Задача формирования железнодорожных составов и расписания их движения[1]

Лазарев А.А.[2,3,4], **Мусатова Е.Г.**[2], **Хуснуллин Н.Ф.**[2]
[2]*Институт проблем управления им. В.А. Трапезникова РАН, г. Москва*
[3]*Московский государственный университет им. М.В. Ломоносова, г. Москва*
[4]*Национальный исследовательский университет Высшая школа экономики, г. Москва*

Ключевые слова: теория расписаний, железнодорожный транспорт, маршрутизация, динамическое программирование.

Введение

В работе рассматривается задача формирования грузовых составов и маршрутов их следования по железнодорожной сети [1, 2]. Необходимо из имеющихся на станциях заказов (вагонов) сформировать составы и определить расписание и маршруты их движения до станций назначения так, чтобы минимизировать суммарное время выполнения работ.

Задачи железнодорожного планирования в тех или иных постановках все больше привлекают внимание специалистов в силу своей нетривиальности и практической значимости. Как правило, специалисты отдельно выделяют задачи составления расписания железнодорожного транспорта (при известных маршрутах) [3-5] и задачи маршрутизации [6, 7]. Существует и ряд комбинированных моделей [8-10]. В частности, задача формирования железнодорожных составов и маршрутов их следования в общем виде требует нахождения как маршрутов, так и расписаний движения транспорта.

В случае большой железнодорожной сети размерность подобных задач оказывается одной из основных проблем при поиске оптимального решения. В этом случае используются различные эвристические подходы [11], метод генерации колонок [9], метод декомпозиции и другие подходы целочисленного программирования [12-14].

Данная работа посвящена частному случаю проблемы, а именно, построению расписания движения одного локомотива между тремя сортировочными станциями. Для такого случая оказывается эффективным метод динамического программирования. В статье предложен полиномиальный алгоритм решения данной задачи, представлены результаты численного эксперимента.

[1] *Статья выполнена при финансовой поддержке гранта 11-08-13121-офи-м-2011-РЖД.*

1. Постановка задачи о трёх станциях

Рассматривается задача движения локомотива между тремя станциями, соединенными железнодорожными путями (см. рис. 1). На каждой станции имеется некоторое множество заказов, предназначенных для перевозки на другие станции. Каждый заказ характеризуется станцией отправления и станцией назначения.

Если заказ состоит из нескольких вагонов $k>1$, то на каждый вагон оформляется отдельный заказ, т.е. во множестве заказов будет k идентичных заказов. В дальнейшем будем предполагать, что множество заказов состоит из одновагонных заказов.

Введем следующие обозначения:

- q – максимальное количество вагонов, которое может перевезти локомотив за один рейс;
- O – множество всех заказов;
- n_{ij} – множество заказов для перевозки со станции i на станцию j, $i, j \in \{1,2,3\}$;
- n – общее количество заказов, т.е. $n = \sum_{i,j} n_{ij}$;
- J_{ijk} – k-й заказ для перевозки со станции i на станцию j, $k \in \{1,2,...,n_{ij}\}$;
- r_{ijk} – момент поступления заказа J_{ijk} на станцию отправления;
- r_{ij} – вектор размерности n_{ij} времен поступления всех заказов со станции i на станцию j;
- p_{ij} – время движения локомотива между станциями i и j.

Рис. 1. *Расположение станций*

Для простоты изложения будем вначале полагать, что $p_{ij} = p \ \forall i \neq j$

В качестве целевой функции выберем суммарное время окончания всех работ:

(1) $$\min F = \sum_{J_{ijk} \in O} C_{ijk},$$

где C_{ijk} – время доставки заказа J_{ijk} на станцию назначения. Данная целевая функция характеризует среднее время нахождения заказа в пути, поскольку может быть переписана в следующем виде:

$$F = \sum_{J_{ijk} \in O} \frac{C_{ijk} - r_{ijk}}{n}.$$

Данная задача является обобщением задачи о двух станциях, для которой известны полиномиальные алгоритмы решения. Так в [1] предложен алгоритм

динамического программирования для задачи о двух станциях. В [15] предложен алгоритм для решения задачи теории расписаний для одного прибора, обслуживающего требования в параллельных группах, которая может быть проинтерпретирована как задача о двух станциях (локомотив – прибор, вагоны в составе – требования в группе). В [16] рассматривается задача движения автобуса между двумя кампусами университета.

Далее предлагается алгоритм динамического программирования для решения задачи о трёх станциях.

2. Алгоритм динамического программирования

Определение 1. *Рейсом в задаче (1) будем называть четвёрку* $(s,\ t,\ s^d,\ o)$, *где s – станция отправления локомотива, t – время отправления,* s^d *– станция назначения, o – множество заказов (вагонов), перевозимых локомотивом со станции s данным рейсом.*

Обозначим за s_o номер станции, на которой находится локомотив в начальный момент времени.

Определение 2. *Допустимым расписанием в задаче (1) будем называть последовательность рейсов* (s_l,t_l,s_l^d,o_l), *l=1, …, r, где r – число рейсов, таких, что*

$$s_l = s_{l-1}^d,\ \ l = 2,\ldots,r,$$

$$t_l \geq t_{l-1} + p,\ \ l = 2,\ldots,r,$$

$$r_{ijk} \leq t_l\ \forall J_{ijk} \in o_l,$$

$$\bigcup_{l=1,\ldots,r} o_l = 0,\ \bigcap_{l=1,\ldots,r} o_l = \varnothing.$$

Рассмотрим возможные стратегии движения локомотива.

1. **Движение с заказами.** Если локомотив находится на некоторой станции, то возможно движение в одном из двух направлений с максимально возможным количеством заказов, не превышающим q.
2. **Ожидание.** Если количество имеющихся в наличии на станции заказов в выбранном направлении меньше q, то возможен режим ожидания до появления новых заказов. Если поступление новых заказов не предвидится, данный режим не возможен.
3. **Холостой ход.** Данный вариант необходим в случае отсутствия заказов на станции в выбранном направлении. Холостой ход не допустим два раза подряд или в направлении станции, на которой нет заказов и не предвидится их поступление в дальнейшем. Также будем полагать, что начало холостого хода возможно только в нулевой момент или после

прибытия локомотива на станцию. Иными словами, после режима ожидания локомотив должен отправиться на выбранную станцию с заказами, а холостой ход невозможен.

Несложно видеть, что использование этих стратегий не отсекает расписаний, доставляющих минимум в задаче (1). Поэтому далее будем считать, что движение локомотива должно удовлетворять перечисленным условиям.

Следующее предложение позволяет определить возможные моменты отправления локомотива.

Предложение 1. *Возможные моменты отправления локомотива в оптимальном расписании принадлежат множеству*

$$T = \{t = r_{ijk} + mp\} \bigcup \{t = mp\}, \quad i, j \in \{1,2,3\}, k \in \{1,\ldots,n_{ij}\}, m \in \{0,\ldots,2n-1\}.$$

Доказательство. Предположим противное. Пусть существует оптимальное расписание φ, при котором некоторые моменты отправления не принадлежат множеству T. Рассмотрим момент t_1 из этого расписания. Если $t_1 \notin T$, то выберем $\bar{t} = \max\{t : t \in T, t < t_1\}$. Очевидно, что такое $\bar{t}$ существует, поскольку $0 \in T$. Тогда в течение временного интервала $(\bar{t}; t_1]$ не поступало новых заказов, так как моменты поступления заказов входят в T. Следовательно, расписание φ', полученное из φ путем замены t_1 на $\bar{t}$, будет допустимым.

Целевая функция в терминах рейсов может быть записана следующим образом:

$$F = \sum_{l=1,\ldots,r} (t_l + p) \cdot |o_l|,$$

где $|o_l|$ – количество заказов в множестве o_l. Тогда для расписания φ имеем

$$F_{\varphi} = F_{\varphi'} + (t_1 - \bar{t}) \cdot |o_1|.$$

Следовательно, расписание φ может быть оптимальным, только если $|o_l| = 0$, т.е. в случае холостого хода в первом рейсе. Но в соответствии со стратегиями 1-3, холостой ход не возможен после режима ожидания. Значит, $t_1 \in T$.

Пусть теперь $t_l \in T, l = 1,2,\ldots,i.$ Рассмотрим (i+1)-й рейс. Если $t_{i+1} \notin T$, выберем $\bar{t} = \max\{t : t \in T, t < t_{i+1}\}$. Поскольку $t_i \in T, t_i + p \in T$, и $t_{i+1} \geq t_i + p$, то $t_i + p < t_{i+1}$. Это значит, что при замене t_{i+1} на $\bar{t}$, мы получим допустимое расписание, с теми же наборами заказов в каждом рейсе. При этом значение целевой функции останется прежним только в случае холостого хода в $(i+1)$-й рейс, что невозможно, поскольку локомотив находился в состоянии ожидания. Полученное противоречие приводит к тому, что $t_l \in T, l = 1,2,\ldots,r$. □

Определение 3. *Будем говорить, что локомотив находится в состоянии* $S(s,t,k_{12},k_{23},k_{31},k_{13},k_{32},k_{21})$, *если в момент времени* $t \in T$ *он расположен на станции s, и к данному моменту доставлено* k_{12} *заказов с первой станции на вторую,* k_{23} *заказов со второй станции на третью, ...,* k_{21} *заказов со второй станции на первую.*

Обозначим через $C(s,t,k_{12},k_{23},k_{31},k_{13},k_{32},k_{21})$ значение целевой функции состояния $S(s,t,k_{12},k_{23},k_{31},k_{13},k_{32},k_{21})$, т.е. наименьшее суммарное время выполнения доставленных к моменту t заказов в частичном расписании, приводящем к состоянию $S(s,t,k_{12},k_{23},k_{31},k_{13},k_{32},k_{21})$.

Переход от одного состояния к другому происходит по указанным выше стратегиям. При этом, если локомотив может перейти из состояния S^1 непосредственно в состояние S^2, то значение состояния может быть пересчитано по следующей формуле:

$$C^2 = C^1 + (t' + p) \cdot k,$$

где t' - момент времени, в котором локомотив находился в состоянии $S^1, k-$ количество доставленных заказов при переходе в новое состояние. Несложно видеть, что если локомотив перешел в новое состояние путем холостого хода или ожидания заказов, значение состояния не меняется.

Таким образом, идея алгоритма заключается в следующем. Строится граф состояний в порядке возрастания *t*. В случае возникновения двух одинаковых состояний выбирается то, которое доставляет наименьшее текущее значение целевой функции. Решением задачи является минимальное значение среди всех конечных состояний, т.е. состояний, в которых доставлены все заказы:

$$\min_{s,t} C(s,t,k_{12},k_{23},k_{31},k_{13},k_{32},k_{21})$$

Сложность алгоритма оценивается количеством возможных состояний. Поскольку количество возможных моментов времени $t \in T$ есть $O(n^2)$, то общее число состояний можно оценить как $O(n^2 \cdot (n_{12}+1) \cdot (n_{23}+1) \cdot (n_{31}+1) \cdot (n_{13}+1) \cdot (n_{32}+1) \cdot (n_{21}+1))$ или $O(m^8)$, где m– максимальное количество заказов в одном направлении: $m = \max_{ij} n_{ij}$.

Рис.2. *Допустимое расписание для примера 1*

Замечание 1. В случае, если времена движения локомотива между станциями различны и равны p_{12}, p_{23}, p_{13}, меняется множество возможных моментов отправления локомотива:

$$T = \{t = r_{ijk} + m_1 p_{12} + m_2 p_{23} + m_3 p_{13}\} \bigcup \{t = m_1 p_{12} + m_2 p_{23} + m_3 p_{13}\},$$

$$i, j \in \{1,2,3\}, k \in \{1,...,n_{ij}\}, m_1 + m_2 + m_3 \in \{1,...,2n-1\}.$$

Мощность множества T составит $O(n^5)$. В остальном идея алгоритма остается той же.

3. Пример

Рассмотрим следующий пример задачи. Пусть *n=6, $r_{1,2}=r_{2,3}=r_{3,1}=\{1,3\}$, q=2, p=2.*
Допустимое расписание для данного примера показано на рис.2. Здесь жирная линия обозначает движение локомотива между станциями. Цифры вдоль этой линии обозначают номера заказов, соответствующих перевозимым вагонам.

Рис. 3. *Возможные переходы из начального состояния*

Находясь в начальный момент времени *t=0* на станции 1, у локомотива есть выбор (см. рис. 3):

- остаться на станции *s=1* до момента времени поступления заказов *t=1*, тем самым перейти в состояние *S(1,1,0,0,0,0,0,0);*
- холостым ходом переехать на станцию *s=2, S(2,2,0,0,0,0,0,0);*
- холостым ходом переехать на станцию *s=3, S(3,2,0,0,0,0,0,0).*

Рис 4. *Дерево всех состояний для примера задачи*

Если в начальный момент времени локомотив ожидает поступления заказов на станцию *s=1*, то в следующий момент времени есть возможность перевезти доступный 1 вагон на станцию 2, перейдя в состояние *S(2,3,1,0,0,0,0,0)*, либо остаться на станции *s=1* до следующего момента поступления заказов на станцию *t=3*, оказавшись в состоянии S(1,3,0,0,0,0,0,0).

Если в начальный момент времени локомотив выбрал возможность холостым ходом переехать на станцию 2, тогда в следующий момент времени локомотив имеет право либо перевезти все доступные на данный момент времени заказы на станцию 3, перейдя в состояние *S(3,4,0,1,0,0,0,0)*, либо остаться на станции 2 в ожидании новых заказов (состояние *S(2,3,0,0,0,0,0,0)).*

В последнем случае у локомотива есть один единственный вариант движения - перевезти все доступные к этому моменту времени заказы на станцию 3 к моменту времени *t=5,* (состояние *S(3,5,0,2,0,0,0,0)*). Отметим, что других вариантов перехода локомотива из предыдущего состояния не имеется. Оставаться на станции не представляется возможным, так как поступления

новых заказов не ожидается. Холостой ход на соседние станции так же невозможен согласно описанным выше правилам (в предыдущий момент времени локомотив стоял в ожидании заказов).

На рис. 4 представлено дерево возможных состояний. Рассмотрим ветку, содержащую конечное состояние с минимальным значением целевой функции, равным 26. Находясь на станции 3 в момент времени $t=5$, локомотив имеет возможность лишь перевезти все доступные ему заказы на станцию 1, перейдя в состояние $S(1,7,0,2,2,0,0,0)$. В свою очередь со станции 1 оставшиеся заказы можно перевезти на станцию 2, достигнув тем самым конечного состояния $S(2,9,2,2,2,0,0,0)$, в котором все поступившие заказы доставлены.

4. Особенности реализации алгоритма

Одним из ключевых моментов, предложенного алгоритма, является «склеивание» одинаковых вершин. Вершины называются одинаковыми, если у них совпадают номера станций, на которых находится локомотив в момент времени t, и количества доставленных заказов на соответствующие станции. Из двух вершин, удовлетворяющих этому условию, в дереве остается лишь одна, с наименьшим значением целевой функции.

Подобная ситуация проиллюстрирована на рис.5. Здесь, значение состояния $S(1,7,2,0,0,0,0,2)$ равно 9, если родителями являются вершины графа, заключенные в четырехугольник. Несложно показать, что если родителями состояния являются вершины, заключенные в пятиугольник, то значение равно 10. Проводя на каждом шаге подобные вычисления, алгоритм принимает решение, какую из двух веток считать родительской. В связи с описанной особенностью, необходимо на каждом шаге просматривать весь граф в поисках вершин-дубликатов.

В текущей реализации полученное дерево полностью хранится в оперативной памяти компьютера. В целях минимизации используемой памяти и увеличения производительности алгоритма предлагается иной подход к построению дерева: в оперативной памяти находятся только состояния, моменты времени которых принадлежат отрезку $[t_i-p;t_k]$, где t_i - момент времени рассматриваемого состояния, p – время движения локомотива между станциями, t_k – максимальный момент времени из множества T. Узлы дерева, находящиеся в оперативной памяти, но не удовлетворяющие этому условию, перемещаются в файл на жестком диске. Впоследствии они будут необходимы для вывода оптимального расписания движения локомотива.

Рис.5. *Демонстрация процесса «склеивания» одинаковых вершин*

При построении дерева, как и в методе ветвей и границ, большую роль играет отсечение заведомо неперспективных ветвей для экономии памяти и вычислительных ресурсов на поиск одинаковых вершин. Суть идеи заключается в следующем. В момент, когда получено некоторое конечное состояние со значением $\tilde{C}$, мы получаем оценку сверху.

Далее, для всех следующих генерируемых состояний проверяется выполнение неравенства (2) с тем, чтобы отсечь вершины с худшими значениями целевой функции:

$$C' + \sum_{ijk} [\max\{t, r_{ijk}\} + p] > \tilde{C}, \quad (2)$$

где C' – значение проверяемого состояния, t – текущий момент времени. Левая часть неравенства (2) – нижняя оценка для текущей вершины (все недоставленные заказы перевозятся на станции назначения сразу после их поступления).

5. Вычислительный эксперимент

В табл. 1 представлены результаты вычислительного эксперимента. В первом столбце входные параметры задачи – моменты времени поступления заказов на станции. Второй столбец содержит общее число заказов. В третьем – указана мощность дерева при решении задачи методом полного перебора. В четвертом – теоретическая оценка количества состояний. В последнем, в пятом – мощность дерева, полученного на практике. Во всех примерах $p=2$, $q=2$.

В настоящем эксперименте сделана попытка получить верхнюю оценку практической сложности предлагаемого алгоритма. Решение предложенных примеров требует пересмотра большого количества стратегий и соответственно приводит к построению графа большей мощности. В силу описанных выше особенностей множества T эксперимент получается существенно сложнее, если на станции поступают по 1 заказу в нечетные моменты времени.

Из таблицы видно, что практическая сложность разработанного алгоритма гораздо ниже её теоретической оценки.

r_{ij}	n	Полный перебор	Теор.оценка динам.програм.	Практ.оценка динам.програм.
$r_{1,2} = r_{2,3} = r_{3,1} = \{1,3\}$ по 1 заказу	6	3^{27}	648	38
$r_{1,2} = r_{2,3} = r_{3,1} = r_{1,3} = r_{3,2} = r_{2,1} = \{1,3\}$ по 1 заказу	12	3^{51}	753	387
$r_{1,2} = r_{2,3} = r_{3,1} = r_{1,3} = r_{3,2} = r_{2,1} = \{1,3,5\}$ по 1 заказу	18	3^{77}	166 212	2 260

$r_{1,2} = r_{2,3} = r_{3,1} = r_{1,3} = r_{3,2} = r_{2,1}$ ={1,3,5,7} по 10 заказов	240	3^{725}	1 154 289 852	1 268 585

Табл.1. *Результаты вычислительного эксперимента*

После добавления алгоритма отсечения заведомо не перспективых веток и организации хранения в оперативной памяти только состояний, которые принадлежат отрезку $[t_i - p;\ t_k]$, где t_i - момент времени рассматриваемого состояния, p – время движения локомотива между станциями, t_k – максимальный момент времени из множества T, количество возможных обсуживаемых заказов возросло до **660**.

6. Заключение

В работе была рассмотрена задача формирования железнодорожных составов и маршрутов их следования по железнодорожной сети. Предложен полиномиальный алгоритм решения задачи о доставке одним локомотивом товаров между тремя железнодорожными сортировочными станциями, соединённые между собой железной дорогой. На примере показаны шаги, как из имеющихся на станциях заказов (вагонов) сформировать составы и определить расписание их движения до станций назначения так, чтобы минимизировать суммарное время выполнения работ. Проведен вычислительный эксперимент, в рамках которого показана верхняя оценка сложности, мощность полученных деревьев при решении задачи различными подходами. Сложность полученного алгоритма равна $O(n^8)$ операций.

Перспективным видится решение следующих задач:

- построение более точных нижних оценок для отсечения заведомо неперспективных веток решений;
- решение задачи минимизации суммарного взвешенного времени выполнения всех работ;
- рассмотрение других конфигураций станций, в рамках которых локомотив будет иметь возможность доставлять заказы (четырехугольник, звезда);
- рассмотрение возможности доставки заказов несколькими локомотивами;
- увеличение производительности алгоритма и уменьшение занимаемой им оперативной памяти;
- параллелизация предложенного в работе алгоритма;
- рассмотрение возможности построения распределенной памяти для решения поставленной задачи существенно большей размерности.

Литература

1. *Лазарев А.А., Мусатова Е.Г., Гафаров Е.Р., Кварацхелия А.Г.* Теория расписаний. Задачи железнодорожного планирования. – М.: ИПУ РАН, 2012. – С.92.
2. *Лазарев А.А., Мусатова Е.Г., Кварацхелия А.Г., Гафаров Е.Р.* Теория расписаний. Задачи управления транспортными системами. - М.: Физический факультет МГУ им. М.В. Ломоносова, 2012. – С.160.
3. *C. Mannino, A. Mascis.* Real-time Traffic Control in Metro Stations // Operations Research. 2009. - 57 (4), P. 1026-1039.
4. *Oliveira O., Smith B.M.* A job shop scheduling model for the single track-railway timetabling problem. // Technical Report 2000.21, University of Leeds. — 2000.
5. *Sahin I.* Railway traffic control and train scheduling based on intertrain conflict management. // Transportation Research Part B. 1999. — No. 33 P. 511–534.
6. *A. Caprara, L. Galli, P. Toth.* Solution of the Train Platforming Problem // Transportation Science. 2011. - 45 (2), P. 246-257.
7. *Zwaneveld P. J., Kroon L. G., van Hoesel S.P.M.* Routing trains through a railway station based on a node packing model. // European Journal of Operational Research. 2001. - No. 128 P.14–33.
8. *Лазарев А.А., Мусатова Е.Г.* Целочисленные постановки задачи формирования железнодорожных составов и расписания их движения // Управление большими системами. Выпуск 38. М.: ИПУ РАН, 2012. – С.161-169.
9. *Cacchiani V., Caprara A., Toth P.* A column generation approach to train timetabling on a corridor. // 4OR. 2008. — No. 6. P. 125–142.
10. *Liu S.-Q., Kozan E.* Scheduling trains as a blocking parallel-machine job shop scheduling problem. // Computers and Operations Research. - 36(10) P. 2840–2852.
11. *F. Corman, A. D`Ariano, D. Pacciarelli, M. Pranzo.* A tabu search algorithm for rerouting trains during rail operations // Transportation Research Part B. 2010. - 44, P. 175-192.
12. *S. Harrod.* Modeling Netowrk Transition Constraints with Hypergraphs // Transportation Science. – 2011. – 45 (1), P. 81-97.
13. *G. Sahin, R.K. Ahuja and C.B. Cunha.* Integer Programming Based Approached for the Train Dispatching Problem // Tech. Rep. Sabanci Univerisity, 2010.
14. *Wolsey L.A., Nemhauser G.L.* Integer and Combinatorial Optimization. – N.Y.: John Wiley & Sons Inc., 1988.
15. *Baptiste Ph.* Batching identical jobs // Math Meth Oper Res 52, 2000. - P. 355-367.
16. *Hagai Ilani, Elad Shufan, Tal Grinshpoun* A General Two-directional Two-campus Transport Problem // Proceedings of the 25th European Conference on Operational Research, Vilnius, 8-11 July, 2012. – P. 200.

Авторы:
1. **Лазарев Александр Алексеевич**,
1958 г.р., зав. лабораторией ИПУ РАН, д.ф.-м.н., профессор,
Телефон: +7 495 334-87-51
e-mail: jobmath@mail.ru
2. **Мусатова Елена Геннадьевна**,
1983 г.р., старший научный сотрудник ИПУ РАН, к.ф.-м.н.,
Телефон: +7 495 334-87-51
e-mail: nekolyap@mail.ru
3. **Хуснуллин Наиль Фаридович**,
1987 г.р., аспирант ИПУ РАН,
Телефон: +7 495 334-87-51
e-mail: nhusnullin@gmail.com