустимых значений переменной, удовлетворяющее ограничения с входящими ребрами в вершину-переменную. Рассмотрим следующие взаимоисключающие случаи для определения области допустимых значений переменной:

- У вершины-переменной отсутствуют входящие ребра. В этом случае область допустимых значений переменной – все множество целых чисел.
- Имеется входящее ребро от унарного ограничения (*A*). В этом случае переменная принимает единственное допустимое значение в соответствии с явной функцией разрешения ограничения, независимо от других наложенных ограничений (*B*), (*C*) и (*D*), которые имеют меньший приоритет. Наличие нескольких ограничений (*A*) условиями утверждения исключаются.
- Имеются входящие ребра от ограничений (*B*) и (*C*) и отсутствует входящее ребро от ограничений (*A*). Независимо от назначенных приоритетов для переменной задачи всегда существует положительный полуинтервал, удовлетворяющий данным ограничениям. Ограничения (*D*) имеют меньший приоритет и поэтому

либо нарушаются, либо удовлетворяются путем уточнения допустимой области переменной в виде интервала или точки.

- Входящие ребра только от ограничений (*D*) не могут существовать в силу принятых допущений.

Таким образом, множество локально согласованных решений системы ограничений с приоритетами не пусто, причем они ограничены снизу. В силу неубывания целевой функции задачи по каждой из переменной ее минимум достигается на данном множестве и обобщенная задача проектного планирования GCPSP имеет решение. ■

Следующее утверждение определяет достаточные условия разрешимости задач проектного планирования GCPSP в случае, когда приоритеты назначаются ограничениям индивидуально независимо от их общего вида.

Утверждение 2. Пусть $\langle D, X, C, P\rangle$ — обобщенная задача GCPSP. Решение задачи всегда существует, если

(1) целевая функция задачи $f(X)$ по каждой переменной не убывает на некотором положительном полуинтервале и не возрастает на некотором отрицательном полуинтервале;

(2) граф задачи $G\langle X, C, R\rangle$ ацикличен;

(3) для каждой переменной задачи и ограничений, в которых она участвует как зависимая переменная, наивысший приоритет имеют либо одно ограничение (A), либо несколько ограничений (B), (C) и (D), либо несколько ограничений (E).

Доказательство.

Система ограничений задачи разрешима в смысле локальной согласованности переменных задачи. В самом деле, для каждой зависимой переменной всегда могут быть разрешены ограничения с наивысшим приоритетом. В случае единственного ограничения (A) это очевидно. В случае нескольких ограничений (B), (C) и (D) всегда существует положительный полуинтервал, которому удовлетворяют данные ограничения. В случае нескольких ограничений (E) существует отрицательный полуинтервал, которому удовлетворяют ограничения данного вида. Выполнимость ограничений с низкими приоритетами не имеет никакого значения для множества допустимых решений. В силу свойств монотонности целевой функции по каждой переменной на данном множестве существует, по крайней мере, одно оптимальное решение. ■

5. Алгоритм приближенного решения задач GCPSP

Опишем алгоритм приближенного решения задач проектного планирования в постановке GCPSP. Его можно рассматривать в качестве обобщения известной схемы последовательной диспетчеризации работ.

На каждом шаге алгоритма $k = 1,2,\dots,n$ определяется одна из переменных задачи x_i, $i \in I(X)$. При этом выбор очередной переменной подчиняется следующей общей схеме. Пусть $I_{PASSIVE} \subseteq I(X)$ — множество индексов переменных, значения которых определены к началу шага k, а $I_{ACTIVE} \subseteq I(X)$ – множество индексов неопределенных переменных, все прямые предшественники которых содержатся в $I_{PASSIVE}$ или вовсе не имеют предшественников. Таким образом, имеют место следующие соотношения между рабочими множествами: $I_{PASSIVE} \cap I_{ACTIVE} = \emptyset$ и если $i \in I_{ACTIVE}$, то либо не существует $j \in I(X)$ такого что $x_j \to x_i$, либо для каждого $j \in I(X)$ такого что $x_j \to x_i$, имеет место $j \in I_{PASSIVE}$.

Тем самым, множество $I(X)\backslash(I_{PASSIVE} \cup I_{ACTIVE})$ включает в себя индексы тех переменных, которые не участвовали в анализе до текущего шага и должны быть определены на следующих шагах алгоритма.

На начальном шаге инициализируются вспомогательные множества $I_{PASSIVE} = \emptyset$ и $I_{ACTIVE} = \{i \in I(X) \mid \nexists\, j \in I(X), x_j \to x_i\}$. В силу условий описанного выше утверждения граф задачи ацикличен, множество переменных, не имеющих предшественников, не пусто и алгоритм корректно стартует. На каждом шаге алгоритма отбирается активная переменная x_i, $i \in I_{ACTIVE}$, определяется ее значение и корректируются рабочие множества индексов таким образом, что активная переменная перемещается из I_{ACTIVE} в $I_{PASSIVE}$, а множество I_{ACTIVE} пополняется новыми переменными, все предшественники которых уже находятся в $I_{PASSIVE}$. Поскольку коррекция рабочих множеств осуществляется на каждом шаге алгоритма для поиска пополняемых переменных достаточно проанализировать только предшественников последователей переменной x_i. Таким образом, при переходе к следующему шагу рабочие множества корректируются следующим образом:

$$I_{ACTIVE} = I_{ACTIVE}\backslash\{i\},$$
$$I_{PASSIVE} = I_{PASSIVE} \cup \{i\},$$
$$I_{ACTIVE} = I_{ACTIVE} \cup \{k \in I\backslash(I_{PASSIVE} \cup I_{ACTIVE}) \mid x_i \to x_k, \forall\, x_{i'} \to x_k, i' \in I_{PASSIVE}\}.$$

Выборка активной переменной из множества I_{ACTIVE} выполняется на основе эвристических правил вида $i' H\, i''$, где $i', i'' \in I_{ACTIVE}$, определяющих линейный порядок на множестве переменных и позволяющих на текущем шаге выбрать переменную, наиболее предпочтительную для достижения оптимума целевой функцией. Иногда правила задают частичный порядок на множестве переменных. В подобных случаях можно определить иерархическую стратегию $H = \{H_l\}$, $l = 1,2,\dots,L$, заключающуюся в последовательном применении элементарных правил H_l в ожидании того, что одно из них позволит вынести окончательный вердикт о предпочтении одной переменной над другой. Если ни одно из правил не позволяет установить это, то приоритетной может считаться та переменная, которая имеет наименьший индекс. Очевидно, что порядок применения элементарных правил в рамках иерархической стратегии может влиять на упорядочивание переменных и качество приближенного решения

задачи. Алгоритмом допускается широкий набор эвристических правил, применяемым в задачах RCPSP [[8]]. Ниже мы приводим математически эквивалентную интерпретацию некоторых популярных правил, сохраняя их оригинальные названия:

- LIS (Least Immediate Successors): выбрать переменную с наименьшим количеством непосредственных последователей;
- MIS (Most Immediate Successors): выбрать переменную с наибольшим количеством непосредственных последователей;
- MTS (Most Total Successors): выбрать переменную с наибольшим количеством всех последователей;
- LTS (Least Total Successors): выбрать переменную с наименьшим количеством всех последователей;
- LSC (Longest Successors Chain): выбрать переменную с наибольшим числом переменных в цепочках последователей;
- SSC (Shortest Successors Chain) выбрать переменную с наименьшим числом переменных в цепочках последователей;
- SPT (Shortest Process Time): выбрать переменную с наименьшей разностью $x_j - x_i$ среди всех пар $x_j \to x_i$, связанных ограничениями вида (A) и (B). Здесь x_i — активная переменная, значение которой определяется в предположении ее отбора, а x_j — ее непосредственный последователь, значение которого рассчитывается на основе правила разрешения соответствующего ограничения и выбора минимально допустимого значения;
- LPT (Longest Process Time): выбрать переменную с наибольшей разностью $x_j - x_i$ среди всех пар $x_j \to x_i$, связанных ограничениями вида (A) и (B).

Существуют и более сложные правила, применение которых предполагает решение вспомогательных задач, связанных с локальной оптимизацией целевой функции. Важно, чтобы применяемые эвристики были релевантны целевой функции.

Значение для выбранной активной переменной определяется, исходя из требования минимизации целевой функции при условии удовлетворения максимального количества ассоциированных с ней ограничений. В случаях, когда система ограничений является переопределенной, ограничения разрешаются последовательно в соответствии с заданными приоритетами. В предположениях описанного выше утверждения всегда существует способ обеспечить локальную согласованность решения, при которой в первую очередь разрешаются ограничения с высокими приоритетами, а затем, если возможно, ограничения с низкими приоритетами.

В самом деле, после построения множеств $I_{PASSIVE}$ и I_{ACTIVE} и выбора активной переменной x_i, $i \in I_{ACTIVE}$ все независимые переменные ассоциированных с ней

ограничений (*A*), (*B*), (*C*), (*D*) и (*E*) находятся в множестве $I_{PASSIVE}$ и поэтому их значения уже определены и могут быть использованы для расчета допустимых значений x_i.

В случае наложенного ограничения вида (*A*) переменная принимает единственное значение в соответствии с правилом разрешения ограничения $x_i = g_j\left(x_{i_2^{(j)}}, x_{i_3^{(j)}}, \dots, x_{i_{k_j}^{(j)}}\right)$, где $\left\{i_2^{(j)}, i_3^{(j)}, \dots, i_{k_j}^{(j)}\right\} \subseteq I_{PASSIVE}$, независимо от других наложенных ограничений.

В случае ограничений вида (*B*), (*C*), (*D*) для переменной x_i вначале определяется допустимое множество значений как пересечение множеств интервалов $D_j^+\left(x_{i_2^{(j)}}, x_{i_3^{(j)}}, \dots, x_{i_{k_j}^{(j)}}\right)$ для всех j таких, что $c_j \to x_i$. Пересечение не пусто, поскольку множества включают в себя положительные полуинтервалы. Значение переменной выбирается, исходя из требования минимизации целевой функции по данной переменной. В случае неубывания целевой функции точка минимума всегда существует, а если она не единственна, то выбирается наименьшее значение. Переменные задачи обычно семантически связаны с датами исполнения и сроками, которые предпочтительно сократить даже при достижении целевой функцией минимума. Таким образом, значение выбранной переменной в этом случае определяется следующим образом:

$$x_i^* = \operatorname{argmin} f(x_i),$$
$$x_i \in \bigcap_{c_j \to x_i} D_j^+\left(x_{i_2^{(j)}}, x_{i_3^{(j)}}, \dots, x_{i_{k_j}^{(j)}}\right)$$

Анализ ограничений (*C*) и (*D*) имеет свои особенности. Чтобы удовлетворить ограничение вида (*C*) достаточно выполнить соответствующее условие для одной зависимой переменной. Действительно, попытка распространить условие $x_{i^{(j)}} \in D_j^+\left(x_{i_1^{(j)}}\right)$ на все зависимые переменные может оказаться обременительной для поиска качественного приближенного решения. Поэтому алгоритм предусматривает иную схему. Ограничение разрешается только для самой последней обрабатываемой переменной ограничения и только в том случае, если оно не было автоматически удовлетворено в результате определения значений предыдущих переменных. С этой целью для каждого ограничения вида (*C*) рассчитывается и хранится статус его выполнения, а также ведется подсчет числа обработанных ограничений. Такая схема обеспечивает разумную стратегию удовлетворения ограничений и минимизации целевой функции в случаях, когда переменные связаны между собой множественными ограничениями.

Для активной переменной все ассоциированные с ней ограничения (*D*) удовлетворяются в рамках общей схемы, поскольку они могут быть разрешены относительно любой переменной независимо от переменных множества

$I(X)\backslash I_{PASSIVE}$, значения которых на текущем шаге алгоритма еще не определены.

Ниже приводится псевдокод описанного выше обобщённого алгоритма.

```
Begin
    X := ⊥;
    I_PASSIVE := ∅;
    I_ACTIVE := {i | i ∈ I, ∄ i' ∈ I, x_i' → x_i};
    While |I_ACTIVE| > 0 do
        i := selectVariable(I_ACTIVE);
        x_i := computeVariable(i, X);
        I_PASSIVE := updatePassive(I_PASSIVE, i);
        I_ACTIVE := updateActive(I_ACTIVE, i);
    End while;
End;

Function selectVariable(I_ACTIVE)
Begin
    i := I_ACTIVE[1];
    For k := 2 to |I_ACTIVE| do
        If I_ACTIVE[k] H i then
            i := I_ACTIVE[k];
        End if;
    End for;
    Return i;
End;
Function computeVariable(i, X)
Begin
    D* := Z;
    J := {j | c_j → x_i | c_j ↔ x_i};
    J := sort(J, P);
    For j := 1 to |J| do
        D := resolve(c_j, X_j, i);
        If j = 1 then
            If D = ∅ then
                «Ошибка в постановке задачи или решения не существует!»
                Stop;
            Else
                D* := D;
            End if;
        Else
```

```
            If D* ∩ D ≠ ∅ then
                D* := D* ∩ D;
            Else
                «Ограничение не может быть удовлетворено!»
            End if;
        End if;
    End for;
    Return argmin_{x_i ∈ D*}{f(x_i)};
End;

Function updatePassive(I_PASSIVE, i)
Begin
    Return I_PASSIVE ∪ {i};
End;

Function updateActive(I_ACTIVE, i)
Begin
    Return (I_ACTIVE \ {i}) ∪ {k ∈ I \ (I_PASSIVE ∪ I_ACTIVE) | x_i → x_k, ∀ x_i' → x_k, i' ∈ I_PASSIVE};
End;
```

Вспомогательная функция $sort(J, P)$ сортирует ограничения J по убыванию приоритетов P. Перед вызовом функции множество J формируется из ограничений, в которых x_i участвует в виде зависимой переменной. Функция $resolve(c_j, X_j, i)$ разрешает ограничение c_j относительно i переменной при фиксированных значениях остальных переменных из множества X_j. Возвращаемый результат — множество допустимых значений переменной D^*. В силу сделанных допущений относительно вида ограничений, функция допускает обобщенную реализацию. Из множества D^* переменной x_i присваивается одно из значений, при котором достигается минимум целевой функции по данной переменной.

В наихудшем случае описанный алгоритм имеет квадратичную вычислительную сложность $O(n^2)$ от количества переменных задачи n, поскольку на каждом шаге алгоритма из множества активных переменных, мощность которого ограничена n, необходимо выбирать наиболее предпочтительную переменную путем последовательного применения эвристических правил. Данная полиномиальная оценка приемлема для решения прикладных задач высокой размерности.

Приведённый алгоритм может быть легко трансформирован для поиска точного решения. Для этого достаточно подменить функцию $selectVariable$ на аналогичную, в которой выбор приоритетной переменной из множества активных I_{ACTIVE} осуществляется на основе перебора вариантов с отсечением

по верхней оценке целевой функции задачи U_f. Псевдокод функции приводится ниже.

```
Function selectVariable(I_ACTIVE, X, U_f)
Begin
    If U_f ≠ ⊥ and f(X) > U_f then
        Return ⊥;
    Else
        If |I_ACTIVE| = 1 then
            Return I_ACTIVE[1];
        Else
            i_BEST := ⊥;
            f_BEST := ⊥;
            For i := 1 to |I_ACTIVE| do
                X' := X;
                x'_i := computeVariable(i, X');
                I'_ACTIVE := updateActive(I_ACTIVE, i);
                While |I'_ACTIVE| > 0 do
                    i' := selectVariable(I'_ACTIVE, X', f_BEST);
                    If i' = ⊥ then
                        Break while;
                        Next i;
                    Else
                        x'_i' := computeVariable(i', X');
                        I'_ACTIVE := updateActive(I'_ACTIVE, i');
                    End if;
                End while;
                If i_BEST = ⊥ then
                    i_BEST := i;
                    f_BEST := f(X');
                Else
                    f := f(X');
                    If f < f_BEST then
                        i_BEST := i';
                        f_BEST := f;
                    End if;
                End if;
            End for;
        End if;
        Return i_BEST;
```

End if;

End;

Таким образом, точное решение находится путём перебора всех последовательностей назначений переменных, при которых достигается минимальное значение целевой функции на множестве локально согласованных решений. При этом алгоритм реализует метод «ветвей и границ», при котором из анализа исключаются последовательности, заведомо приводящие к неоптимальным решениям.

Утверждение 3. Класс задач RCPSP принадлежат классу задач проектного планирования GCPSP. В предположениях классической постановки RCPSP существуют решения эквивалентных задач GCPSP. Обобщенный алгоритм проектного планирования сводится к алгоритму последовательной диспетчеризации работ в случае задач RCPSP.

Доказательство. Покажем, что любая задача RCPSP удовлетворяет условиям обобщенной постановки GCPSP. Выберем в качестве неизвестных переменных задачи целочисленные даты начала работ $x = (x_1, x_2, \dots x_n)$. Очевидно, что функция минимизации проектного времени $\min(\max_{i=1,\dots N}(x_i + d_i))$ удовлетворяет условиям задачи GCPSP. Условия начала проекта $x_i \geq t_0, i = 1, \dots, N$ выражаются ограничениями вида (B), в которых отсутствуют независимые переменные, а область допустимых значений зависимых переменных определяется положительными полуинтервалами. Отношения предшествования работ $x_{i_{(l)}} + d_{i_{(l)}} \leq x_{i^{(l)}},\ l = 1,2, \dots, L$ также могут быть представлены ограничениями вида (B), если в качестве зависимых переменных использовать даты начала работ-последователей. Наконец, ресурсные условия

$$\sum_{i \in I(k)} \theta(t, x_i, d_i)\, r_{ik} \leq R_k, k = 1,2, \dots, K$$

могут быть отнесены к ограничениям (D), поскольку последовательно разрешимы относительно каждой переменной при фиксированных значениях других.

В предположениях классической постановки задач RCPSP для эквивалентных задач GCPSP также существует решение. Для доказательства воспользуемся достаточными условиями утверждения 1. Во-первых, целевая функция задачи RCPSP монотонно не убывает по каждой переменной. Во-вторых, граф задачи GCPSP ациклиен при условии, что отношения предшествования не образуют циклов. В-третьих, порождаемые условиями RCPSP ограничения вида (B) всегда разрешимы. Порождаемые ограничения вида (D) разрешимы, если уровни потребления ресурсов индивидуальными работами не превышают их доступное количество. Наконец, поскольку в эквивалентной задаче отсутствуют ограничения (A) и (E), а условия старта проекта распространяются на все переменные задачи, то при любом назначении приоритетов на

индивидуальные ограничения (B) и (D) решение задачи GCPSP существует согласно утверждению 1.

Для доказательства сводимости обобщенного алгоритма в случае задачи RCPSP достаточно адресоваться к описанию алгоритма последовательной диспетчеризации [[3]], а также проинтерпретировать работы в терминах переменных, а отношения предшествования — в терминах ограничений и соответствующих правил разрешения задачи GCPSP. При использовании упомянутых выше эвристик LIS, MIS, MTS, LTS, LSC, SSC, связанных с количеством непосредственных или общих последователей-переменных, предпочтение будет отдаваться тем же активным работам, что и в алгоритме последовательной диспетчеризации с аналогичными эвристиками, но выраженными в терминах предшествования работ. ■

Вместе с тем, область практического применения рассмотренной обобщенной постановки GCPSP существенно шире, поскольку охватываются прикладные задачи проектного планирования в расширенных постановках с учетом альтернативных критериев составления расписаний, сложных моделей исполнения работ, многопараметрических взаимосвязей, особенностей привлечения ресурсов, применения календарных графиков, финансового и логистического обеспечения проектных работ. Для краткости остановимся на некоторых примерах, характерных для расширенных постановок.

При планировании обычно применяются составные работы, которые обеспечивают иерархическую многоуровневую структуризацию проектного плана. Даты начала составных работ x_i и их продолжительности d_i связаны с параметрами дочерних работ $j \prec i$ следующим образом:

$$x_i = \min_{j=1,\dots,N \mid j \prec i}(x_j)$$

$$d_i = \max_{j=1,\dots,N \mid j \prec i}(x_j + d_j - x_i)$$

Данные условия представимы ограничениями (A) в постановке GCPSP и порождают ациклический подграф задачи, допускающий последовательное вычисление значений переменных, начиная с параметров дочерних работ и заканчивая параметрами составных работ. Вычисления естественным образом обобщаются на случай многоуровневых составных работ, для которых обход начинается с простых работ нижнего уровня и распространяются на работы верхних уровней.

Для представления вех в проектном плане можно использовать простые работы с нулевой продолжительностью. Более содержательным является моделирование «коротких гамаков» и «длинных гамаков», которые можно описать в постановке GCPSP с помощью соответствующих ограничений (A).

Пусть дата начала «короткого гамака» x_i и его продолжительность d_i, тогда

$$x_i = \max_{j=1,\dots,N \mid x_j \to x_i}(x_j + d_j)$$

$$d_i = \min_{j=1,\dots,N \mid x_i \to x_j}(x_j - x_i)$$

Для «длинного гамака» соотношения приобретают вид

$$x_i = \min_{j=1,\dots,N \mid x_j \to x_i} (x_j + d_j)$$
$$d_i = \max_{j=1,\dots,N \mid x_i \to x_j} (x_j - x_i)$$

При условии, что взаимосвязи «гамаков» не образуют циклов, данные ограничения в постановке GCPSP также приводят к ацикличному графу и могут быть последовательно разрешены.

Другим примером расширенной постановки задач проектного планирования могут служить отношения предшествования типа «начало-начало», «начало-окончание», «окончание-начало» и «окончание-окончание», для которых могут быть установлены величины задержки. Пусть $i_{(l)}$ — индекс работы-предшественника, участвующего в связи l, $i^{(l)}$ — индекс работы-последователя, участвующего в связи l, и Δt_l — задержка связи, для которой допустимы и отрицательные значения. Тогда отношения предшествования выражаются следующими ограничениями в соответствии с перечисленными выше типами:

$$x_{i^{(l)}} \geq x_{i_{(l)}} + \Delta t_l$$
$$x_{i^{(l)}} \geq x_{i_{(l)}} - d_{i^{(l)}} + \Delta t_l$$
$$x_{i^{(l)}} \geq x_{i_{(l)}} + d_{i_{(l)}} + \Delta t_l$$
$$x_{i^{(l)}} \geq x_{i_{(l)}} + d_{i_{(l)}} - d_{i^{(l)}} + \Delta t_l$$

В постановке GCPSP данные условия выражаются ограничениями вида (B).

Другим важным аспектом является учет рабочих календарей, в соответствии с которыми исполняются работы и привлекаются ресурсы. Продолжительности, трудозатраты, задержки обычно выражаются в единицах рабочего времени, которые должны быть пересчитаны в календарные даты. Например, если x_i — дата начала простой работы и d_i — ее продолжительность, то дата завершения определяется не выражением $x_i + d_i$, как в приведенных выше формулах, а функцией календаря как $g(x_i, d_i)$. Примечательно, что общий вид ограничений в рамках обобщенной постановки GCPSP при этом не меняется.

Характерной особенностью прикладных постановок является также задание алгебраических условий для дат начала и завершения работ. Данные условия типа «начать не раньше», «завершить не раньше» или «начать не позже», «завершить не позже» выражаются в постановке GCPSP ограничениями вида (B) и (E) соответственно:

$$x_i \geq t_i$$
$$x_i \geq t_i - d_i$$
$$x_i \leq t_i$$
$$x_i \leq t_i - d_i,$$

где t_i — соответствующие директивные сроки. Согласованность ограничений обычно не контролируется пользователем и они могут оказаться

переопределенными даже в случае одной проектной работы. Для разрешения подобных ситуаций ограничениям могут быть приписаны приоритеты, а выполнимость условий проинтерпретирована в терминах локальной согласованности. Принципиально, что обобщенная постановка GCPSP предусматривает и такую возможность.

Рассмотрим более интересный случай, когда временные условия задаются пользователем для дат начала и завершения составных работ. Поскольку сами параметры составных работ являются зависимыми переменными от соответствующих параметров дочерних работ, непосредственный анализ таких ограничений не возможен. Однако подобные условия могут быть переопределены эквивалентным образом. Например, если для составной работы i задано условие «начать не раньше» в виде $x_i \geq t_i$, то оно может быть переопределено для всех дочерних работ $j \prec i$ как $x_j \geq t_i$, поскольку $x_i = \min_{j=1,\dots,N \mid j \prec i} (x_j)$. Тем самым условие в постановке GCPSP представимо ограничениями вида (B). Если для составной работы i задано условие «завершить не раньше» в виде $x_i \geq t_i - d_i$, то оно переопределяется для дочерних работ $j \prec i$ как $\bigvee_{j \prec i} x_j \geq t_i - d_j$, поскольку $d_i = \max_{j=1,\dots,N \mid j \prec i} (x_j + d_j - x_i)$. Данное условие в постановке GCPSP представляется ограничением вида (C).

Наконец, в классической постановке RCPSP рассматриваются только возобновимые ресурсы с постоянным профилем использования. В расширенных постановках обычно участвуют модели возобновимых и невозобновимых ресурсов с переменными профилями использования и доступности. Ресурсные ограничения в подобных случаях могут быть описаны следующим образом:

$\forall\, t \geq t_0$ имеет место $\sum_{i \in I(t,r)} q_{i,r}(t, x_i, d_i) \leq Q_r(t), r = 1,2, \dots, R,$

где монотонно неубывающая ограниченная функция $Q_r(t)$ определяет доступное количество ресурса r на момент времени t в предположении, что данный ресурс не расходовался. Ограниченная функция $q_{i,r}(t, x_i, d_i)$ задает профиль использования ресурса r задачей i на момент времени t, а множество $I(t, r)$ определяет индексы работ, использовавших или использующих ресурс r на момент времени t.

Пусть для возобновимых ресурсов $q_{i,r}(t, x_i, d_i) = 0$ при $t < x_i$ или $t \geq x_i + d_i$ и $q_{i,r} = \max_{x_i \leq t < x_i + d_i} \{q_{i,r}(t, x_i, d_i)\}$, а для невозобновимых ресурсов имеет место $q_{i,r}(t, x_i, d_i) = 0$ при $t < x_i$ и $q_{i,r}(t, x_i, d_i) = q_{i,r}$ при $t \geq x_i + d_i$, где $q_{i,r} = \max_{t \geq x_i} \{q_{i,r}(t, x_i, d_i)\}$. Тогда можно показать, что условие в постановке GCPSP представляется ограничением вида (D), если для невозобновимых ресурсов выполняется $\sum q_{i,r} \leq Q_r$, а для возобновимых ресурсов имеет место $q_{i,r} \leq Q_r$, где $Q_r = \max_{t \geq t_0} \{Q_r(t)\}$.

Рассмотренные примеры демонстрируют общность предложенной постановки GCPSP для разнообразных прикладных задач проектного планирования.

Заключение

В работе предложена, формализована и обоснована обобщенная математическая постановка задач проектного планирования GCPSP, сформулированы и доказаны достаточные условия разрешимости задач данного класса, а также описан эффективный приближенный алгоритм полиномиальной сложности. В рамках дальнейших исследований предполагается реализовать данный алгоритм в составе программно-инструментальной среды общего назначения, обеспечивающей задание условий и решение разнообразных прикладных задач проектного планирования.

Список литературы

[1]. Kolisch R. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. // European Journal of Operational Research, том 90, 1996 г., стр. 320-333.

[2]. Kolisch R., Sprecher A. PSPLIB - A project scheduling library. // European Journal of Operational Research, том 96, 1996 г., стр. 205-216.

[3]. Лазарев А. А., Гафаров Е. Р. Теория расписаний. Задачи и алгоритмы. // МГУ им. М. В. Ломоносова, Москва, 2011 г., 222 стр.

[4]. Kelley James E. Jr., Walker Morgan R. Critical-Path Planning and Scheduling. // Proceedings of the eastern joint computer conference, 1959 г., стр. 160-173.

[5]. Land A. H., Doig A. G. An automatic method of solving discrete programming problems. Econometrica, том 28, выпуск 3, 1960 г., стр. 497-520.

[6]. Brucker P., Knust S. Complex scheduling. Springer, Берлин, 2006 г., 342 стр.

[7]. Аничкин А. С., Семенов В. А. Современные модели и методы теории расписаний. Труды ИСП РАН, том 26, вып. 3, 2014 г., стр. 5-50, DOI: 10.15514/ISPRAS-2014-26(3)-1.

[8]. Kolisch R. Project Scheduling under Resource Constraints: Efficient Heuristics for Several Problem Classes. Springer, Берлин, 1995 г., 212 стр.

Mathematical formalization of project scheduling problems

[1]A.S. Anichkin <anton.anichkin@ispras.ru>
[1,2]V.A. Semenov <sem@ispras.ru>
[1]Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia
[2]Moscow Institute of Physics and Technology (State University)
9 Institutskiy per., Dolgoprudny, Moscow Region, 141700, Russia

Abstract. Theory of scheduling and project planning is widely applied in diverse scientific and industrial areas. To effectively solve application-specific problems it is necessary to take into

account a lot of factors such as task execution models, precedence relationship between tasks, resource limitations, directive deadlines, working calendars, conditions for financial and logistics support of project tasks, specific spatio-temporal requirements et al. Therefore, in practice there is a need to generalize the project scheduling problems and to consider their extended formulations. The paper provides a classical mathematical statement of RCPSP problems (Resource-Constrained Project Scheduling Problem) and a short description of the serial dispatching algorithm for their approximate solution. Shortcomings and limitations of the RCPSP statement are discussed and systemized in more details. The paper presents a mathematical formalization of project scheduling problems in extended definitioins taking into account numerous features of practical problems. The proposed statement of GCPSP problems (Generally Constrained Project Scheduling Problem) can be considered as an evolution of RCPSP problems. This statement implies a mathematically neutral specification of the optimization problem under algebraic constraints of the predefined sorts and priorities. Essentially, that the constraints are interpreted in terms of the local consistency that allows some violations in the case of overloaded algebraic systems. An effective algorithm for the approximate solution of GCPSP problems is also presented and explained by following to the classical serial algorithm. Moreover, the equivalence of the algorithms is proved for the cases when a solved GCPSP problem is reduced to the RCPSP. It is expected that the obtained results will allow developing a general-purpose software library for solving of diverse project scheduling problems.

Keywords: scheduling theory; project planning and scheduling.

DOI: 10.15514/ISPRAS-2017-29(2)-9

For citation: Anichkin A.S., Semenov V.A. Mathematical formalization of project scheduling problems. Trudy ISP RAN/Proc. ISP RAS, vol. 29, issue 2, 2017. pp. 231-256 (in Russian). DOI: 10.15514/ISPRAS-2017-29(2)-9

References

[1]. Kolisch R. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. // European Journal of Operational Research, vol. 90, 1996, pp. 320-333.

[2]. Kolisch R., Sprecher A. PSPLIB - A project scheduling library. // European Journal of Operational Research, vol. 96, 1996, pp. 205-216.

[3]. Lazarev A. A., Gafarov E. R. [Scheduling theory. Tasks and algorithms.] // Lomonosov Moscow State University, Moscow, 2011, 222 p. (in Russian).

[4]. Kelley James E. Jr., Walker Morgan R. Critical-Path Planning and Scheduling. // Proceedings of the eastern joint computer conference, 1959, pp. 160-173.

[5]. Land A. H., Doig A. G. An automatic method of solving discrete programming problems. Econometrica, vol. 28, issue 3, 1960, pp. 497-520.

[6]. Brucker P., Knust S. Complex scheduling. Springer, Berlin, 2006, 342 p.

[7]. Anichkin A. S., Semenov V. A. A survey of emerging models and methods of scheduling. Trudy ISP RAN/Proc. ISP RAS, vol. 26, issue 3, 2014. pp. 5-50 (in Russian). DOI: 10.15514/ISPRAS-2014-26(3)-1.

[8]. Kolisch R. Project Scheduling under Resource Constraints: Efficient Heuristics for Several Problem Classes. Springer, Berlin, 1995, 212 p.