

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования

УРАЛЬСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
имени первого Президента России Б. Н. Ельцина

Институт математики и компьютерных наук
Кафедра алгебры и дискретной математики

Задача построения графика горячей прокатки, сбалансированного по видам продукции.

Алгоритмы построения графика

Допущен к защите

«__»_____ 2016 г.

Квалификационная работа

на степень магистра наук

по направлению «Математика

и компьютерные науки»

студента группы МГКН-2

Березина Антона Александровича

Научный руководитель

Баранский Виталий Анатольевич,

доктор физико-математических

наук, профессор

Екатеринбург

2016

РЕФЕРАТ

Березин А.А. ЗАДАЧА ПОСТРОЕНИЯ ГРАФИКА ГОРЯЧЕЙ ПРОКАТКИ, СБАЛАНСИРОВАННОГО ПО ВИДАМ ПРОДУКЦИИ. АЛГОРИТМЫ ПОСТРОЕНИЯ ГРАФИКА,

выпускная квалификационная работа на степень магистра наук: стр. ?, рис. ?, табл. ?, форм. ?, библи. ? назв.

Ключевые слова: ГОРЯЧАЯ ПРОКАТКА, ОРИЕНТИРОВАННЫЕ ГРАФЫ, ВЕЩЕСТВЕННО-ЦЕЛОЧИСЛЕННОЕ ПРОГРАММИРОВАНИЕ

Объект исследования — раскрашенные вершинно-взвешенные ориентированные графы специального вида, возникающие при планировании расписания для стана горячей прокатки.

Цель работы — поиск в этих графах простых цепей, в которых суммарный вес узлов каждого цвета максимально близок к заданному значению.

Решение задачи поиска «сбалансированной» простой цепи сводится к решению серии задач вещественно-целочисленного программирования. Этот подход основан на результатах Леоновой С.И. по изучению структуры рассматриваемых графов [1] и позволяет существенно снизить размерность получаемых задач вещественно-целочисленного программирования по сравнению с прямым использованием классических техник для их построения (например, [4]). Полученные результаты применяются для решения более общих задач в области планирования металлургического производства.

Оглавление

Введение	3
1 Формализация задачи построения сбалансированного графика прокатки	6
1.1 Ограничения на порядок следования партий	6
1.2 Сбалансированность графика прокатки	7
1.3 Граф предшествования партий	7
1.4 Задача существования сбалансированной простой цепи	8
1.5 Актуальность задачи существования сбалансированной простой цепи .	9
1.6 Дополнительные обозначения	9
2 Метод решения и структура работы	10
2.1 Локальные задачи существования сбалансированной простой цепи . . .	10
2.1.1 Задача с фиксированными входами и выходами	12
2.1.2 Задача со свободными входами и выходами	13
2.2 Исходная задача существования сбалансированной простой цепи	14
3 Формализация в виде задачи математического программирования	15
3.1 Локальные задачи существования сбалансированной простой цепи . . .	16
3.1.1 Задача с фиксированными входами и выходами	16
3.1.2 Задача со свободными входами и выходами	19
3.2 Исходная задача существования сбалансированной простой цепи	20
4 Программная реализация	22
4.1 Основные структуры данных	22
4.2 Загрузка задания	23
4.3 Модель	23
4.4 Перевод задания в неравенства	23
4.5 Результаты экспериментов	24
4.6 Инструкция пользователю	25
Заключение	26
Список литературы	27
Приложение	28

Введение

В настоящей работе исследуется задача оперативного планирования работы непрерывных полосовых станов горячей прокатки стали.

Горячая прокатка стали это процесс производства полос различной ширины и толщины из стальных заготовок, имеющих форму параллелепипедов. В металлургии такие заготовки называются слябами. Обычно ширина сляба заметно больше его толщины. Во время прокатки слябы один за другим проходят через непрерывный полосовой стан горячей прокатки, который преобразует их в полосы давлением в клетях. Клеть это основной рабочий механизм стана, состоящий из рабочих валков, которые непосредственно воздействуют на заготовку, и опорных валков, которые располагаются над и под рабочими валками и служат для увеличения жесткости валковой системы. При этом ширина прокатанной полосы близка к ширине сляба, из которого она получена, а ее толщина на порядок или два меньше толщины сляба. Длина полосы может исчисляться в сотнях метров, длина сляба от нескольких метров до десяти-двенадцати метров.

При планировании работы стана оперируют не отдельными слябами, а их группами, так называемыми партиями прокатки. Партия прокатки представляет собой набор слябов, имеющих одинаковые геометрические размеры и одинаковый химический состав (с погрешностью, которой можно пренебречь). Для всех слябов в партии заданы одни и те же значения ширины и толщины, которые должны получить прокатанные из них полосы на выходе из стана. Таким образом, слябы каждой партии считаются идентичными, как и прокатываемые из них полосы. После того, как все слябы прокатаны в полосы, полученную совокупность также называют партией. Далее по контексту будет понятно, о каком состоянии партии идет речь. Во время прокатки слябы каждой партии следуют друг за другом непрерывно, не перемежаясь слябами из других партий, то есть партии прокатываются целиком и не дробятся на части.

Можно выделить два рабочих цикла стана. Первый, или большой цикл, заключен между двумя последовательными заменами опорных валков стана. Замена делается обычно один раз в несколько дней. Второй, или малый цикл, заключен между двумя последовательными заменами рабочих валков стана. Длительность цикла работы стана между двумя последовательными заменами рабочих валков составляет несколько часов. Задача оперативного планирования работы стана состоит в формировании последовательности партий, которая будет прокатана на стане в период между двумя очередными заменами рабочих валков. Такую последовательность партий называют графиком прокатки. Обычно в сутки требуется сформировать примерно четыре-шесть последовательных графиков прокатки.

В данной работе рассматривается ситуация, когда для каждой партии прокатки известно, для какого вида продукции она изготавливается, пойдут ли полосы партии сразу на отгрузку потребителю (после порезки) или будут направлены для дальнейшей обработки в другие цеха предприятия (например, травление, холодная прокатка и прочее). В зависимости от того, куда будет направлена партия после прокатки, ее относят к тому или другому виду продукции. Обычно оперируют десятию-

пятнадцатью видами продукции.

Для того, чтобы обеспечить выполнение заказов, отгрузку продукции непосредственному потребителю и равномерную загрузку сырьем для работы других цехов предприятия, при формировании каждого следующего графика прокатки создается так называемое задание на прокатку. Задание на прокатку включает в себя общее количество металла, которое должен содержать формируемый график, а также количество металла по отдельным видам продукции. Целевые значения количества металла рассчитываются на момент планирования исходя из текущего состояния складов полуфабрикатов, количества готовой продукции, состава уже сформированных графиков прокатки, информации о трудоемкости логистических операций, производительности последующих в технологических цепочках цехов и агрегатов и так далее. Целевые значения количества металла в графике прокатки даются вместе с диапазонами допустимого отклонения от них, составляющими несколько процентов. Обычно вместо целевого значения количества металла по какому-либо виду продукции рассматривают сразу диапазон допустимых значений. Например, задание на прокатку может выглядеть так: из данного множества партий, включающего в себя партии вида «а», «b» и «с», требуется сформировать график прокатки, в котором суммарная длина прокатанных полос находится в диапазоне от 100 до 120 километров, суммарная длина полос партий вида «а» находится в диапазоне от 35 до 40 километров, суммарная длина полос партий вида «b» находится в диапазоне от 50 до 60 километров, а на суммарную длину полос вида «с» ограничение не накладывается.

Существует несколько критериев оценки качества графика прокатки. В данной работе основным критерием оценки качества графика прокатки выбрана степень соответствия суммарной длины полос графика и суммарной длины полос партий по каждому виду продукции заданным диапазонам допустимых значений. Если в графике прокатки суммарная длина полос партий по каждому виду продукции находится в заданном диапазоне, то такой график прокатки будем называть сбалансированным по видам продукции, или просто сбалансированным.

Задача формирования сбалансированного графика прокатки является весьма нетривиальной, поскольку существует целый ряд технологических ограничений, которым график прокатки должен удовлетворять.

Основными технологическими ограничениями являются ограничения на допустимый порядок следования партий друг за другом в графике прокатки. В данной работе рассмотрены два из них, оба ограничения возникают из-за конструктивных особенностей стана горячей прокатки. Первое ограничение состоит в том, что требуется формировать только такие графики прокатки, в которых ширина каждой следующей партии не возрастает. Оно обусловлено тем, что во время прокатки рабочие валки стана сильно изнашиваются, и на их поверхности появляется углубление по ширине прокатываемых заготовок. Поэтому если после заготовки с меньшей шириной пропустить через стан заготовку с большей шириной, то на поверхности последней могут появиться дефекты. Также к первому ограничению относится требование, чтобы ширина каждой следующей партии в графике прокатки не уменьшалась больше, чем на заданную величину. Второе ограничение на порядок следования партий друг за другом в графике прокатки заключается в том, что толщина каждой следующей партии не должна слишком сильно отличаться от толщины предыдущей. Это ограничение обусловлено тем, что во время прокатки необходимо подстраивать расстояние между верхними и нижними валками стана в соответствии с толщиной прокатываемой заготовки, при этом хочется минимизировать эти перестроения валков.

Кроме того в данной работе рассматривается еще один класс ограничений связанных со временем готовности партий к прокатке.

Поскольку обычно производственный цикл металлургического предприятия является непрерывным, то при формировании каждого следующего графика прокатки помимо текущего множества готовых к прокатке партий, также известно, какие еще партии будут произведены и станут готовы к прокате в ближайшем будущем. Эти еще не произведенные партии участвуют в формировании графиков прокатки наравне с уже готовыми партиями. Рассматривается ситуация, когда для каждой партии известен момент времени, начиная с которого партия готова к прокатке, а также известно какой промежуток времени требуется на прокатку данной партии. Эта информация позволяет формировать графики, в которых некоторые партии еще не готовы к прокатке на момент начала прокатки графика. При этом накладывается требование, чтобы к моменту, когда очередь дойдет до каждой следующей партии, она была готова к прокатке.

В настоящей работе будет предложена математическая модель для задачи формирования сбалансированного графика прокатки и алгоритм ее решения.

Глава 1

Формализация задачи построения сбалансированного графика прокатки

Обозначим все множество партий через P . Рассматривается конечный набор партий, то есть $|P| \in \mathbb{N}$.

1.1 Ограничения на порядок следования партий

Для начала формализуем ограничения на порядок следования партий друг за другом в графике прокатки.

Введем следующие обозначения.

- $w : P \rightarrow \mathbb{R}^+$ — ширина полос, получаемых из слябов партии, значение w на элементе $p \in P$ будем обозначать w_p .

Множество всех ширин партий обозначим через $W = \{w_p | p \in P\}$.

- $\delta \in \mathbb{R}^+$ — максимальная величина допустимой разности значений ширины полос между любыми двумя соседними партиями.
- $t : P \rightarrow \mathbb{R}^+$ — толщина полос, получаемых из слябов партии, значение t на элементе $p \in P$ будем обозначать t_p .
- $r : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ — функция, определяющая максимальную величину, на которую может отличаться значение толщины партии от значения толщины соседней партии в графике прокатки. Данная функция должна быть монотонно неубывающей, то есть $\forall t_1, t_2 \in \mathbb{R}^+ t_1 \leq t_2 \Rightarrow r(t_1) \leq r(t_2)$.

Ограничение на предшествование партий определим следующим образом: партия q может непосредственно следовать за партией p в графике прокатки в том и только в том случае, когда выполнено

1. ограничение «для перехода по толщине»: $|t_p - t_q| \leq \min\{r(t_p), r(t_q)\}$;
2. ограничение «для перехода по ширине»: $0 \leq w_p - w_q \leq \delta$.

Определенное таким образом ограничение представляет собой формализацию базовых условий предшествования партий, используемых на некоторых отечественных предприятиях. В частности, на магнитогорском металлургическом комбинате

используется значения $\delta = 250 \text{ mm}$. Далее представлен пример функции r .

$$r(t) = \begin{cases} 0.8, & \text{если } t \in [1.29, 2]; \\ 1.5, & \text{если } t \in (2, 3]; \\ 2, & \text{если } t \in (3, 16]; \\ 4, & \text{если } t \in (16, 20]; \\ \text{не определена,} & \text{если } t \in [0, 1.29) \cup (20, +\infty). \end{cases}$$

Пример. Пусть имеются две партии p и q .

Если $w_p = w_q$, это означает, что для этих партий тривиально выполнено «ограничение для перехода по ширине».

Если $t_p = 2, t_q = 3$, то для этих партий «ограничение для перехода по толщине» не выполнено.

Если $t_p = 2.5, t_q = 2$, то для этих партий «ограничение для перехода по толщине» выполнено.

1.2 Сбалансированность графика прокатки

Обозначим множество видов продукции партий через F . Различных видов продукции конечное число, то есть $|F| \in \mathbb{N}$. Введем следующие обозначения.

- $f : P \rightarrow F$ — функция, сопоставляющая каждой партии соответствующий ей вид продукции.
- $l : P \rightarrow \mathbb{R}^+$ — весовая функция, характеризующая полезность партии, значение l на элементе $p \in P$ будем обозначать $l(p)$.

В зависимости от конкретной задачи, в качестве полезности партии могут выступать разные характеристики. Этими характеристиками могут быть длина (в километрах), вес (в тоннах) партии, а также более сложные величины, учитывающие спрос, время изготовления, стоимость продукции и так далее.

В данной работе $l(p)$ — суммарная длина полос партии p . Поскольку функция l названа «весовой», то далее будем говорить про вес партии, имея ввиду значение $l(p)$.

- $\mu : F \rightarrow \mathbb{R}^+$ — функция, задающая целевые значения суммарного веса партий каждого вида продукции в графике прокатки.
- $\Delta : F \rightarrow \mathbb{R}^+$ — функция, задающая абсолютную величину допустимого отклонения от целевого значения суммарного веса партий по каждому виду продукции в графике прокатки.
- $L_{min}, L_{max} \in \mathbb{R}^+$ — числа, задающие диапазон допустимых значений суммарного веса партий в графике прокатки.

1.3 Граф предшествования партий

В настоящей работе предложен подход, при котором ограничения на порядок следования партий формализуются при помощи графа предшествования партий. Пусть G раскрашенный вершинно-взвешенный ориентированный граф со множеством узлов P и с множеством дуг $E \subseteq P \times P$ таким, что $pq \in E$ в том и только в том

случае, когда партии p и q различны, и q может непосредственно следовать за p в графике прокатки. Весом каждого узла в G положим вес соответствующей партии $l(p)$, цвет каждого узла в G соответствует виду продукции, к которому относится соответствующая партия.

В прошлой работе ([3]) был придуман удобный способ укладки графа предшествования партий на координатной плоскости, при котором абсциссой каждого узла является значение функции t , а ординатой значение функции w . На рисунке 1.1 изображен фрагмент графа предшествования партий.

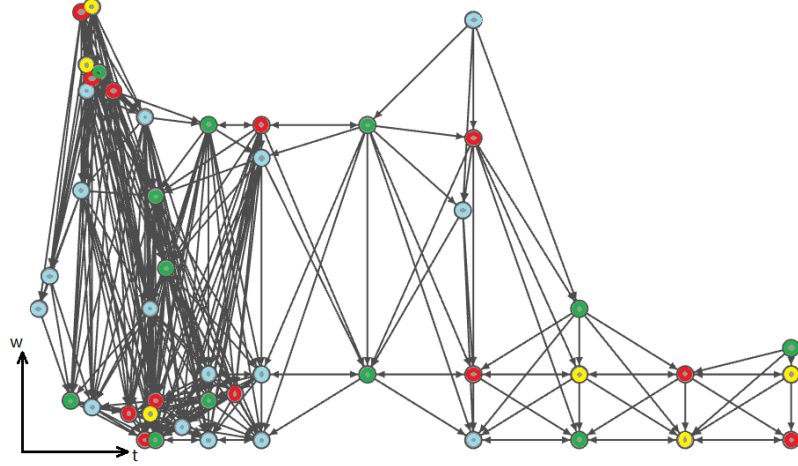


Рис. 1.1: Граф предшествования партий

На этом рисунке можно наглядно увидеть ограничение «для перехода по ширине». Оно заключается в том, что все дуги в графе предшествования партий направлены либо сверху вниз, либо горизонтально, и в графе отсутствуют дуги идущие снизу вверх. Ограничение «для перехода по толщине» так же проиллюстрировано тем, что между собой соединены лишь те узлы, которые имеют достаточно близкие абсциссы.

Любой дуге в графе G соответствует упорядоченная пара партий, которая является технологически пригодной для прокатки в таком порядке. Поэтому множеству всех простых цепей в G соответствует множество всех допустимых графиков прокатки для рассматриваемого множества партий P . Множество всех простых цепей в G обозначим через $\mathbb{B}(G)$.

Далее будем отождествлять партии прокатки и узлы графа предшествования партий.

1.4 Задача существования сбалансированной простой цепи

Пусть $P(M)$ — множество узлов произвольного подграфа графа G .

Для простой цепи $H \in \mathbb{B}(G)$ через $l(H)$ обозначим её вес.

$$l(H) \triangleq \sum_{p \in P(H)} l(p)$$

Пронумеруем виды продукции, пусть $F = \{f_1, f_2, \dots, f_s\}$.

Сформулируем задачу поиска допустимого решения — найти в $\mathbb{B}(G)$ такую цепь H , что выполнены ограничения:

$$L_{min} \leq \sum_{p \in P(H)} l(p) \leq L_{max} \quad (1.1)$$

$$\forall k \in \overline{1, s} \quad \mu_k - \Delta_k \leq \sum_{p \in P(H), f(p)=f_k} l(p) \leq \mu_k + \Delta_k \quad (1.2)$$

1.5 Актуальность задачи существования сбалансированной простой цепи

В предыдущих работах ([2, 3]) приводится обзор схожих по тематике работ и поясняется преимущество рассмотрения задачи построения графика прокатки именно как задачи поиска простой цепи в графе предшествования партий.

В данной работе, в отличие от предыдущих, выбран другой критерий оценки графика прокатки — сбалансированность. Существует несколько способов того, как можно формализовать практическую задачу построения графика прокатки сбалансированного по видам продукции. По существу это оптимизационная задача, в которой требуется приблизить распределение суммарного веса партий по каждому виду продукции максимально близко к целевым значениям.

Можно поставить, например, такую задачу с одним критерием оптимизации: найти в $\mathbb{B}(G)$ простую цепь H , где

$$\sum_{k=1}^s \left| \mu(f_k) - \frac{\sum_{f(p_i)=f_k, p_i \in P(H)} l(p_i)}{\sum_{p_i \in P(H)} l(p_i)} \right| \rightarrow \min, p_i \in P(H)$$

Такая задача имеет недостатком то, что в ней учитывается только «общая» сбалансированность решения. При этом нет гарантии, что в решении не будет «перекозов» и сильных отклонений по отдельным видам продукции, что является нежелательным с практической точки зрения. Можно рассматривать многокритериальную постановку оптимизационной задачи, однако такая постановка существенно сложнее в том числе и с вычислительной точки зрения. В данной работе выбрана именно задача существования сбалансированной простой цепи, поскольку она является хорошим инструментом для решения практической задачи, при этом постановка является достаточно простой, что позволяет решать ее за приемлемое с практической точки зрения время.

1.6 Дополнительные обозначения

Пусть p — произвольный узел в G . Для дальнейшего изложения удобно ввести следующие обозначения.

- $In(p)$ — множество узлов q таких, что дуга $qp \in E$,
 $|In(p)|$ — входящая валентность узла p .
- $In^+(p) = \{q \in In(p) | w_q > w_p\}$.
- $Out(p)$ — множество узлов q таких, что дуга $pq \in E$,
 $|Out(p)|$ — исходящая валентность узла p .

Также рассмотрим множество всевозможных ширин узлов $W = \{w_p | p \in P\}$. Для $w \in W$ обозначим через $P_w = \{p \in P | w_p = w\}$ множество узлов, имеющих данное значение w . G_w — граф, индуцированный множеством P_w .

Глава 2

Метод решения и структура работы

Инструментом для решения практической задачи построения сбалансированного графика прокатки является задача существования сбалансированной простой цепи в графе следования партий. В предыдущих работах ([2, 3]) рассмотрена оптимизационная постановка задачи поиска простой цепи в графе предшествования партий, в которой критерием оптимальности выбран суммарный вес вошедших в простую цепь узлов. Эту задачу удалось решить, предложив конструктивный полиномиальный алгоритм построения простой цепи с временной сложностью $O(n^3)$, где n — число узлов в графе предшествования партий. Для решения задачи построения сбалансированной простой цепи создание подобного полиномиального алгоритма оказалось гораздо более сложным, поскольку помимо «маршрутных» ограничений на порядок следования узлов в задаче присутствуют «объемные» ограничения на суммарный вес узлов по каждому виду продукции. По этой причине было решено прибегнуть к аппарату математического программирования, поскольку он позволяет единообразно рассматривать как «маршрутные», так и «объемные» ограничения. Этот подход ценен еще и по той причине, что позволяет расширять рассматриваемую постановку, добавляя другие ограничения, которые существуют в реальной практической задаче. «Объемные» ограничения не исчерпывают все виды ограничений, которые можно рассматривать при моделировании практической задачи. Существуют еще, например, «временные» ограничения, связанные со временем готовности партий к прокатке и длительностью прокатки каждой партии. Используемый в данной работе подход позволяет учесть их наравне с остальными ограничениями, тогда как конструктивный полиномиальный алгоритм скорее всего стал бы непригоден для решения новой задачи.

Итак, в данной работе решение задачи существования «сбалансированной» простой цепи сводится к решению серии задач вещественно-целочисленного линейного программирования. Формализация «маршрутных» ограничений на порядок следования узлов в искомой простой цепи базируется на исследовании структуры графа предшествования партий ([1, 2]). Благодаря этому удастся существенно снизить размерность получаемых задач линейного программирования по сравнению с прямым использованием классических техник для построения маршрутных задач (например, [4]).

2.1 Локальные задачи существования сбалансированной простой цепи

Решение задачи существования сбалансированной простой цепи начнем с рассмотрения некоторых приближенных и упрощенных вариантов, затем рассмотрим исходную

задачу. Рассматриваемые приближенные варианты задачи ценны, поскольку для их решения требуется существенно меньше времени, в некоторых ситуациях это может быть полезно. Кроме того, начиная изложение с приближенных задач, можно на более простых примерах продемонстрировать идеи, которые используются и для решения исходной задачи.

Для ориентированного графа предшествования партий G рассмотрим его подграфы G_w , где $w \in W$. Для каждого фиксированного $w \in W$ подграф G_w порожден узлами p исходного графа G , имеющих значение $w_p = w$.

Для любых двух принадлежащих G_w узлов p и q ограничение «для перехода по ширине» тривиально выполняется, поэтому наличие дуги pq в графе следования партий определяется лишь ограничением «для перехода по толщине». Поскольку это ограничение является симметричным, то из существования дуги pq следует существование дуги qp , поэтому подграф G_w можно рассматривать как обыкновенный.

Итак, граф G состоит из некоторого количества обыкновенных подграфов. Если узлы графа G расположить на плоскости в соответствии со значением w и t (см. раздел 1.3 Граф предшествования партий), то каждый обыкновенный подграф G_w будут состоять из узлов, имеющих одинаковую ординату w . При этом в соответствии с ограничением «для перехода по ширине» дуги между узлами различных подграфов могут идти только сверху вниз, и не могут идти снизу вверх. Поэтому любая достаточно длинная простая цепь в графе G будет проходить через узлы подграфов G_w сверху вниз (рисунок 2.1).

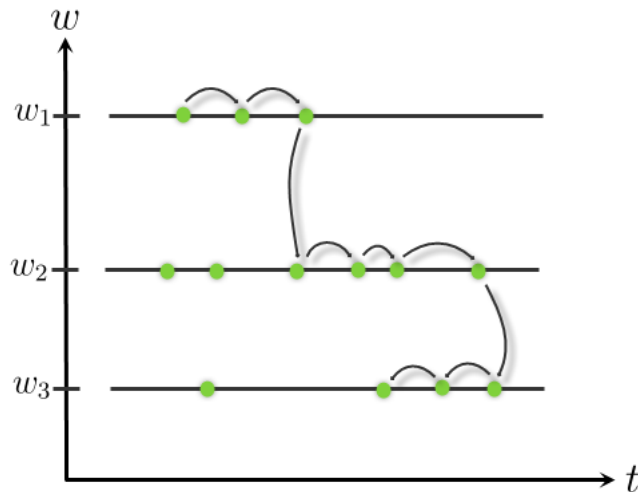


Рис. 2.1: Структура достаточно длинной простой цепи в G

Решение задачи существования сбалансированной простой цепи в G проходит через какую-то последовательность подграфов G_{w_i} . Идея, на которой основано создание приближенных задач, заключается в том, что можно заранее выбрать и зафиксировать некоторую последовательность подграфов, и потребовать, чтобы искомая простая цепь в обязательном порядке проходила через выбранные подграфы. Это позволяет решать задачу существенно меньшей размерности, чем исходная задача существования, поскольку фиксированная последовательность подграфов, через которые должна пройти искомая простая цепь позволяет исключить многие «маршрутные» ограничения на порядок следования узлов. Задачу существования сбалансированной простой цепи с выбранной последовательностью подграфов назовем локальной задачей существования. Подробное описание локальной задачи существования будет изложено в следующем разделе.

Открытый вопрос, который сопутствует локальной задаче существования это то,

каким образом можно эффективно выбирать последовательность подграфов. Далее предложено некоторое эвристическое соображение, используемое в этой работе.

Разработанный в прошлых работах ([2, 3]) алгоритм построения простой цепи максимального веса в графе G перебирает подграфы G_w в порядке от большего значения w к меньшему и строит максимальную по весу цепь H_{max} . Эта цепь отлично подходит в качестве некоего начального приближения к решению задачи поиска сбалансированной простой цепи в G , поскольку обычно вес H_{max} заведомо больше, чем требуется для сбалансированной цепи (а если он меньше, то значит, поставленная задача существования сбалансированной цепи несовместна), и кроме того в H_{max} обычно заведомо больше узлов, относящихся к каждому виду продукции, чем того требует условие сбалансированности.

Далее дадим неформальное описание двух видов локальных задач существования.

2.1.1 Задача с фиксированными входами и выходами

Пусть H — простая цепь в графе G . Зафиксируем последовательность подграфов $\{G_{w_i}\}$, через которые проходит простая цепь H в порядке уменьшения значения w_i . Пронумеруем узлы цепи $H = (q_1, q_2, \dots, q_n)$. Ввиду того, что в графе G дуги, соединяющие различные подграфы G_w направлены только от подграфа с большим значением w к подграфу с меньшим значением w , узлы цепи H идут через подграфы последовательности $\{G_{w_i}\}$ по порядку. Для каждого подграфа зафиксируем узел, являющийся первым узлом цепи H , принадлежащим данному подграфу, и назовем этот узел входом в подграф для цепи H . Зафиксируем также узел, являющийся последним узлом цепи H , принадлежащим данному подграфу, и назовем этот узел выходом из подграфа для цепи H . На рисунке 2.2 подписаны входы и выходы в подграфы G_w (s_i и e_i соответственно).

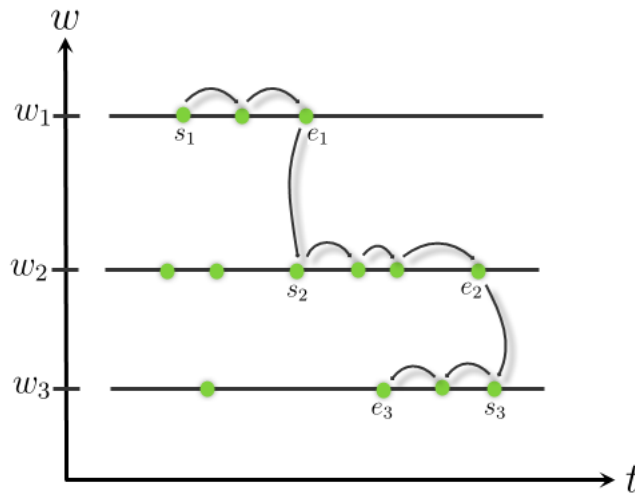


Рис. 2.2: Входы и выходы подграфов G_w

Потребуем, чтобы входы и выходы каждого подграфа обязательно входили в простую цепь, являющуюся решением локальной задачи существования сбалансированной простой цепи. Это требование позволяет обеспечить связность частей цепей-решения, относящихся к различным подграфам $\{G_{w_i}\}$. Благодаря этому, при формализации локальной задачи существования в виде задачи математического программирования для обеспечения связности всей цепи в целом потребуется описать только

«маршрутные» ограничения на порядок следования узлов в рамках отдельных подграфов.

Заметим, что входом и выходом в подграфе может быть один и тот же узел p . В этом случае часть простой цепи-решения, относящаяся к данному подграфу будет состоять из единственного узла p .

Заметим, что для того, чтобы поставить локальную задачу существования сбалансированной простой цепи с фиксированными входами и выходами не обязательно иметь какую-либо простую цепь H . Можно выбирать последовательность подграфов $\{G_{w_i}\}$ и входы и выходы в них произвольным образом. Однако на практике решению этой задачи предшествует выбор цепи, являющейся начальным приближением.

Итак, локальная задача существования сбалансированной простой цепи с фиксированными входами и выходами формулируется следующим образом.

Пусть заданы

- граф предшествования партий G ,
- последовательность подграфов $\{G_{w_i}\}$,
- пары узлов s_i, e_i — вход и выход для каждого подграфа,
- ограничения предшествования узлов в рамках отдельных подграфов,
- ограничения на суммарный вес узлов и на суммарный вес по каждому виду продукции.

Существует ли сбалансированная простая цепь при заданных ограничениях?

2.1.2 Задача со свободными входами и выходами

Зафиксируем последовательность подграфов $\{G_{w_i}\}$ в порядке уменьшения значения w_i . Цепь, являющаяся решением локальной задачи существования должна в обязательном порядке пройти через каждый из выбранных подграфов. При этом не будем фиксировать, какие именно узлы должны быть входами или выходами. Наличие фиксированной последовательности подграфов $\{G_{w_i}\}$ позволяет при формализации локальной задачи существования описать только «маршрутные» ограничения на порядок следования узлов в рамках отдельных подграфов и ограничения на порядок следования узлов между соседними подграфами в последовательности.

Пусть заданы

- граф предшествования партий G ,
- последовательность подграфов $\{G_{w_i}\}$,
- ограничения предшествования узлов в рамках отдельных подграфов,
- ограничения предшествования для узлов соседних подграфов,
- ограничения на суммарный вес и вес по каждому виду продукции.

Существует ли сбалансированная простая цепь при заданных ограничениях?

2.2 Исходная задача существования сбалансированной простой цепи

Для того, чтобы формализовать исходную задачу существования простой цепи во всем графе G , рассматривается последовательность всех подграфов $\{G_{w_i}\}$ в порядке уменьшения значения w_i , при этом не накладывается требование на то, чтобы цепь-решение в обязательном порядке проходила через каждый подграф последовательно, и не накладывается требование на то, какие именно узлы должны быть входами и выходами в подграфах. В этом случае при формализации задачи существования сбалансированной простой цепи в виде задачи математического программирования требуется описать как ограничения на порядок следования узлов в рамках отдельных подграфов, так и ограничения на порядок следования узлов между любыми подграфами, а не только соседними, как это было в локальной задаче существования сбалансированной простой цепи со свободными входами и выходами.

Пусть заданы

- граф предшествования партий G ,
- последовательность подграфов $\{G_{w_i}\}$,
- ограничения предшествования для связи всех подграфов,
- ограничения предшествования в рамках подграфов,
- ограничения на суммарный вес и вес по каждому виду продукции.

Существует ли сбалансированная простая цепь при заданных ограничениях?

Глава 3

Формализация в виде задачи математического программирования

Пусть $G_{w_1}, G_{w_2}, \dots, G_{w_m}$ — набор подграфов графа предшествования партий G с множеством вершин P .

Пронумеруем вершины подграфов в порядке увеличения значения функции t . Пусть p_{ij} — j -я вершина в i -м подграфе. Каждой вершине p_{ij} поставим в соответствие булеву переменную b_{ij} по следующему правилу.

$$b_{ij} = \begin{cases} 1, & \text{если } p_{ij} \text{ входит в цепь-решение;} \\ 0, & \text{если } p_{ij} \text{ не входит в цепь-решение.} \end{cases}$$

Обозначим

- $n_i = |P(G_{w_i})|$ — количество вершин в подграфе G_{w_i} ,
- $l_{ij} = l(p_{ij})$ — вес вершины p_{ij} ,
- L_{min}, L_{max} — заданные минимальное и максимальное значения суммарного веса вершин в цепи-решении,
- f_1, f_2, \dots, f_s — набор видов продукции,
- $\mu_i = \mu(f_i) = (\underline{\mu}_i, \bar{\mu}_i)$ — распределение видов продукции, заданное допустимым диапазоном значения суммарного веса вершин, имеющих f_i вид продукции,
- $f_{ij} = f(p_{ij})$ — вид продукции, соответствующий вершине p_{ij} .

Ограничение на суммарный вес цепи

$$L_{min} \leq \sum_{i=1}^m \sum_{j=1}^{l_i} b_{ij} l_{ij} \leq L_{max} \quad (3.1)$$

Ограничение на суммарный вес по видам продукции

$$\forall k \in \overline{1, s} \quad \underline{\mu}_k \leq \sum_{f_{ij}=f_k} b_{ij} l_{ij} \leq \bar{\mu}_k \quad (3.2)$$

Указанные «объемные» ограничения являются базовыми, они присутствуют в любой постановке задачи существования сбалансированной простой цепи. Способы формализации «маршрутных» ограничений зависят от выбранной постановки. Прежде чем перейти к их описанию введем некоторые вспомогательные обозначения.

Дополнительные обозначения

Пусть p, q и r — вершины подграфа G_{w_i} , то есть $p, q \in P_{w_i}$. Введем вспомогательные понятия.

- Вершина p *левее* вершины q (обозначим $p < q$), если $t_p < t_q$.
- Вершина p *не правее* вершины q (обозначим $p \leq q$), если $t_p \leq t_q$.
- Вершина p *между* вершинами r и q если $r \leq p$ и $p \leq q$.
- Вершина p *правее* вершины q (обозначим $p > q$), если $t_p > t_q$.
- Вершина p *не левее* вершины q (обозначим $p \geq q$), если $t_p \geq t_q$.

Обозначим множество вершин подграфа G_{w_i} , находящихся правее вершины p через $P_i^{>,p}$, зафиксируем множество индексов этих вершин и обозначим его через $J_i^{>,p}$.

- $P_i^{>,p} = \{q \in P_{w_i} \mid q > p\}$
- $J_i^{>,p} = \{j \mid p_{ij} \in P_i^{>,p}\}$

Обозначим множество вершин подграфа G_{w_i} , находящихся правее вершины p и смежных с ней через $P_i^{>,p,o}$, зафиксируем множество индексов этих вершин и обозначим его через $J_i^{>,p,o}$.

- $P_i^{>,p,o} = \{q \in P_{w_i} \mid q > p, |t_q - t_p| \leq \min\{r(t_q), r(t_p)\}\}$
- $J_i^{>,p,o} = \{j \mid p_{ij} \in P_i^{>,p,o}\}$

Аналогичные обозначения можно ввести для множества вершин подграфа G_{w_i} , находящихся левее вершины p , не правее вершины p и не левее вершины p .

Множество вершин подграфа G_{w_i} , находящихся между вершинами p и q (предположим, $p < q$) обозначим через $P_i^{p,\geq,q}$. Соответственно множество индексов этих вершин $J_i^{p,\geq,q}$.

Замечание 1. В подграфе G_{w_i} наличие ребра между двумя вершинами проверяется лишь ограничением «для перехода по толщине». Ограничение «для перехода по ширине» тривиально выполняется, так как значения w всех вершин в графе G_{w_i} равны. В этом случае, когда две вершины p и q не связаны ребром и лежат между вершинами p' и q' , то p' и q' также не связаны ребром. В самом деле, не ограничивая общности можно считать, что вершина p не правее узла q и вершина p' не правее узла q' . Тогда $r_{t_{p'}} \leq r_{t_p} \leq r_{t_q} \leq r_{t_{q'}}$, отсюда $|t_{p'} - t_{q'}| \geq |t_p - t_q| > \min\{r(t_p), r(t_q)\} \geq \min\{r(t_{p'}), r(t_{q'})\}$. Следовательно, согласно определению p' и q' не связаны ребром.

Аналогично можно показать, что если две вершины p и q связаны ребром, то находящиеся между ними вершины p' и q' также связаны ребром.

3.1 Локальные задачи существования сбалансированной простой цепи

3.1.1 Задача с фиксированными входами и выходами

- G_{w_i} — подграф графа G , s_i, e_i — заданная пара вершин, вход и выход для подграфа G_{w_i} . Допускается, что вход и выход это одна и та же вершина.

Пусть $p \in P$, соответствующую вершине p переменную обозначим b^p .

Тогда b^{s_i} , b^{e_i} это переменные, соответствующие входу и выходу. Потребуем, чтобы вход и выход были включены в цепь-решение.

$$b^{s_i} = 1, \quad b^{e_i} = 1 \quad (3.3)$$

- Пусть $p_{ij_0} \in G_{w_i}$ — отличная от входа и выхода вершина, то есть $p_{ij_0} \neq s_i$ и $p_{ij_0} \neq e_i$.

Замечание 2. Поскольку в графе G_{w_i} любые две вершины p и q связаны ребром pq тогда и только, когда они связаны ребром qp , то для любой простой цепи с обратным порядком вершин также существует. Следовательно, не ограничивая общности, далее можно считать, что вход всегда не правее выхода, поскольку в противном случае вход и выход можно поменять местами.

Если $s_i < p_{ij_0} < e_i$, то

$$b_{ij_0} \leq \sum_{j' \in J_i^{\circ, p}} b_{ij'}, \quad b_{ij_0} \leq \sum_{j' \in J_i^{\circ, p}} b_{ij'} \quad (3.4)$$

Это означает, что вершина p_{ij_0} , находящаяся между входом и выходом может быть включена в цепь-решение, если среди смежных с ней вершин в цепь-решение включена хотя бы одна вершина, находящаяся правее, и хотя бы одна вершина, находящаяся левее p_{ij_0} .

Если $p_{ij_0} < s_i$, то

$$2b_{ij_0} \leq \sum_{j' \in J_i^{\circ, p}} b_{ij'} \quad (3.5)$$

Это означает, что вершина p_{ij_0} , находящаяся левее входа может быть включена в цепь-решение, если среди смежных с ней вершин в цепь-решение включены хотя бы две вершины, находящиеся правее p_{ij_0} .

Если $e_i < p_{ij_0}$, то

$$2b_{ij_0} \leq \sum_{j' \in J_i^{\circ, p}} b_{ij'} \quad (3.6)$$

Это означает, что вершина p_{ij_0} , находящаяся правее выхода может быть включена в цепь-решение, если среди смежных с ней вершин в цепь-решение включены хотя бы две вершины, находящиеся левее p_{ij_0} .

Если $t_{p_{ij_0}} = t_{s_i}$ или $t_{p_{ij_0}} = t_{e_i}$, то ограничение на возможность включения вершины p_{ij_0} в цепь-решение не накладывается.

Локальная задача существования состоит в том, чтобы найти решение следующей системы уравнений и неравенств, либо установить, что его не существует.

$$\left\{ \begin{array}{l}
L_{min} \leq \sum_{i=1}^m \sum_{j=1}^{n_i} b_{ij} l_{ij} \leq L_{max}; \\
\forall k \in \overline{1, s} \quad \underline{\mu}_k \leq \sum_{f_{ij}=f_k} b_{ij} l_{ij} \leq \overline{\mu}_k; \\
\forall i \in \overline{1, m} : \\
\left\{ \begin{array}{l}
b^{s_i} = 1, \quad b^{e_i} = 1; \\
\forall j_0 \in \overline{1, n_i} (p_{ij_0} \notin \{s_i, e_i\}) : \\
b_{ij_0} \leq \sum_{j' \in J_i^{>, p}} b_{ij'}, \quad b_{ij_0} \leq \sum_{j' \in J_i^{<, p}} b_{ij'}, \quad \text{если } s_i < p_{ij_0} < e_i; \\
2b_{ij_0} \leq \sum_{j' \in J_i^{>, p}} b_{ij'}, \quad \text{если } p_{ij_0} < s_i; \\
2b_{ij_0} \leq \sum_{j' \in J_i^{<, p}} b_{ij'}, \quad \text{если } e_i < p_{ij_0}.
\end{array} \right.
\end{array} \right. \quad (*)$$

Решение вложенной подсистемы неравенств (*) означает, что из вершин, вошедших в решение, принадлежащих одному подграфу вместе с входом и выходом, возможно построить простую цепь в G с началом во входной вершине и окончанием в выходной вершине. Возможность собрать все вершины в решении локальной задачи в одну простую цепь гарантируется в том случае, если входы и выходы в следующих друг за другом подграфах выбраны согласованно.

Лемма 1. *В подграфе G_w существует простая (s, e) -цепь тогда и только тогда, когда существует решение системы (*).*

Доказательство. \Rightarrow Пусть в G_{w_i} существует простая (s_i, e_i) -цепь H , покажем, что тогда существует решение системы (*). Рассмотрим набор переменных $\{b_{ij}\}$, соответствующих вершинам G_{w_i} . Присвоим переменным значения в соответствии с цепью H . Если вершина p_{ij} входит в цепь H , то присвоим $b_{ij} = 1$, в противном случае присвоим $b_{ij} = 0$.

Рассмотрим $j_0 \in \overline{1, n_i}$.

Если $p_{ij_0} \notin P(H)$, то все неравенства, соответствующие переменной b_{ij_0} выполнены, поскольку в их левой части стоит 0, а в правой части стоит сумма неотрицательных чисел.

Пусть $p_{ij_0} \in P(H)$. Это значит, что $b_{ij_0} = 1$.

- Если $p_{ij_0} < s_i$, то рассмотрим множества $P_i^{\leq, p_{ij_0}}$ и $P_i^{>, p_{ij_0}}$. Поскольку простая (s, e) -цепь H посещает вершину $p_{ij_0} \in P_i^{\leq, p_{ij_0}}$ и заканчивается в вершине $e_i \in P_i^{>, p_{ij_0}}$, то среди ребер цепи H найдется пара ребер, соединяющих эти множества. То есть $\exists u_1, u_2 \in P_i^{\leq, p_{ij_0}} \cup P(H)$ и $\exists v_1, v_2 \in P_i^{>, p_{ij_0}} \cup P(H)$ ($v_1 \neq v_2$, так как H — простая), что $u_1 v_1 \in E$ и $u_2 v_2 \in E$. Поскольку $u_1 \leq p_{ij_0} < v_1$ и $u_2 \leq p_{ij_0} < v_2$, то по замечанию 1 $p_{ij_0} v_1 \in E$ и $p_{ij_0} v_2 \in E$. Заметим, что $v_1, v_2 \in P(H)$, значит $b^{v_1} = 1$ и $b^{v_2} = 1$. Таким образом, неравенства, соответствующие переменной b_{ij_0} выполнены, поскольку имеют вид

$$2b_{ij_0} \leq b^{v_1} + b^{v_2} + K,$$

где K — сумма неотрицательных чисел.

- Случай $e_i < p_{ij_0}$ рассматривается аналогично предыдущему с использованием множеств $P_i^{<,p_{ij_0}}$ и $P_i^{\geq,p_{ij_0}}$.
- Если $s_i < p_{ij_0} < e_i$, то снова рассмотрим множества $P_i^{\leq,p_{ij_0}}$ и $P_i^{>,p_{ij_0}}$. Поскольку простая (s, e) -цепь H посещает вершину $p_{ij_0} \in P_i^{\leq,p_{ij_0}}$ и заканчивается в вершине $e_i \in P_i^{>,p_{ij_0}}$, то среди ребер цепи H найдется ребро, соединяющее эти множества. То есть $\exists u \in P_i^{\leq,p_{ij_0}} \cup P(H)$ и $\exists v \in P_i^{>,p_{ij_0}} \cup P(H)$, что $uv \in E$. Поскольку $u \leq p_{ij_0} < v$, то по замечанию 1 $up_{ij_0} \in E$ и $p_{ij_0}v \in E$. Заметим, что $u, v \in P(H)$, значит $b^u = 1$ и $b^v = 1$. Таким образом, неравенства, соответствующие переменной b_{ij_0} выполнены, поскольку имеют один из двух видов

$$b_{ij_0} \leq b^u + K_1, \quad b_{ij_0} \leq b^v + K_2,$$

где K_1 и K_2 — суммы неотрицательных чисел.

- В случае $t_{p_{ij_0}} = t_{s_i}$ или $t_{p_{ij_0}} = t_{e_i}$ переменной b_{ij_0} не соответствует никаких неравенств.

Таким образом, набор значений переменных b_{ij_0} , выбранный в соответствии с цепью H , удовлетворяет всем неравенствам и является решением системы (*).

\Leftarrow Пусть существует решение системы (*), покажем, что в подграфе G_{w_i} существует простая (s_i, e_i) -цепь. Выберем множество вершин подграфа G_{w_i} , которым соответствуют переменные, имеющие значение 1, обозначим его M . Множество M не пусто, поскольку существует хотя бы одна ненулевая переменная $b^{s_i} = 1$ или $b^{e_i} = 1$, возможно это одна и та же переменная, если $s_i = e_i$.

В случае, если $s_i = e_i$, то простая (s_i, e_i) -цепь состоит из одной вершины s_i .

Рассмотрим случай, когда $s_i \neq e_i$. Начнем строить простую цепь (s_i, e_i) -цепь H по следующему алгоритму.

1. Вначале положим $H := (s_i)$.
2. Будем просматривать вершины $p \in M$, такие что $s_i \leq p \leq e_i$, в порядке увеличения значения t .
 - (а) Если очередная вершина p имеет $t_p = t_{s_i}$, то p и s_i смежны, добавляем p в конец цепи H . Повторяем пункт 2(а) пока остаются непросмотренные
 - (б) Далее рассмотрим первую вершину p ,

□

3.1.2 Задача со свободными входами и выходами

В рамках этой задачи также рассматривается фиксированная последовательность подграфов $\{G_{w_i}\}$. Отличие от предыдущей задачи в том, что теперь не зафиксировано то, какие вершины должны быть входами и выходами, а значит, необходимо ввести ограничения, описывающие связность цепи-решения между соседними подграфами.

Добавляется два набора переменных. Каждой вершине p_{ij} соответствуют

$$s_{ij} = \begin{cases} 1, & \text{если } p_{ij} \text{ является входом в } G_{w_i}; \\ 0, & \text{если } p_{ij} \text{ не является входом в } G_{w_i}; \end{cases}$$

$$e_{ij} = \begin{cases} 1, & \text{если } p_{ij} \text{ является выходом из } G_{w_i}; \\ 0, & \text{если } p_{ij} \text{ не является выходом из } G_{w_i}. \end{cases}$$

В задаче со свободными входами и выходами рассматривается следующий ряд ограничений.

В каждом подграфе должен быть ровно один вход и ровно один выход, то есть

$$\forall i \sum_j s_{ij} = 1, \quad \forall i \sum_j e_{ij} = 1 \quad (3.7)$$

Вершины, являющиеся входами или выходами должны быть включены в цепь-решение, то есть

$$\forall i \forall j b_{ij} \geq s_{ij}, \quad \forall i \forall j b_{ij} \geq e_{ij} \quad (3.8)$$

Рассмотрим подграф G_{w_i} . Предположим, что существует вершина x , имеющая значение функции $t = t_x$ и $w = w_i$. Обозначим множество узлов подграфа G_{w_i} смежных с вершиной x через $P_i(t_x)$.

$$P_i(x) = \{p \in P_{w_i} \mid |t_p - t_x| \leq \min\{r(t_p), r(t_x)\}\}$$

Зафиксируем множество индексов этих узлов.

$$J_i(x) = \{j \mid p_{kj} \in P_i(t_x)\}$$

Смысл следующего ограничения в том, что для каждой вершины-входа в подграф, среди вершин-предков этого входа, принадлежащих предыдущему подграфу, должна существовать вершина, являющийся выходом из предыдущего подграфа.

$$\forall i = 2, \dots, m \forall j_0 s_{ij_0} \leq \sum_{j \in P_{i-1}(p_{ij_0})} e_{i-1j} \quad (3.9)$$

По сравнению с задачей с фиксированными входами и выходами ограничения предшествования в рамках отдельных подграфов (система $(*)$) заменяются новыми (система $(**)$).

$$\begin{cases} 2b_{ij_0} \leq \sum_{j' \in J_i^{>, p_{ij_0}} b_{ij'} + (2 - \sum_{j \in J_i^{>, p_{ij_0}} (s_{ij} + e_{ij})) \\ 2b_{ij_0} \leq \sum_{j' \in J_i^{<, p_{ij_0}} b_{ij'} + (2 - \sum_{j \in J_i^{<, p_{ij_0}} (s_{ij} + e_{ij})) \end{cases} \quad (**)$$

Лемма 2. В подграфе G_w существует простая (s, e) -цепь тогда и только тогда, когда существует решение системы $(**)$.

3.2 Исходная задача существования сбалансированной простой цепи

В предыдущей задаче была задана фиксированная последовательность подграфов, в том смысле что для построения цепи-решения необходимо было, чтобы в нее были включены вершины из каждого подграфа последовательности. Теперь же сформулируем ограничения для связности подграфов таким образом, чтобы избежать этой необходимости. Это позволит рассматривать произвольные последовательности подграфов, то есть, перейти от локальных задач существования к исходной.

Для каждого подграфа G_{w_i} графа G поставим в соответствие переменную

$$q_i = \begin{cases} 1, & \text{если подграф } G_{w_i} \text{ включен в цепь-решение (хотя бы 1 узел из } G_{w_i}); \\ 0, & \text{иначе.} \end{cases}$$

В исходной задаче существования сбалансированной простой цепи рассматривается следующий ряд ограничений.

Если подграф G_{w_i} не включен в цепь-решение, то ни один его узел не представлен в решении.

$$\forall i \forall j \ b_{ij} \leq q_i \quad (3.10)$$

Если хотя бы один узел подграфа G_{w_i} включен в цепь-решение, то подграф включен в решение.

$$\forall i \ q_i \leq \sum_j b_{ij} \quad (3.11)$$

Если в подграфе есть вход и есть выход, то подграф включен в решение.

$$\forall i \ q_i = \sum_j e_{ij} = \sum_j s_{ij} \quad (3.12)$$

Входы и выходы в подграф должны быть включены в цепь-решение.

$$\forall i \forall j \ b_{ij} \geq s_{ij}, \quad \forall i \forall j \ b_{ij} \geq e_{ij} \quad (3.13)$$

Если подграф включен в решение, то среди вершин-предков входа в этот подграф должны быть включенные в решение вершины.

$$s_{ij} \leq \sum_{k < l < i} q_l + \sum_{z \in In_{B_k}(p_{ij})} e_{kz} + (1 - q_k) \quad (3.14)$$

Ограничения предшествования вершин в рамках подграфов такие же как и в локальной задаче существования со свободными входами и выходами.

$$\begin{cases} 2b_{ij_0} \leq \sum_{j' \in J_i^{>, p_{ij_0}} o} b_{ij'} + (2 - \sum_{j \in J_i^{>, p_{ij_0}}} (s_{ij} + e_{ij})) \\ 2b_{ij_0} \leq \sum_{j' \in J_i^{<, p_{ij_0}} o} b_{ij'} + (2 - \sum_{j \in J_i^{<, p_{ij_0}}} (s_{ij} + e_{ij})) \end{cases} \quad (**)$$

Глава 4

Программная реализация

В работе использован язык программирования Java.

4.1 Основные структуры данных

Для начала определим основные структуры данных, используемые в алгоритме поиска блоков графа G .

1. *Vertex* p представляет вершину p в графе G . Определим для такой вершины
 - $p.width$ - значение функции w , то есть ширина соответствующей партии p .
 - $p.thick$ - значение функции t , то есть толщина соответствующей партии p .
 - $p.measure$ - значение функции l , то есть полезность соответствующей партии p .
 - $p.flowType$ - вид готовой продукции, которой станет партия p после прокатки.
 - $p.readyTime$ - время, начиная с которого партия p готова к прокатке.
 - $p.rollDuration$ - длительность прокатки партии p .
2. *Crit* $crit$ предназначен для проверки того, может ли вершина q следовать за вершиной p в графике прокатки. *crit* содержит в себе следующие функции:
 - $widthCrit(Vertex\ from, Vertex\ to)$ - функция, отвечающая на вопрос "удовлетворяют ли ограничению «для перехода по ширине» вершины $from$ и to ?"
 - $thicknessCrit(Vertex\ from, Vertex\ to)$ - функция, отвечающая на вопрос "удовлетворяют ли ограничению «для перехода по толщине» вершины $from$ и to ?"
 - $test(Vertex\ from, Vertex\ to)$ - функция, отвечающая на вопрос "может ли вершина to непосредственно следовать за вершиной $from$ в графике прокатки?". Данная функция проверяет, что для различных вершин $from$ и to выполняются с положительными результатами функции $widthCrit(from, to)$ и $thicknessCrit(from, to)$. Таким образом, функция $test(from, to)$ определяет набор ребер в графе G .
3. *Level* l представляет граф в одной ширине G_w^o с вершинами $l.vertex$, шириной $l.width = w$ и размером $l.size$, равным количеству вершин в *vertex*. Также для графа в одной ширине определим вершины *Vertexenter* и *Vertexexit*, представляющие собой вход и выход из графа G_w^o в графике прокатки.

4.2 Загрузка задания

Класс *RollingShedullingTaskrst* определяет задание для расчета. Для *rst* определим:

- *rst.vertex* - набор партий из задания. Каждая партия представляет собой вершину *Vertexv*.
- *rst.totalLength* - суммарная длина партий из задания.
- *rst.flows* - набор видов готовой продукции, которые должны получиться после прокатки. Для каждой вершины *v* из *vertex* во *flows* найдется *v.flowType*.
- *rst.flowsTotalLength* - целевое значение суммарной длины партий по каждому виду продукции
- *rst.levels* - набор графов в одной ширине (*Levell*) на множестве вершин *rst.vertex*
- *rst.levelAmount* - количество графов в одной ширине
- *rst.Lmin* - минимальное значение суммарной длины партий в графике прокатки
- *rst.Lmax* - максимальное значение суммарной длины партий в графике прокатки
- *rst.startTime* - время начала прокатки
- *rst.prepareTask()* - функция, которая делает задание корректным в случае противоречий. (Например, если целевая суммарная длина графика прокатки больше общей суммарной длины всех партий и т.д.)
- *RollingSchedulingTask()* - загрузка задания из базы данных в программу для дальнейших расчетов.

4.3 Модель

1. *Problempr*. Для *p* определим

- *p.varArrays* - набор переменных?
- *p.constraints* - набор ограничений (неравенств?)
- *p.constrAmount* - количество неравенств
- *p.objectiveFunction* - целевая функция?

2. *RollingMathModelmodel*

3. *RollingMathOptFuncfunc*

4.4 Перевод задания в неравенства

Класс *RollingShedullingProblemrsp* предназначен для перевода задания из *RollingShedullingTaskrst* в задачу математического программирования для ее дальнейшего решения. Для *rsp* определим:

- *createProblem()* - функция, создающая по заданию *rst* задачу *Problempr*.


```

function createProblem(rst, model, crit, func):Problem p
  var Problem p
  begin
    if () then
      ...
    if () then
      ...
    return p
  end

```

4.5 Результаты экспериментов

Локальная задача существования может быть использована для получения решений и в более компактной схеме. В предыдущей работе [3] предложен алгоритм кубической временной трудоемкости, решающий задачу поиска пути максимального веса. По такому пути естественным образом строится набор блоков с входами и выходами. Используя эти данные в качестве начальных в локальной задаче существования иногда возможно построить решение. Вычислительные эксперименты, проведенные с использованием системы IBM ILOG CPLEX, показали, что зачастую таким коротким путём возможно получить решение локальной задачи существования и, соответственно, решение исходной задачи. При этом решение получается за короткое время при числе партий порядка 100. При этом оптимальное решение локальной задачи зачастую строится за 1-2 секунды.

Данные для тестов берутся из корпоративной сети ОАО "ММК". Обычно в распоряжении имеются выборки порядка 200-300 партий.

Так же отметим, что программа работает достаточно быстро, что важно, потому что в реальной жизни требуется многократный запуск алгоритма. Видно, что скорость работы программы довольно сильно зависит от структуры конкретных начальных данных. Имеет значение соотношение между количеством подграфов в исходном графе и количеством входящих в них узлов. Например в случае, когда исходный граф состоит из малого числа подграфов, то время работы уменьшается за счет сокращения перебора дуг между подграфами (выборка 6).

Тесты проводились на персональном компьютере.

4.6 Инструкция пользователю

Заключение

Литература

- [1] *Леонова С.И.* Задача о формировании графика работы стана горячей прокатки, сбалансированного по составу продукции, 2016
- [2] *Леонова С.И.* Разработка и исследование алгоритмов построения экстремальных цепей в вершинно-взвешенных ориентированных графах. Частный алгоритм, 2014
- [3] *Березин А.А.* Разработка и исследование алгоритмов построения экстремальных цепей в вершинно-взвешенных ориентированных графах. Общий алгоритм, 2014
- [4] *Dantzig G.B., Fulkerson D.R., Johnson S.* On a linear programming combinatorial approach to the traveling salesman problem, *Operations Research*, 7 (1959), pp. 58–66
- [5] *Balas E., Clarence H. M.* Combinatorial optimization in steel rolling. Workshop on Combinatorial Optimization in Science and Technology, April, 1991.
- [6] *Lixin Tang, Jiyin Liu, Aiyong Rong, Zihou Yang.* A review of planning and scheduling systems and methods for integrated steel production. *European Journal of Operational Research*. Volume 133, Issue 1, 16 August 2001, PP. 1–20.
- [7] *Shixin Liu.* Model and Algorithm for Hot Rolling Batch Planning in Steel Plants. *International Journal of Information and Management Sciences*. 21 (2010), PP. 247-263.

Приложение

Пример исходного кода