

ГЛАВА 1

НЕОГРАНИЧЕННАЯ ЗАДАЧА РАЗМЕЩЕНИЯ СРЕДСТВ ОБСЛУЖИВАНИЯ

Неограниченная задача размещения производства или по иному неограниченная задача размещения средств обслуживания — центральная задача семейства дискретных задач размещения. Она имеет достаточно простые математические формулировки и, вместе с тем, является труднорешаемой задачей, которую по интенсивности изучения и использования можно отнести к «основному ядру» списка известных труднорешаемых экстремальных задач.

Математически неограниченная задача размещения производства формулируется как задача целочисленного или смешанного линейного программирования на минимум или максимум. Возможен и комбинаторный вариант записи задачи размещения в виде задачи минимизации функции, заданной на подмножествах некоторого конечного множества. Указанные варианты записи задачи преобразуются друг в друга в результате несложных замен переменных. При этом оптимальные значения одних переменных достаточно просто строятся по оптимальным значениям других переменных.

Нетривиальными трансформациями задачи размещения средств обслуживания являются эквивалентные формулировки этой задачи в виде других известных экстремальных задач. К числу таких задач, прежде всего, относится хорошо известная задача о покрытии множествами и менее известная и изученная так называемая задача минимизации псевдобулевых функций или, другими словами, задача минимизации полиномов от булевых переменных.

Впервые внимание к задаче отыскания экстремальных значений функций от булевых переменных, названных псевдобулевыми функциями, было привлечено работами П. Хаммера (P. Hammer) [166,167], в которых, в частности, указано на многочисленные приложения этой задачи к дискретной оптимизации, теории игр, теории графов и т.д. Возможность представления неограниченной задачи размещения производства в виде задачи минимизации псевдобулевой функции также была впервые подмечена П. Хаммером [169]. Независимо от этого в [13] дано представление задачи размещения средств обслуживания в виде задачи минимизации полиномов от булевых переменных с неотрицательными коэффициентами при нелинейных членах, а в последующих работах [17, 19] показано, что задача минимизации полинома может быть представлена в виде задачи размещения средств обслуживания и что эти задачи эквивалентны. Важное вспомогательное значение при исследовании задачи размещения и задачи минимизации полиномов от булевых переменных играет задача о покрытии множествами, которая хотя и эквивалентна указанным выше задачам, не может рассматриваться как еще одна форма записи задачи размещения средств обслуживания. В некотором смысле задачу о покрытии множествами можно считать менее сложной, чем задача размещения.

Неограниченная задача размещения, как следует из приведенной ее содержательной интерпретации, имеет большое прикладное значение и может быть использована при выборе мест размещения предприятий, складов, ремонтных станций и т.д.

Кроме этой содержательной интерпретации задачи возможны и другие ее интерпретации не менее интересные с практической точки зрения. Задача размещения средств обслуживания может рассматриваться как математическая формулировка так называемой задачи выбора оптимального состава системы технических средств. Такая интерпретация менее известна, чем классическая содержательная постановка, однако она не менее важна, поскольку описывает многочисленные практические ситуации выбора типажа и состава оборудования, планирования развития парка машин, решения вопроса о целесообразности производства изделий нового образца и т.д. Задача размещения может рассматриваться также как математическая модель при решении некоторых вопросов стандартизации, в частности, при обосновании так называемых параметрических (типоразмерных) рядов изделий.

Поскольку основная цель данной работы — построение работоспособных алгоритмов решения рассматриваемых экстремальных задач, то в §1 данной главы даются основные определения, связанные с понятиями «задача», «алгоритм», «сложность». При этом уровень формализации определений не всегда будет математически строгим, но достаточным для целей настоящего исследования.

В §2 дается подробная содержательная постановка неограниченной задачи размещения средств обслуживания как в «классическом» варианте, так и в виде задачи выбора оптимального состава системы технических средств. Приводятся различные варианты эквивалентных записей задачи и делаются основные предположения о виде исходных данных задачи. Рассматривается также некоторое обобщение неограниченной задачи размещения, так называемая нелинейная задача размещения, которая при естественных предположениях сводится к неограниченной задаче размещения.

В §3 рассматривается задача минимизации полиномов от булевых переменных и устанавливается тесная взаимосвязь между неограниченной задачей размещения и задачей минимизации полиномов от булевых переменных с неотрицательными коэффициентами при нелинейных членах. С использованием этой связи определяются классы эквивалентных исходных данных задачи, обладающих тем свойством, что оптимальные решения задач с эквивалентными входными данными совпадают.

§4 посвящен исследованию связи между неограниченной задачей размещения, задачей о покрытии множества системой подмножеств и частным случаем этой задачи — задачей о вершинном покрытии. Приводятся формулировки задачи размещения и задачи минимизации полиномов от булевых переменных с неотрицательными коэффициентами в виде задачи о покрытии множествами и задачи о вершинном покрытии.

1 Экстремальные задачи, алгоритмы, сложность

Вопросы сложности неограниченных задач размещения средств обслуживания и алгоритмов решения таких задач, как уже отмечалось, занимают в дальнейшем изложении одно из центральных мест. Поэтому ниже напомним основные определения, связанные с понятиями «экстремальная задача» и «алгоритм». При этом будем использо-

вать уровень формализации определений, соответствующий целям нашего исследования, которые в большей степени предполагают разработку работоспособных алгоритмов решения задач, чем доказательство того, что для тех или иных задач не существует алгоритмов решения с теми или иными свойствами. Для исследования вопросов сложности необходимо более или менее строго определить такие понятия как «задача», «алгоритм», «время работы», «быстрый алгоритм» и т.д. Мы не будем, как отмечено выше, давать математически строгие определения этих понятий, а ограничимся в некоторых случаях интуитивными представлениями, достаточными, тем не менее, для того, чтобы дать определения таким важным сложностным классам задач как **P** и **NP**. Эти множества на интуитивном уровне представляют собой соответственно класс задач, каждую из которых можно быстро решить, и класс задач, для каждой из которых можно быстро проверить решена она или нет. Вопросам сложности посвящена обширная литература. Более подробное и строгое изложение можно найти, например, в работах [40, 56, 61]. Отметим также обстоятельную монографию [49], являющуюся прекрасным введением в теорию сложности.

1.1 Задачи и алгоритмы

Неформально *задачей* является множество однотипных вопросов или заданий, в которых требуется по известным исходным данным определить требуемое решение или убедиться, что его не существует. Для формализации понятия «задача» необходима соответствующая формализация основных элементов задачи: исходных данных и результата. Универсальной формой такого задания являются слова в некотором *алфавите*. *Словом* в алфавите Σ называется конечная упорядоченная последовательность символов из алфавита Σ , а *языком* — некоторое множество слов. Алфавит может быть конечным и состоять, например, только из двух символов 0 и 1, а может иметь бесконечное число элементов, как, например, множество действительных чисел.

Для любого алфавита существуют стандартные способы записи в виде слов таких объектов как векторы, матрицы, графы, системы линейных неравенств и т.д. При этом представления этих объектов, базирующихся на естественных алфавитах, оказываются эквивалентными, поскольку могут быть быстро преобразованы друг в друга. Использование разных алфавитов может привести лишь к различной длине слов, отображающих кодируемые объекты. Однако для наших целей эти различия в длине слов не имеют принципиального значения, поэтому в качестве основного алфавита будем использовать алфавит действительных чисел. Этот алфавит позволяет, вероятно, реализовывать самый простой вариант кодировки интересующих нас объектов, при котором всякое число кодируется одним символом, вектор длины n — словом длины n , матрица размера $m \times n$ — словом длины mn и т.д.

С формальной точки зрения, под *задачей (поиска)* понимается подмножество $\Pi \subseteq P \times Z$, где P — множество слов в алфавите Σ , задающих множество возможных входных данных, а Z — множество слов в алфавите Σ , кодирующих возможные решения. При этом сама задача Π формулируется следующим образом: задано слово $p \in P$, найти

слово $z \in Z$ такое, что $(p, z) \in \Pi$ или убедиться, что такого слова z не существует. Слова p называются *исходными данными*, а слова z – *решениями* задачи Π . Таким образом, задача — это бинарное отношение Π между элементами двух множеств: множеством исходных данных P и множеством решений Z .

Задача Π называется *задачей распознавания*, если $Y = \{\emptyset\}$, то есть если при любых входных данных решением является пустое слово. В этом случае под задачей можно понимать множество $P_0 \subset P$, а саму задачу распознавания формулировать следующим образом: задано слово $p \in P$, выяснить $p \in P_0$.

Пусть $\Pi \in P \times Z$ – некоторая задача и пусть $P' \subset P$. Множество $\Pi' = \{(p, z) \mid p \in P', (p, z) \in \Pi\}$ будем называть *подзадачей* или *частным случаем* исходной задачи Π . В подзадаче Π' сформулирован тот же вопрос, что и в задаче Π , но уже не для всех исходных данных, а только для некоторого их подмножества. В частности, если $p \in P$, то $\Pi'(p) = \{(p, z) \mid (p, z) \in \Pi\}$ будем называть *конкретной задачей* или *примером*, имея в виду, что задача $\Pi'(p)$ – это уже не множество однотипных вопросов, а один конкретный вопрос.

Неформально, *алгоритмом* A над множеством слов-входов P будем называть точные предписания о том, какие действия и в какой последовательности производить, чтобы переработать вход $p \in P$ в требуемый результат $A(p)$. Если $\Pi \in P \times Z$ – некоторая задача, то алгоритм ее решения для любых входных данных $p \in P$ определяет решение $z \in Z$ такое, что $(p, z) \in \Pi$, либо останавливается, если требуемого решения не существует.

Алгоритм можно понимать и как формально описанные предписания о действиях, то есть как *программу* для вычислительной машины или ее модели. На этом базируется формальное определение алгоритма как программы для *машины Тьюринга*.

Для решения каждой задачи может быть предложено несколько алгоритмов, которые могут требовать для своей реализации различных вычислительных усилий. Среди этих алгоритмов естественно выбирать лучшие. Среди множества показателей качества алгоритма принято выделять *временную сложность* или *трудоемкость* алгоритма, характеризующую алгоритм с точки зрения времени работы алгоритма. Этот показатель можно считать наиболее существенным, поскольку ограничения по времени часто являются доминирующим фактором, определяющим практическую пригодность алгоритма. При этом время работы алгоритма измеряется числом элементарных операций, необходимых для реализации алгоритма, а под элементарными операциями понимаются операции сложения, вычитания умножения, деления и операция сравнения.

Для алгоритма A решения задачи $\Pi \in P \times Z$ обозначим через $t(p)$ время работы алгоритма в случае исходных данных $p \in P$. Нас будет интересовать зависимость времени работы алгоритма от размерности $|p|$ исходных данных p . Но поскольку для разных входных данных p_1 и p_2 одинаковой размерности время $t(p_1)$ и $t(p_2)$ работы алгоритма может быть существенно различным, то необходимо использовать некоторое усреднение по всем входным данным одной размерности. При определении временной сложности алгоритма A используется усреднение по худшему случаю, а *временной сложностью* или *трудоемкостью* алгоритма A называют функцию

$$f_A(k) = \max \{t(p) \mid p \in P, |p| = k\}.$$

О том, насколько хорошим является алгоритм A судят по тому, насколько быстро растет временная сложность $f_A(k)$ с ростом длины входных данных. Алгоритм называют *полиномиальным* или *эффективным*, а также говорят, что алгоритм имеет *полиномиальную временную сложность*, если функция $f_A(k)$ полиномиально ограничена, то есть $f_A(k) = O(k^n)$, где n – фиксированная величина, не зависящая от k . Напомним, что обозначение $f(k) = O(g(k))$ используется тогда, когда существует положительная константа C такая, что $f(k) \leq Cg(k)$ при $k \geq k'$. Так же говорят, что задача *полиномиально разрешима* или *разрешима за полиномиальное время*, если для нее существует полиномиальный алгоритм.

1.2 Сложностные классы задач распознавания

Задача распознавания формулировалась выше как задача распознавания множества слов P_0 в множестве слов P . Таким образом, задача распознавания состоит из двух множеств: множества P всех индивидуальных входов и множества P_0 индивидуальных входов с ответом «да». Для таких задач определены сложностные классы, ранжирующие эти задачи по степени трудности.

Первым важным классом задач является класс задач (языков) P , *распознаваемых за полиномиальное время*. Задача P_0 ($P_0 \subset P$), принадлежит классу P , если существует полиномиальный алгоритм A с входами $p \in P$, который дает ответ «да» тогда и только тогда, когда $p \in P_0$.

Другим, возможно более широким сложностным классом задач распознавания является класс языков NP , *проверяемых за полиномиальное время*. Задача P_0 принадлежит классу NP , если для всякого $p \in P_0$ существует слово – *сертификат (решение)* y , $|y| = O(|p|^k)$ и существует полиномиальный алгоритм A с входами (p, y) , который для всякого $p \in P_0$ с помощью соответствующего сертификата y может показать, что $p \in P_0$.

Понятно, что если задача P_0 принадлежит P , то также принадлежит и NP . Таким образом $P \subset NP$. В настоящее время неизвестно, совпадают ли классы P и NP , и этот вопрос является серьезной открытой проблемой, связанной с существованием задач, решения которых не могут быть найдены за полиномиальное время, но решения которых могут быть проверены за полиномиальное время.

Вероятно, наиболее убедительным аргументом в пользу того, что $P \neq NP$ является существование в классе NP еще одного сложностного класса NPC , называемого *классом NP-полных задач* и включающем в себя самые трудные задачи из класса NP . Точный смысл утверждению о том, что одна задача труднее другой позволяет придать понятие сводимости.

Определим сводимость сразу применительно к общей задаче поиска. Задача Π' *полиномиально сводима (сводима по Тьюрингу)* к задаче Π , если существует алгоритм решения задачи Π' , содержащий в качестве подпрограммы алгоритм решения задачи Π .

и являющийся полиномиальным при условии полиномиальности алгоритма решения задачи Π .

Задача P_0 принадлежит классу NPC , если, во-первых, P_0 принадлежит классу NP и, во-вторых, любая задача P'_0 из класса NP сводится к задаче P_0 . Отсюда следует, что если хотя бы одна задача из класса NP будет решена за полиномиальное время, то тем самым все задачи из NP будут решены за полиномиальное время. Поэтому задачи из класса NPC можно считать самыми трудными в NP . Чтобы доказать, что задача P_0 из класса NP является NP -полной задачей, нужно показать, что некоторая задача P'_0 из класса NPC полиномиально сводится к задаче P_0 .

До сих пор сложностные классы задач рассматривались только для задач распознавания. Но поскольку полиномиальная сводимость определена и для задач, лежащих вне класса NP , то введем в рассмотрение самый важный для дальнейшего класс так называемых NP -трудных задач. Задача Π является NP -трудной, если любая задача P'_0 из класса NP полиномиально сводится к задаче Π . Отметим, что в этом определении не предполагается, что задача Π является задачей распознавания или принадлежит классу NP . Здесь в качестве задачи Π может рассматриваться любая задача поиска. Чтобы доказать, что данная задача Π является NP -трудной, достаточно показать, что некоторая задача P'_0 из класса NPC полиномиально сводится к задаче Π .

Если задача Π полиномиально сводится к задаче Π' , а задача Π' полиномиально сводится к задаче Π , то эти задачи называются *полиномиально эквивалентными*. Понятно, что NP -полные задачи полиномиально эквивалентны, а NP -трудные задачи таковыми не являются.

1.3 Экстремальная задача

Экстремальная или оптимизационная задача (на минимум) – это такая задача поиска Π , в которой заданы функции $f(x)$, $x \in X$, и множество $D \subset X$ и требуется найти элемент $x^* \in D$, такой что $f(x^*) \leq f(x)$ для всякого $x \in D$. Функцию $f(x)$ называют *целевой*, элементы множества X – *решениями*, элементы множества D – *допустимыми решениями*, а допустимое решение x^* – *оптимальным решением*. Далее, в некоторых случаях элементы множества D также будем называть решениями, опуская слово «допустимыми». Кроме того величину $f(x^*)$ будем называть *оптимальным значением* целевой функции и будем предполагать, что $f(x^*) > 0$.

Отметим, что введенная терминология не совсем соответствует принятой выше при определении абстрактной задачи Π . В смысле этой терминологии решением оптимизационной задачи является оптимальное решение x^* .

Формально экстремальная задача на минимум записываются следующим образом:

$$\begin{aligned} \min f(x); \\ x \in D; \end{aligned}$$

или

$$\min \{f(x) | x \in D\}$$

Аналогичным образом формулируется и записывается экстремальная задача на максимум.

Говоря об экстремальной задаче будем дополнительно предполагать, что функция $f(x)$, $x \in X$, множество $D \subset X$ и решение $x \in X$ могут быть закодированы конечными векторами p_1 , p_2 и z и что существуют полиномиальный алгоритм A_1 с входом (p_1, z) , вычисляющий для всякого $x \in X$ значение $f(x)$ и полиномиальный алгоритм A_2 с входом (p_2, z) , определяющий для всякого $x \in X$ выполнение включения $x \in D$. С учетом этого замечания исходными данными рассматриваемой экстремальной задачи считаем конечный вектор $p = (p_1, p_2)$, кодирующий функцию $f(x)$, $x \in X$ и множество $D \subset X$, а решением – вектор z^* , кодирующий оптимальное решение x^* .

Сформулированная экстремальная задача, как уже отмечалось, является задачей поиска. *Распознавательный вариант* этой задачи выглядит следующим образом. *Зада- ны* функция $f(x)$, $x \in X$, множество $D \subset X$, неотрицательная величина k . *Существует ли* решение $x \in D$ такое, что $f(x) \leq k$. Эта задача, с учетом сделанного дополнительного предположения о вычислимости функции $f(x)$ и распознавания множества D , является задачей класса **NP**. В качестве сертификата может быть использовано допустимое решение $x \in D$, для которого $f(x) \leq k$. Понятно, что распознавательный вариант экстремальной задачи сводится к самой оптимизационной задаче. Поэтому, если распознавательный вариант принадлежит классу **NPC**, то оптимизационная задача является **NP**-трудной. В силу этого, для доказательства того, что данная экстремальная задача является **NP**-трудной, достаточно показать, что к ней сводится экстремальная задача, распознавательный вариант которой есть задача из класса **NPC**.

Экстремальную задачу называют *дискретной*, если множество D есть конечное множество. Наиболее известным видом экстремальных задач являются *линейные задачи*, в которых функция $f(x)$ есть линейная функция, а множество D есть множество решений системы линейных неравенств (уравнений), называемых множеством *ограничений* экстремальной задачи.

Если решениями линейной задачи являются неотрицательные векторы, то такую задачу называют *задачей линейного программирования* и записывают следующим образом

$$\begin{aligned} \min \sum_{i=1}^m c_i x_i ; \\ \sum_{i=1}^m a_{ij} x_i \geq b_j, j = 1, \dots, n; \\ x_i \geq 0, i = 1, \dots, m. \end{aligned}$$

Если к указанным ограничениям задачи добавляются еще условия

$$x_i \in \{0, 1, 2, \dots\}, i = 1, \dots, m,$$

то полученную задачу называют *задачей целочисленного линейного программирования*. Если эти условия распространяются только на часть переменных, то задачу называют

задачей *смешанного* линейного программирования. Наконец, если условия целочисленности переменных имеют вид

$$x_i \in \{0,1\}, i = 1, \dots, m,$$

то задачу называют еще задачей булева линейного программирования. Последняя задача является задачей дискретного программирования.

Введенные выше понятия сводимости и эквивалентности для общих задач поиска играют особую роль в случае экстремальных задач, поскольку используются не только с целью установления сложности задачи и доказательства того, что данная задача является **NP**-трудной, но и в целях построения алгоритмов решения экстремальных задач. Дело в том, что часто такие алгоритмы оказывается более удобным строить не для непосредственно исследуемой задачи, а для некоторой другой экстремальной задачи, к которой полиномиально сводится исследуемая задача.

Напомним, что согласно общему определению задача Π_1 полиномиально сводится к задаче Π_2 , если существует алгоритм решения задачи Π_1 , содержащий в качестве подпрограммы алгоритм решения задачи Π_2 и являющийся полиномиальным при условии полиномиальности алгоритма решения задачи Π_2 . Рассмотрим некоторую расшифровку данного определения более удобную при доказательстве сводимости одной экстремальной задачи к другой.

Пусть экстремальные задачи Π_1 и Π_2 имеют в качестве множества исходных данных соответственно множества P_1 и P_2 . Согласно приведенному выше определению, сводимость задачи Π_1 к задаче Π_2 будет доказана, если будет показано, что, во-первых, существует полиномиальный алгоритм A_1 с входами $p_1 \in P_1$, который по произвольным исходным данным p_1 задачи Π_1 строит исходные данные $p_2 \in P_2$ задачи Π_2 . Во-вторых, существует полиномиальный алгоритм A_2 с входами (p_2, z_2^*) , который по исходным данным $p_2 = A_1(p_1) \in P_2$ задачи Π_2 и некоторому оптимальному решению z_2^* задачи Π_2 с исходными данными $p_2 = A_1(p_1)$ строит оптимальное решение z_1^* задачи Π_1 с исходными данными p_1 .

При использовании приведенной схемы доказательства сводимости одной экстремальной задачи к другой придется устанавливать оптимальность решения задачи Π_1 , полученного с помощью алгоритма A_2 из оптимального решения задачи Π_2 . Для этого окажется полезным следующий *признак оптимальности*.

Пусть $f_1(x)$ – целевая функция задачи Π_1 с исходными данными p_1 , а $f_2(y)$ – целевая функция задачи Π_2 с входными данными $p_2 = A_1(p_1)$. Пусть y^* – оптимальное решение задачи Π_2 и пусть этому решению поставлено в соответствие допустимое решение x задачи Π_1 . Имеет место следующее достаточное условие оптимальности решения x .

Лемма 1.1. Если $f_2(y^*) \geq f_1(x)$ и для некоторого оптимального решения x^* задачи Π_1 существует допустимое решение y задачи Π_2 такое, что $f_1(x^*) \geq f_2(y)$, то x – оптимальное решение задачи Π_1 .

Доказательство. Справедливость утверждения вытекает из следующей цепочки неравенств

$$f_1(x) \leq f_2(y^*) \leq f_2(y) \leq f_1(x^*).$$

1.4 Алгоритмы решения экстремальных задач

Согласно общему определению алгоритма для задачи поиска, *алгоритм решения экстремальной задачи* – это программа, которая либо преобразует исходные данные в оптимальное решение, либо останавливается, если такого решения не существует. Алгоритм решения экстремальной задачи называется *полиномиальным*, если функция временной сложности алгоритма полиномиально ограничена, а экстремальная задача называется *полиномиально разрешимой*, если для ее решения существует полиномиальный алгоритм.

Для алгоритмов решения экстремальных задач временная сложность является основным показателем качества алгоритма. По виду этой функции алгоритмы делятся на «хорошие» и «плохие». Но для многих экстремальных задач в том числе и рассматриваемых ниже дискретных задач размещения хороших алгоритмов не существует, поскольку эти задачи являются **NP**-трудными. Поэтому при решении таких задач приходится либо обращаться к рассмотрению экспоненциальных алгоритмов, работающих приемлемое время на реальных данных, либо расширять представление об алгоритмах решения экстремальных задач и считать результатом работы таких алгоритмов не только оптимальные, но и допустимые решения.

Приближенным алгоритмом решения экстремальной задачи назовем программу, которая либо преобразует исходные данные в допустимое решение, либо останавливается, если такого решения не существует. Решение, выдаваемое приближенным алгоритмом, назовем *приближенным решением*. Приближенный алгоритм назовем *точным*, если выдаваемое им приближенное решение всегда есть оптимальное решение. Понятно, что точный алгоритм есть алгоритм решения экстремальной задачи в смысле данного ранее определения.

Для приближенных алгоритмов временная сложность остается одной из главных характеристик. При этом, хотя приближенные алгоритмы, как правило, полиномиально ограниченную временную сложность, важное значение приобретает показатель степени полинома, ограничивающего временную сложность. Однако для приближенных алгоритмов одного показателя временной сложности совершенно недостаточно, чтобы характеризовать алгоритм. Необходимо еще оценить алгоритм с точки зрения качества приближенного решения, то есть с точки зрения величины отклонения найденного приближенного решения от оптимального.

Рассмотрим приближенный алгоритм A решения экстремальной задачи Π с множеством исходных данных P . Пусть $f(x)$ – целевая функция задачи. Обозначим через x^0 приближенное решение конкретной задачи Π с исходными данными $p \in P$, полученное с помощью алгоритма A , а через x^* – оптимальное решение этой конкретной задачи.

Функцию $\Delta_A(n)$ натурального аргумента такую, что для всякого $p \in P$ имеет место неравенство

$$f(x^0) \leq \Delta_A(|p|) f(x^*)$$

назовем *оценкой точности* приближенного алгоритма A . Если функция $\Delta(|p|)$ есть константа, то оценку точности называют *гарантированной*. Оценка точности $\Delta(|p|)$ зависит

только от длины исходных данных конкретной задачи, но не от самих этих исходных данных. Эта оценка известна еще до применения алгоритма к конкретной задаче, поэтому такую оценку точности называют *априорной*, а сам алгоритм A – *приближенным алгоритмом с априорной оценкой точности*.

Иногда удобнее характеризовать качество приближенного алгоритма, оценивая не отношение $f(x^0) / f(x^*)$, а измеряя относительную погрешность приближенного решения. Функцию $\rho_A(n)$ такую, что для всякого $p \in P$ имеет место неравенство

$$\frac{f(x^0) - f(x^*)}{f(x^*)} \leq \rho_A(p)$$

назовем *оценкой погрешности* приближенного алгоритма A .

Понятно, что оценки точности и погрешности алгоритма A связаны равенством

$$\Delta_A(n) = 1 + \rho_A(n).$$

Поэтому, если алгоритм A имеет оценку точности, то имеет и оценку погрешности и наоборот.

К сожалению, не для всякого приближенного алгоритма удастся установить априорную оценку точности. Вместе с тем многие приближенные алгоритмы для всякой конкретной задачи P с исходными данными $p \in P$ одновременно с поиском приближенного решения x^0 вычисляют величину $\Delta_A(p)$ такую, что

$$f(x^0) \leq \Delta_A(p)f(x^*).$$

Оценка точности $\Delta_A(p)$ не известна до применения алгоритма A к конкретной задаче с исходными данными p , поэтому такую оценку называют *апостериорной*, а сам алгоритм A – *приближенным алгоритмом с апостериорной оценкой точности* или *эвристическим алгоритмом*. Поскольку апостериорная оценка точности $\Delta_A(p)$ связана с конкретной задачей и конкретным приближенным решением этой задачи, то величину $\Delta_A(p)$ будем называть еще *оценкой точности данного приближенного решения*.

Оценка точности приближенного решения, полученного с помощью эвристического алгоритма, как следует из сказанного выше, не известна до применения алгоритма к данной конкретной задаче. Поэтому о качестве решения, даваемого эвристическим алгоритмом, заранее, еще до применения алгоритма, ничего сказать нельзя. Некоторыми априорными ориентирами могут служить лишь накопленные результаты численных экспериментов с алгоритмом на различных классах конкретных задач исследуемой экстремальной задачи.

В заключение обратимся к вопросу о форме описания алгоритмов. Поскольку алгоритм понимается как программа, для описания алгоритмов часто используются некоторые псевдоязыки, напоминающие хорошо известные языки программирования, такие как ПАСКАЛЬ или АЛГОЛ. Мы не будем привлекать для описания алгоритмов какие-либо полуформальные языки, а будем описывать действия, производимые алгоритмом, обычными словами, считая, что такое описание более понятно, а формальности могут лишь заслонить существо дела.

Далее нам придется рассматривать различные по своему устройству алгоритмы, однако при их описании будем придерживаться одной схемы. Алгоритмы будем пред-

ставлять состоящими из конечной последовательности однотипных *шагов (итераций)*, на каждом из которых производится пересчет значений набора некоторых данных, называемого еще *записью*. Вычисленные на данном шаге значения элементов записи являются исходными значениями для следующего шага и т. д. Чаще всего рассматриваемые далее алгоритмы будут естественным образом разбиваться на два и более последовательных подалгоритмов, для описания каждого из которых будет использоваться указанная схема. Такие подалгоритмы называются еще *этапами* алгоритма.

2 Формулировки неограниченной задачи размещения средств обслуживания

Математически неограниченная задача размещения средств обслуживания, начиная с работы [212], чаще всего формулируется в виде задачи целочисленного линейного программирования. Она может быть сформулирована также в виде задачи смешенного линейного программирования или целочисленного линейного программирования на максимум или в комбинаторном варианте, как задача минимизации функции, заданной на подмножествах конечного множества. Такие формы записи задачи в ряде случаев оказываются более удобными для исследования. Отмеченные формулировки неограниченной задачи размещения, а также ссылки на работы, содержащие такие постановки, можно найти в известном обзоре [188], а также в более поздней монографии [250].

Упомянутые выше эквивалентные формулировки неограниченной задачи размещения имеют одинаковые исходные данные и отличаются друг от друга используемыми для записи задачи переменными. По этой причине данные задачи нельзя в полной мере называть различными математическими задачами, а можно скорее рассматривать как различные трансформации одной и той же задачи. Более интересным было бы получение эквивалентных формулировок неограниченной задачи размещения в виде других известных дискретных задач. Важную роль при установлении таких связей будет играть понятие характеристической матрицы для матрицы затрат на обслуживание потребителей. Характеристические матрицы впервые были рассмотрены в монографии [17], а в последующих работах [6, 18, 19] использованы при исследовании неограниченной задачи размещения. На основе характеристической матрицы строится так называемая каноническая форма матрицы затрат на обслуживание. Использование такой матрицы вместо исходной не приводит к изменению оптимального решения. Вместе с тем эта матрица имеет более простое строение, что упрощает исследование задачи, например, на предмет наличия свойств, полезных при построении полиномиальных алгоритмов решения задачи.

Возможны некоторые усложнения неограниченной задачи размещения, не приводящие по существу к новым математическим задачам. Одно из них – нелинейная задача размещения, в которой затраты на производство продукции выражаются нелинейными функциями от объема выпускаемой продукции. Если такие функции являются

кусочно-линейными и выпуклыми вверх, что хорошо согласуется с реальными свойствами затрат на производство, то нелинейная задача сводится к линейной.

Кроме хорошо известной многократно упоминавшейся выше содержательной интерпретации неограниченной задачи размещения, как задачи выбора мест размещения предприятий, позволяющего наилучшим образом удовлетворить заданный спрос потребителей, возможны и другие, не менее интересные с практической точки зрения, содержательные интерпретации. Имеется в виду прежде всего так называемая задача выбора оптимального состава системы технических средств. Такая задача возникает в ситуации, когда, с одной стороны, имеется некоторое множество работ (задач), которые необходимо выполнить, а с другой стороны, имеется множество разновидностей технических средств (машин, механизмов, приборов и т.п.), которые в принципе могут выполнять указанные работы. Требуется выяснить, какое количество средств каждой разновидности должно быть использовано для выполнения работ, чтобы суммарные затраты на создание такой системы технических средств и затраты на выполнение работ были бы наименьшими. Эта содержательная задача менее известна, чем классическая содержательная постановка неограниченной задачи размещения, однако прикладное значение такой формулировки задачи не менее важно. Она отражает многочисленные практические ситуации выбора типажа и состава оборудования, планирование развития парка машин, решения вопроса целесообразности разработки и производства изделия нового образца и т.д. С использованием указанной интерпретации задачи размещения средств обслуживания в Институте математики им С.Л. Соболева СО РАН выполнен целый ряд прикладных исследований. Подробное изложение результатов в этой области содержится в монографии [17]. Здесь же рассматривается частный случай этой задачи – так называемая задача выбора оптимального параметрического ряда изделий. Такая задача впервые была предложена в работе [41], которая фактически положила начало исследований в области математических моделей выбора оптимального состава систем. Данная задача оказалась очень полезной при решении некоторых вопросов в стандартизации, в частности, при обосновании параметрических (типоразмерных) рядов изделий.

Прикладным исследованиям в области построения параметрических рядов посвящены работы [15, 28].

2.1 Эквивалентные формулировки неограниченной задачи размещения

Как уже отмечалось выше неограниченная задача размещения производства или, другими словами, неограниченная задача размещения средств обслуживания содержательно чаще всего формулируется как задача выбора мест размещения предприятий, производящих однородный продукт для удовлетворения спроса в этом продукте заданного множества потребителей. Выбор мест размещения предприятий производится из множества возможных мест открытия предприятий и осуществляется таким образом, чтобы суммарные затраты, включающие фиксированные затраты на открытие предпри-

ятий и затраты на удовлетворение спроса потребителей были наименьшими. При этом предполагается, что каждый потребитель для удовлетворения своего спроса прикрепляется к открытому предприятию, у которого затраты на удовлетворение спроса данного потребителя наименьшие. Считается также, что возможности каждого открытого предприятия по объему удовлетворяемого спроса потребителей не ограничены и каждое открытое предприятие может, вообще говоря, удовлетворить спрос всех потребителей.

Для формальной записи задачи введем следующие обозначения:

$J = \{1, \dots, n\}$ – множество потребителей с заданными потребностями;

$I = \{1, \dots, m\}$ – множество предприятий (возможных мест, в которых могут быть открыты предприятия);

f_i – величина затрат на открытие предприятия $i \in I$.

c_{ij} – величина затрат на удовлетворение спроса потребителя $j \in J$ предприятием $i \in I$;

Введем также переменные $z_i \in \{0,1\}, i \in I, x_{ij} \in \{0,1\}, i \in I, j \in J$, определяемые следующим образом:

z_i – переменная, показывающая открывается или нет предприятие i ; $z_i=1$, если открывается и $z_i=0$, если нет.

x_{ij} – переменная, показывающая прикрепляется ли потребитель j для удовлетворения своего спроса к предприятию i ; $x_{ij} = 1$, если прикрепляется, $x_{ij} = 0$, если нет.

С использованием введенных обозначений и переменных неограниченная задача размещения предприятий (средств обслуживания) записывается в виде задачи линейного целочисленного программирования следующим образом:

$$\min_{(z_i), (x_{ij})} \left\{ \sum_{i \in I} f_i z_i + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \right\}; \quad (1.2.2)$$

$$\sum_{i \in I} x_{ij} = 1, j \in J; \quad (1.2.2)$$

$$z_i \geq x_{ij}, i \in I, j \in J; \quad (1.2.3)$$

$$z_i, x_{ij} \geq 0, i \in I, j \in J; \quad (1.2.4)$$

$$z_i, x_{ij} \in \{0,1\}, i \in I, j \in J; \quad (1.2.5)$$

Эту задачу далее будем обозначать *FL*. Целевая функция (1.2.1) данной задачи выражает величину суммарных затрат на открытие предприятий в возможных местах размещения и на удовлетворение открытыми предприятиями потребностей прикрепленных к каждому из них потребителей. Ограничения (1.2.2) гарантируют, что каждый потребитель будет прикреплен к некоторому предприятию и тем самым спрос каждого потребителя будет удовлетворен. Наконец, неравенство (1.2.3) показывает, что предприятие не может быть использовано для удовлетворения спроса потребителей, если оно не будет открыто.

Исходные данные задачи FL представляют собой пару матриц (F_0, C) , где F_0 – диагональная матрица размера $m \times m$ с диагональными элементами f_i , $i \in I$, а $C = (c_{ij})$ – матрица размера $m \times n$. Поскольку диагональную матрицу можно отождествлять с вектором–столбцом ее диагональных элементов, то через F_0 будем обозначать также вектор столбец (f_i) и считать, что исходными данными задачи FL является пара матриц, где $F_0 = (f_i)$ – вектор–столбец длины m , а $C = (c_{ij})$ – матрица размера $m \times n$. Отметим, что в достаточно общем случае можно считать, что элементы матрицы $C = (c_{ij})$ представляются следующим образом

$$c_{ij} = p_i c_i + h_{ij},$$

где p_j – величина, задающая количество продукта, необходимое для удовлетворения спроса потребителя j ; c_i – величина затрат на производство единицы продукта на предприятии i ; h_{ij} – затраты на транспортировку необходимого количества продукта от предприятия i к потребителю j .

Исходными данными задачи FL , как отмечено выше, является пара матриц (F_0, C) . На знаки элементов этих матриц в общем случае не накладывается никаких ограничений. Но не трудно понять, что все эти элементы можно считать неотрицательными. Действительно, если $f_{i_0} < 0$ для некоторого $i_0 \in I$, то задача FL с исходной парой (F_0, C) сводится к задаче FL с парой (F'_0, C) , где $f'_i = f_i$, если $i \neq i_0$, и $f'_{i_0} = 0$. Заметим далее, что добавление константы ко всем элементам любого столбца матрицы C не изменяет оптимального решения задачи FL . Поэтому, если $\min_{i \in I} c_{ij_0} = c_{j_0} < 0$ для некоторого $j_0 \in J$, то задача FL с парой (F_0, C) сводится к задаче FL с парой (F_0, C') , где $c'_{ij_0} = c_{ij_0} - c_{j_0} \geq 0$, $i \in I$.

Заметим также, что кроме предположения о неотрицательности элементов пары матриц (F_0, C) , не умаляя общности, можно предполагать, что для исходных данных (F_0, C) задачи FL выполняется неравенство

$$\min_{i \in I} \{f_i + \sum_{j \in J} c_{ij}\} < \sum_{j \in J} \max_{i \in I} c_{ij}. \quad (1.2.6)$$

Это неравенство будем рассматривать как условие существования нетривиального решения задачи FL . Действительно, если это условие не выполняется, то оптимальным решением задачи FL будет решение $((z_i), (x_{ij}))$ такое, что $z_i = 0$, $i \neq i_0$, и $z_{i_0} = 1$, где

$$i_0 = \arg \min_{i \in I} \{f_i + \sum_{j \in J} c_{ij}\}.$$

Далее, если не оговорено противное, мы всегда будем предполагать, что рассматриваемая пара матриц (F_0, C) удовлетворяет приведенному выше условию.

Важным частным случаем задачи FL является задача размещения средств обслуживания на сети. Рассмотрим ориентированный граф без петель $G = (V, E)$ с множеством вершин V и множеством дуг E . Пусть каждой дуге $e \in E$ приписан вес d_e , называемый длиной дуги e , каждой вершине $i \in V$ приписаны величины f_i и p_i , обозначающие соответственно затраты на открытие предприятия в вершине i и спрос потребите-

ля, размещенного в вершине i , и пусть $|V|=m$. Рассмотрим пару матриц (F_0, C) , где F_0 – диагональная матрица размера $m \times m$ с диагональными элементами f_i , $i \in I$, а $C=(c_{ij})$ – квадратная матрица размера $m \times m$, элементы которой определяются следующим образом

$$c_{ij} = p_j s_{ij},$$

где s_{ij} – наименьшая длина пути в графе G из вершины i в вершину j . Задачу FL с парой (F_0, C) , определенной указанным выше образом по взвешенному графу G будем называть *задачей размещения производства (средств обслуживания) на сети* и обозначать далее NFL .

Неограниченная задача размещения производства кроме формулировки в виде задачи целочисленного линейного программирования (1.2.1)–(1.2.5) допускает и другие эквивалентные формы записи. Прежде всего следует отметить, что переменные x_{ij} не обязательно считать целочисленными. Эти переменные можно рассматривать как величины, определяющие долю спроса потребителя j удовлетворяемую предприятием i , и заменить в задаче FL условие (1.2.5) на условие

$$z_i \in \{0,1\}, i \in I; \quad (1.2.5')$$

Таким образом, получаем задачу смешанного линейного программирования (1.2.1)–(1.2.4), (1.2. 5'). Эта задача и задача FL , очевидно, эквивалентны. Действительно, если $((z_i), (x_{ij}))$ оптимальное решение задачи (1.2.1)–(1.2.4), (1.2. 5') такое, что $x_{i_1 j_0} > 0$, $x_{i_2 j_0} > 0$ для некоторого $j_0 \in J$, то в силу оптимальности этого решения имеем $c_{i_1 j_0} = c_{i_2 j_0}$. Следовательно, можно перейти к рассмотрению оптимального решения с меньшим числом ненулевых компонент и, в конечном итоге, построить оптимальное целочисленное решение задачи (1.2.1)–(1.2.4), (1.2. 5').

Эквивалентная форма записи задачи размещения производства получается и в случае замены в задаче FL условий (1.2.3) на следующие неравенства

$$\sum_{j \in J} x_{ij} \leq n z_i, i \in I. \quad (1.2.3')$$

Такую задачу называют *задачей FL в слабой форме* в отличие от задачи FL , которую будем называть еще *задачей FL в сильной форме*.

Задача FL и задача FL в слабой форме эквивалентны. Действительно, множества допустимых решений у этих задач совпадают. Если $((z_i), (x_{ij}))$ – допустимое решение задачи FL , то ограничение (1.2.3') для этого решения, очевидно, выполняется. С другой стороны, если для решения $((z_i), (x_{ij}))$ выполняются неравенства (1.2.3'), то справедливы и неравенства (1.2.3). В самом деле, если для некоторых $i \in I$ и $j \in J$ имеем $x_{ij} > z_i$, то тогда $x_{ij} = 1$, $z_i = 0$, что противоречит справедливости неравенств (1.2.3').

Задача FL может быть эквивалентным образом переформулирована в виде задачи целочисленного линейного программирования на максимум. Такая формулировка будет соответствовать несколько иной содержательной постановке задачи размещения производства. Эта постановка предполагает, что предприятие, удовлетворяющее спрос

потребителя, получает некоторый доход. Требуется выбрать места размещения предприятий и прикрепить потребителей к открытым предприятиям таким образом, чтобы суммарный доход предприятий за вычетом затрат на открытие был максимальным. Пусть

d_{ij} – величина дохода, получаемая предприятием $i \in I$ при удовлетворении спроса потребителя $j \in J$.

Неограниченная задача размещения производства (средств обслуживания) на максимум записывается следующим образом:

$$\begin{aligned} \max_{(z_i), (x_{ij})} & \left\{ \sum_{i \in I} f_i z_i + \sum_{i \in I} \sum_{j \in J} d_{ij} x_{ij} \right\}; \\ & \sum_{i \in I} x_{ij} = 1, j \in J; \\ & z_i \geq x_{ij}, i \in I, j \in J; \\ & z_i, x_{ij} \in \{0, 1\}, i \in I, j \in J. \end{aligned}$$

Эту задачу будем обозначать $MAXFL$. Исходные данные этой задачи представляют собой пару матриц (F_0, D) , где $D=(d_{ij})$ – матрица размера $m \times n$, называемая *матрицей доходов*.

Несложно понять, что задача FL и $MAXFL$ эквивалентны. Действительно, рассмотрим задачу FL с парой (F_0, C) и задачу $MAXFL$ с парой (F_0, D) такие, что $C=-D$. Понятно, что множества допустимых решений обеих задач одинаковые. Кроме того, имеет место равенство:

$$\min_{(z_i), (x_{ij})} \left\{ \sum_{i \in I} f_i z_i + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \right\} = - \max_{(z_i), (x_{ij})} \left\{ - \sum_{i \in I} f_i z_i + \sum_{i \in I} \sum_{j \in J} d_{ij} x_{ij} \right\},$$

из которого следует, что если $((z_i), (x_{ij}))$ – оптимальное решение одной из этих задач, то оно будет также оптимальным решением и для другой задачи.

Рассмотрим также так называемый комбинаторный вариант формулировки неограниченной задачи размещения производства. Определим функцию $f_0(X)$, заданную на подмножествах $X, X \subset I$, следующим образом:

$$f_0(X) = \sum_{i \in X} f_i + \sum_{j \in J} \min_{i \in X} c_{ij}.$$

При этом считаем, что если $X = \emptyset$, то

$$\min_{i \in X} c_{ij} = \max_{i \in I} c_{ij}, j \in J.$$

Задачей выбора множества строк пары матриц (F_0, C) назовем задачу отыскания минимума функции $f_0(X)$, то есть задачу

$$\min_X \{f_0(X) \mid X \subset I\}.$$

Эту задачу далее будем обозначать $MINF_0$. Отметим, что если пара матриц (F_0, C) удовлетворяет условию (1.2.6), то оптимальным решением задачи является непустое множество.

Покажем, что задачи FL и $MINF_0$ эквивалентны. Для множества $X \subset I$, $X \neq \emptyset$ положим

$$i(j) = \arg \min_{x \in X} c_{ij}, j \in J.$$

Решения $((z_i), (x_{ij}))$ и X задач FL и $MINF_0$ назовем соответствующими, если

$$z_i = \begin{cases} 1, & \text{если } i \in I, \\ 0, & \text{иначе;} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{если } i = i(j), \\ 0, & \text{иначе.} \end{cases}$$

Заметим, что на соответствующих решениях значения целевых функций рассматриваемых задач равны. Заметим также, что если X – оптимальное решение задачи $MINF_0$, то $X \neq \emptyset$ и, следовательно, по нему очевидным образом строится соответствующее решение задачи FL . С другой стороны, рассмотрим оптимальное решение $((z_i), (x_{ij}))$ задачи FL . Для всякого $j \in J$ через $i(j)$ обозначим элемент множества I , для которого $x_{i(j)j} = 1$. Поскольку рассматриваемое решение оптимальное, то выполняются равенства

$$c_{i(j)j} = \min_{i|z_i=1} c_{ij}, j \in J.$$

Отсюда получаем, что если $((z_i), (x_{ij}))$ – оптимальное решение задачи FL , то множество $X = \{i | z_i = 1\}$ и данное оптимальное решение будут соответствующими. Таки образом, в силу признака оптимальности (лемма 1.1) получаем, что если одно из соответствующих решений $((z_i), (x_{ij}))$ и X является оптимальным, то оптимальным будет и другое решение.

Отметим, что задачу $MINF_0$ можно рассматривать как результат исключения из задачи FL переменных x_{ij} . Это оказалось возможным, поскольку возможно определить оптимальные значения переменных x_{ij} через оптимальные значения переменных z_i . Аналогичным образом можно поступить и с переменными z_i . Заметим, то если $((z_i), (x_{ij}))$ – оптимальное решение задачи FL , то

$$z_i = \max_{j \in J} x_{ij}, i \in I.$$

Отсюда получаем, что задача FL эквивалентна следующей задаче:

$$\min_{(x_{ij})} \left\{ \sum_{i \in I} f_i \cdot \max_{j \in J} x_{ij} + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \right\};$$

$$\sum_{i \in I} x_{ij} = 1, j \in J;$$

$$x_{ij} \in \{0, 1\}, i \in I, j \in J.$$

Исходными данными задачи LF и эквивалентной ей задачи $MINF_0$ является, как уже отмечено ранее, пара матриц (F_0, C) с неотрицательными элементами, удовлетво-

ряющими условию (1.2.6). Заметим в дополнение к этому, что элементы вектор-столбца F_0 , не умаляя общности, можно считать положительными величинами. Для этого покажем, что если $f_{i_0} = 0$ для некоторого $i_0 \in I$, то задача $MINF_0$ с парой (F_0, C) сводится к задаче $MINF_0$, в которой $I' = I \setminus \{i_0\}$, а элементы матриц F'_0 и C' определяются следующим образом:

$$\begin{aligned} f'_i &= f_i, i \in I'; \\ c'_{ij} &= \min\{c_{ij}; c_{i_0j}\}, i \in I', j \in J. \end{aligned}$$

Пусть X – оптимальное решение задачи $MINF_0$ с парой (F_0, C) . Если $i_0 \notin X$, то для любого $j \in J$ имеем

$$\min_{i \in X} c_{ij} \leq c_{i_0j}.$$

Поэтому для решения $X' = X$ задачи $MINF_0$ с парой (F'_0, C') справедливы равенства

$$f'_0(X') = \sum_{i \in X'} f'_i + \sum_{j \in J} \min_{i \in X'} c'_{ij} = \sum_{i \in X} f_i + \sum_{j \in J} \min_{i \in X} c_{ij} = f_0(X).$$

Если же $i_0 \in X$, то для решения $X' = X \setminus \{i_0\}$ также справедливо $f'_0(X) = f_0(X)$.

Пусть теперь X' – оптимальное решение задачи $MINF_0$ с парой (F'_0, C') . Рассмотрим решение $X = X' \cup \{i_0\}$ задачи $MINF_0$ с парой (F_0, C) , для которого, очевидно, справедливо равенство $f_0(X) = f'_0(X')$. Отсюда в силу леммы 1.1 получаем, что если X' – оптимальное решение задачи $MINF_0$ с парой (F'_0, C') , то $X = X' \cup \{i_0\}$ – оптимальное решение исходной задачи $MINF_0$. Это завершает доказательство предположения, позволяющего считать диагональные элементы матрицы F_0 положительными величинами.

Рассмотренные выше эквивалентные задачи имеют одинаковые исходные данные в виде пары матриц (F_0, C) и отличаются друг от друга, в основном, используемыми для формулировки задачи переменными. В этом смысле их можно не считать различными задачами, а говорить о различных способах формальной записи одной и той же задачи. Более интересным было бы получение эквивалентных формулировок неограниченной задачи размещения производства в виде других известных экстремальных задач, таких как, например, задача о покрытии множествами. Для выявления связей задачи FL с другими дискретными экстремальными задачами нам потребуется понятие характеристической матрицы для матрицы затрат на обслуживание.

2.2 Характеристическая матрица

Рассмотри пару матриц (F_0, C) , где $F_0 = (f_i)$ – вектор-столбец длины m , а $C = (c_{ij})$ – матрица размера $m \times n$. Для всякого $j \in J$ рассмотрим перестановку $\{i_1^j, i_2^j, \dots, i_m^j\}$ множества I такую, что

$$c_{i_1^j j} \leq c_{i_2^j j} \leq \dots \leq c_{i_m^j j};$$

и определим неотрицательные коэффициенты

$$\Delta c_0^j = c_{i_1^j j},$$

$$\Delta c_k^j = c_{i_{k+1}^j j} - c_{i_k^j j}, \quad k = 1, \dots, m-1.$$

Указанную перестановку назовем *порожденной j -м столбцом* матрицы C , а введенные коэффициенты – *коэффициентами роста элементов j -го столбца*.

Пусть $\beta \subset I \mid |\beta| = k, 1 \leq k \leq m-1$, – такое подмножество, что множество

$$J(\beta) = \{j \in J \mid \{i_1^j, \dots, i_k^j\} = \beta, \Delta c_k^j > 0\}$$

не пусто. Множество β назовем *характеристическим множеством* длины k матрицы C , а величину $b(\beta) = \sum_{j \in J(\beta)} \Delta c_k^j$ – *коэффициентом* или *весом* характеристического множества β .

Пусть β_1, \dots, β_S – все характеристические множества матрицы C . Булеву матрицу $H = (h_{is})$ размера $m \times S$, где

$$h_{is} = \begin{cases} 0, & \text{если } i \in \beta_s, \\ 1, & \text{иначе} \end{cases}$$

назовем *характеристической* для матрицы C , а вес $b(\beta)$ характеристического множества β_s назовем *весом s -го столбца* характеристической матрицы H . При этом для определенности будем считать, что столбцы матрицы H лексикографически упорядочены по убыванию.

Отметим, что хотя перестановки, задаваемые столбцами матрицы C , определяются, вообще говоря, неоднозначно, тем не менее, совокупность характеристических множеств матрицы C и значения их весов определяются по матрице C однозначно. Поэтому характеристическая матрица и веса столбцов этой матрицы определяются по матрице C однозначно.

Если $H = (h_{is})$ – характеристическая матрица для матрицы C , а (b_s) – вектор весов столбцов характеристической матрицы, то матрицу $\tilde{C} = (b_s h_{is})$ назовем *канонической формой* матрицы C . Каноническая форма \tilde{C} имеет более простое строение по сравнению с матрицей C , поскольку ненулевые элементы каждого столбца матрицы \tilde{C} одинаковые. Далее мы увидим, что оптимальные решения задачи FL с парой (F_0, C) и задачи FL с парой (F_0, \tilde{C}) совпадают. Поэтому матрицу \tilde{C} можно рассматривать как носитель информации о строении матрицы C . Кроме того, более простое строение матрицы \tilde{C} открывает дополнительные возможности при исследовании матрицы C на предмет наличия у нее полезных свойств, облегчающих поиск оптимального решения задачи FL .

В качестве примера построим характеристическую матрицу H и каноническую форму \tilde{C} для следующей матрицы C

$$C = \begin{bmatrix} 6 & 3 & 3 \\ 4 & 2 & 4 \\ 3 & 3 & 1 \\ 1 & 6 & 2 \end{bmatrix}$$

Характеристическими множествами 1-го столбца матрицы C являются множества $\{4\}$, $\{3, 4\}$, $\{2, 3, 4\}$ с весами равными соответственно 2, 1, 2.

Характеристическими множествами 2-го столбца будут множества $\{2\}$, $\{1, 2, 3\}$ с весами 1, 3.

Характеристическими множествами 3-го столбца будут множества $\{3\}$, $\{3, 4\}$, $\{1, 3, 4\}$, веса которых равняются соответственно 1, 1, 1.

Все перечисленные выше множества являются характеристическими множествами матрицы, таких множеств всего 7:

$$\beta_1 = \{2\}, \beta_2 = \{3\}, \beta_3 = \{4\}, \beta_4 = \{3, 4\}, \beta_5 = \{1, 2, 3\}, \beta_6 = \{1, 3, 4\}, \beta_7 = \{2, 3, 4\}.$$

Значения весов этих множеств следующие

$$b_1=1, b_2=1, b_3=2, b_4=2, b_5=3, b_6=1, b_7=2.$$

Характеристическая матрица H и каноническая форма \tilde{C} записываются следующим образом:

$$H = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}, \quad \tilde{C} = \begin{pmatrix} 0 & 0 & 2 & 1 & 2 & 1 & 2 \\ 0 & 1 & 0 & 0 & 2 & 1 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 2 \\ 3 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

2.3 Неограниченная нелинейная задача размещения производства

С учетом того, что величина затрат c_{ij} на удовлетворение спроса потребителя j предприятием i представляется в виде

$$c_{ij} = p_j c_j + h_{ij},$$

приведем еще одну запись неограниченной задачи размещения производства, эквивалентную приведенным выше формулировкам. Для этого введем переменные $v_i \geq 0$, $i \in I$, где v_i – величина, равная объему продукции, производимой на предприятии i и потребляемой всеми потребителями, прикрепленными к данному предприятию.

Рассмотрим следующую задачу:

$$\min_{(v_i), (x_{ij})} \left\{ \sum_{i \in I} g_i(v_i) + \sum_{i \in I} \sum_{j \in J} h_{ij} x_{ij} \right\}; \quad (1.2.7)$$

$$v_i = \sum_{j \in J} p_j x_{ij}, \quad i \in I; \quad (1.2.8)$$

$$\sum_{i \in I} x_{ij} = 1, \quad j \in J; \quad (1.2.9)$$

$$v_i \geq 0, \quad i \in I; \quad (1.2.10)$$

$$x_{ij} \in \{0,1\}, \quad i \in I, j \in J; \quad (1.2.11)$$

В этой задаче функция $g_i(v)$ выражает величину затрат на производство v единиц продукции на предприятии i и имеет вид

$$g_i(v) = \begin{cases} 0, & \text{если } v = 0, \\ f_i + c_i v, & \text{если } v > 0. \end{cases}$$

такие функции называют *полулинейными*.

Приведенная задача (1.2.7) – (1.2.11) и задача FL эквивалентны, поскольку по оптимальному решению каждой из этих задач легко строится допустимое решение другой с тем же значением целевой функции. Поэтому задачу (1.2.7) – (1.2.11) можно считать еще одной формой записи неограниченной задачи размещения производства.

Из вида функций $g_i(v)$, $i \in I$, присутствующих в этой задаче, можно заключить, что неограниченная задача размещения описывает ситуацию, когда затраты на производство единицы продукции на каждом предприятии являются постоянными величинами и не зависят от объема выпуска продукции. Вместе с тем на практике возможна и более общая ситуация, когда затраты на производство единицы продукции измеряются в зависимости от объема выпуска.

Для учета такой возможности рассмотрим в качестве функции затрат на производство функцию $g_i(v)$ вида

$$g_i(v) = \min_{r=1, \dots, R} g_{ir}(v),$$

где $g_{ir}(v)$, $r=1, \dots, R$, полулинейные функции

$$g_{ir}(v) = \begin{cases} 0, & \text{если } v = 0, \\ f_{ir} + c_{ir} v, & \text{если } v > 0, \end{cases}$$

такие, что

$$\begin{aligned} 0 &\leq f_{i1} < f_{i2} < \dots < f_{iR}, \\ c_{i1} &> c_{i2} > \dots > c_{iR} \geq 0 \end{aligned}$$

Отметим, что функция $g_i(v)$ указанного вида описывает ситуацию, когда на предприятии i производство может быть организовано по одной из возможных технологий, выбираемых из множества $\{1, \dots, R\}$. При этом технологию r характеризует величина начальных затрат f_{ir} на открытие предприятия для производства продукции по технологии r и величина затрат c_{ir} на производство единицы продукции, выпускаемой по технологии r . В зависимости от того, какой объем выпуска продукции будет на предприятии i , зависит выбор используемых технологий.

Функцию затрат на производство $g_i(v)$ указанного вида назовем *кусочно-линейной*, поскольку такая функция может быть записана следующим образом:

$$g_i(v) = \begin{cases} 0, & \text{если } v = 0, \\ \dots\dots\dots, \\ f_{ir} + c_{ir}v, & \text{если } V_{ir-1} < v \leq V_{ir}, \\ \dots\dots\dots, \\ f_{iR} + c_{iR}v, & \text{если } V_{iR-1} < v \leq V_{iR}. \end{cases}$$

Здесь точки

$$0 = V_{i0} < V_{i1} < \dots < V_{iR} = \sum_{j \in J} p_j,$$

называемые *узлами* функции $g_i(v)$ удовлетворяют равенствам

$$f_{ir} + c_{ir}v_{ir} = f_{ir+1} + c_{ir+1}v_{ir}, \quad r = 1, \dots, R-1.$$

Отметим. Что введенные кусочно-линейные функции являются возрастающими выпуклыми вверх функциями. Это соответствует реальным свойствам затрат на производство продукции в зависимости от объема выпуска, которые возрастают с увеличением объема выпуска, но при этом затраты на единицу продукции уменьшаются.

Задачу (1.2.7) – (1.2.11) с кусочно-линейными функциями затрат на производство $g_i(v)$, $i \in I$, будем называть *нелинейной* задачей. Хотя кусочно-линейные функции являются более общими и более сложно устроенными по сравнению с полулинейными функциями, тем не менее, нелинейная задача (1.2.7) – (1.2.11) не становится, как показывает нижеследующая лемма, существенно более сложной.

Лемма 1.2 Нелинейная задача (1.2.7) – (1.2.11) сводится к задаче *FL*.

Доказательство. Для каждой из функций $g_i(v)$, $i \in I$, рассмотрим полулинейные функции $g_{ir}(v)$, $r = 1, \dots, R$, такие что $g_i(v) = \min_r g_{ir}(v)$. Нелинейной задаче (1.2.7) – (1.2.11) поставим в соответствие следующую задачу вида (1.2.7) – (1.2.11) с полулинейными функциями затрат на производство $g_{ir}(v)$, $i \in I$, $r = 1, \dots, R$:

$$\min_{(v_{ir})(x_{irj})} \left\{ \sum_{i \in I} \sum_{r=1}^R g_{ir}(v_{ir}) + \sum_{i \in I} \sum_{r=1}^R \sum_{j \in J} h_{ij} x_{irj} \right\}, \quad (1.2.12)$$

$$v_{ir} = \sum_{j \in J} p_j x_{irj}, \quad i \in I, r = 1, \dots, R; \quad (1.2.13)$$

$$\sum_{i \in I} \sum_{r=1}^R x_{irj} = 1, \quad j \in J; \quad (1.2.14)$$

$$v_{ir} \geq 0, \quad i \in I, r = 1, \dots, R; \quad (1.2.15)$$

$$x_{irj} \in \{0,1\}, \quad i \in I, r = 1, \dots, R, j \in J. \quad (1.2.16)$$

Пусть $((v_{ir}^*), (x_{irj}^*))$ – оптимальное решение этой задачи. Покажем, что решение $((v_i), (x_{ij}))$ где

$$v_i = \sum_{r=1}^R v_{ir}^*, \quad i \in I;$$

$$x_{ij} = \sum_{r=1}^R x_{irj}^*, \quad i \in I, j \in J,$$

будет оптимальным решением исходной нелинейной задачи (1.2.7) – (1.2.11).

Заметим, прежде всего, что данное решение является допустимым поскольку выполняются равенства:

$$\begin{aligned} v_i &= \sum_{r=1}^R v_{ir}^* = \sum_{j \in J} p_j \sum_{r=1}^R x_{irj}^* = \sum_{j \in J} p_j x_{ij}, \\ \sum_{i \in I} x_{ij} &= \sum_{i \in I} \sum_{r=1}^R x_{irj}^* = 1. \end{aligned}$$

Для доказательства оптимальности решения воспользуемся леммой 1.1.

Покажем, что при любом $i \in I$ среди величин $v_{ir}^*, r = 1, \dots, R$ не может быть более одной ненулевой. Действительно, если $v_{ip}^* > 0$ и $v_{iq}^* > 0$ при $p < q$, то с учетом неравенства

$$g_{ip}(v_{ip}^*) + g_{iq}(v_{iq}^*) > f_{iq} + c_{iq}(v_{ip}^* + v_{iq}^*) = g_{iq}(v_{ip}^* + v_{iq}^*)$$

получаем допустимое решение $((v_{ir}), (x_{irj}))$, отличающееся от оптимального тем, что

$$\begin{aligned} v_{ip} &= 0, \\ v_{iq} &= v_{ip}^* + v_{iq}^*, \\ x_{ipj} &= 0, \quad j \in J, \\ x_{iqj} &= x_{ipj}^* + x_{iqj}^*, \quad j \in J. \end{aligned}$$

Значение целевой функции на этом решении меньше, чем оптимальное, что невозможно и доказывает требуемое.

Покажем далее, что при любом $i \in I$, если $v_{ip}^* > 0$, то $g_{ip}(v_{ip}^*) = \min_r g_{ir}(v_{ip}^*)$. Действительно, если $g_{iq}(v_{ip}^*) < g_{ip}(v_{ip}^*)$, то получаем допустимое решение $((v_{ir}), (x_{irj}))$, отличающееся от оптимального тем, что

$$\begin{aligned} v_{ip} &= 0, \\ v_{iq} &= v_{ip}^*, \\ x_{ipj} &= 0, \quad j \in J, \\ x_{iqj} &= x_{ipj}^*, \quad j \in J. \end{aligned}$$

Значение целевой функции на этом решении лучше, чем оптимальное значение, что невозможно и доказывает требуемое

С учетом последнего замечания для всякого $i \in I$ такого, что $v_{ip}^* > 0$ для некоторого p , $1 \leq p \leq P$, можно написать

$$\sum_{r=1}^R g_{ir}(v_{ir}^*) = g_{ip}(v_{ip}^*) = \min_r g_{ir}(v_{ip}^*) = g_i(v_{ip}^*) = g_i\left(\sum_{r=1}^R v_{ir}^*\right) = g_i(v_i)$$

Отсюда получаем, что значение целевой функции нелинейной задачи (1.2.7) – (1.2.11) на построенном решении $((v_i), (x_{ij}))$ равняется оптимальному значению целевой функции задачи (1.2.12) – (1.2.16).

С другой стороны, пусть $((v_i^*), (x_{ij}^*))$ – оптимальное решение нелинейной задачи (1.2.7) – (1.2.11). Для всякого $i \in I$ положим

$$r(i) = \arg \min_r g_{ir}(v_i^*).$$

Построим допустимое решение $((v_{ir}), (x_{irj}))$ задачи (1.2.12) – (1.2.16) следующим образом

$$v_{ir} = \begin{cases} 0, & \text{если } r \neq r(i), \\ v_i, & \text{если } r = r(i); \end{cases}$$

$$x_{irj} = \begin{cases} 0, & \text{если } r \neq r(i), \\ x_{ij}, & \text{если } r = r(i). \end{cases}$$

Понятно, что это допустимое решение, на котором значение целевой функции (1.2.12) равняется оптимальному значению целевой функции (1.2.7). Это означает, что критерий оптимальности выполняется и решение $((v_i), (x_{ij}))$, построенное по оптимальному решению $((v_{ir}^*), (x_{irj}^*))$ задачи (1.2.12) – (1.2.16), также является оптимальным, что завершает доказательство леммы.

Таким образом, получаем, что обобщение задачи *FL* путем введения нелинейных функций затрат на производство при естественных предположениях о свойствах этих функций (кусочно-линейные функции) не приводит к новой математической задаче, а лишь увеличивает размерность задачи *FL*. Увеличение размерности происходит за счет расширения множества потенциально открываемых предприятий. Элементами такого множества являются пары (i, r) , $i \in I$, $r = 1, \dots, R$, каждая из которых помимо места размещения предприятия указывает еще и на технологию производства продукции, используемую на этом предприятии.

2.4 Задача выбора оптимального состава системы технических средств

Кроме рассмотренной выше содержательной интерпретации задачи *FL* как задачи размещения средств обслуживания приведем еще одну содержательную постановку задачи *FL*. Эта постановка носит название *задача выбора оптимального состава системы технических средств*. Хотя такая содержательная интерпретация задачи *FL* менее известна, чем «классическая» постановка задачи *FL* в терминах размещения средств обслуживания, тем не менее она открывает не меньшие перспективы для практического использования задачи *FL*.

Под *системой технических средств* понимается совокупность таких средств (машин, механизмов, приборов и т.п.) нескольких разновидностей (типов, образцов), объединенных общностью функционального назначения. Система предназначена для выполнения некоторого заданного множества работ (задач), определенных видов. Эта совокупность работ образует *область применения* системы. *Состав системы*, то есть набор конкретных разновидностей технических средств, взятых в определенных количествах, может быть различным. Например, систему могут образовывать технические средства только одной разновидности или, наоборот, состав системы может быть достаточно разнородным и включать в себя широкий спектр разновидностей технических средств. Задача состоит в том, чтобы выбрать состав системы технических средств, который позволил бы выполнить все работы области применения с наименьшими затратами, включающими затраты на создание системы и затраты на выполнение работ. При этом затраты на создание системы складываются из начальных затрат на средства каждой разновидности (затрат на разработку, организацию производства) и затрат на производство (покупку) в нужных количествах средств данной разновидности. Затраты на выполнение работ могут включать в себя, например, затраты на эксплуатацию технических средств при выполнении ими работ области применения. Понятно, что затраты на создание системы и затраты на выполнение работ находятся в некотором противоречии. Уменьшение первых приводит к сокращению разновидностей технических средств в составе системы, что может привести к возрастанию эксплуатационных затрат поскольку для выполнения некоторых видов работ придется использовать средства, которые могут быть малопригодными для этих целей. С другой стороны, экономия в сфере использования может потребовать неоправданного роста многообразия специализированных технических средств, что приведет к возрастанию затрат на создание системы.

Для формальной записи задачи выбора оптимального в смысле суммарных затрат состава системы введем следующие обозначения.

$J = \{1, \dots, n\}$ – множество работ (видов работ), образующих область применения системы;

$I = \{1, \dots, m\}$ – множество разновидностей технических средств, допустимых для вхождения в состав системы;

f_i – начальные затраты на вхождение в состав системы технических средств разновидности $i \in I$;

c_{ij} – затраты на выполнение техническими средствами разновидности $i \in I$ работы вида $j \in J$.

Введем также переменные $z_i \in \{0, 1\}$, $i \in I$; $x_{ij} \in \{0, 1\}$, $i \in I$, $j \in J$, имеющие следующий содержательный смысл:

z_i – переменная, показывающая входят или нет технические средства разновидности i в состав системы; $z_i = 1$, если входят, и $z_i = 0$, если нет.

x_{ij} – переменная, показывающая используются или нет входящие в состав системы средства разновидности i для выполнения работы j ; $x_{ij} = 1$, если используются, $x_{ij} = 0$, если нет.

Легко видеть, что введенные выше параметры и переменные те же, что и у задачи FL . Поэтому, не выписывая повторно соответствующих соотношений, задачу (1.2.1) – (1.2.5) будем рассматривать как формулировку задачи выбора оптимального состава системы технических средств. В этой задаче целевая функция (1.2.1) выражает величину суммарных затрат на создание и использование системы технических средств. Ограничения (1.2.2) обеспечивают обязательное выполнение всех работ области применения системы, а неравенства (1.2.3) показывают, что если технические средства некоторой разновидности не входят в состав системы, то такие средства не могут быть использованы для выполнения работ области применения.

Отметим, что величины c_{ij} , представленные выше как затраты на выполнение средствами разновидности i работы j , в достаточно общем случае имеют следующий вид:

$$c_{ij} = p_{ij}c_i + h_{ij},$$

где

p_{ij} – число средств разновидности i , необходимых одновременно для выполнения работы j ;

c_i – затраты на производство (покупку) одного средства разновидности i ;

h_{ij} – затраты на эксплуатацию соответствующего количества средств разновидности i , одновременно используемых для выполнения работы j .

Заметим, что в случае задачи размещения средств обслуживания при расшифровке вида величины c_{ij} вместо p_{ij} участвует величина p_j , значение которой зависит только от потребителя j . Необходимо подчеркнуть, что в случае рассматриваемой задачи зависимость величины p_{ij} и от разновидности средства и от вида работы является существенной, поскольку количество средств, привлекаемых для выполнения работы, определяется не только самой работой, но и типом средств ее выполняющих.

2.5 Задача выбора оптимального ряда изделий

Частным случаем рассмотренной выше задачи выбора оптимального состава системы технических средств является задача выбора оптимального параметрического (типоразмерного) ряда изделий. Такие задачи возникают в стандартизации, одной из задач которой является исключение нерационального многообразия видов, марок, типоразмеров и моделей продукции. Эту задачу стандартизация решает в том числе посредством установления так называемых параметрических (типоразмерных) рядов изделий. Под *параметрическим рядом* понимается совокупность образцов изделий, однородных по функциональному назначению и отличающихся значением некоторого параметра (показателя), характеризующего данные изделия. Другими словами, это есть ряд образцов изделий, одинаковых по функциональному назначению, но отличающихся значением некоторой основной характеристики этих изделий.

При построении параметрического ряда возникает вопрос, какие образцы из множества всевозможных образцов образуют ряд и, следовательно, будет использо-

ваться для удовлетворения спроса на изделия других образцов, не входящих в ряд. При этом предполагается, что изделие с большим значением параметра может быть использовано вместо изделия с меньшим значением параметра. Для ответа на этот вопрос в качестве обобщенного критерия выбора может быть использована величина затрат на удовлетворение изделиями с выбранными значениями параметра спроса потребителей на изделия данного назначения в целом. При этом нельзя иметь в виду только затраты на производство, поскольку учет только интересов производителей неизбежно приведет к выбору неоправданно «редкого» ряда образцов изделий. С другой стороны, учет затрат только в сфере потребления приведет к неоправданно «густому» ряду изделий, поскольку потребителям выгодно иметь достаточно широкое разнообразие изделий для сокращения потерь из-за несоответствия значений параметра у требуемого изделия и у предлагаемого. Разрешение этого противоречия лежит на пути выбора ряда, которому соответствуют наименьшие суммарные затраты. Таким образом, приходим к задаче *выбора оптимального параметрического ряда изделий*, то есть такого ряда, при котором заданный спрос на рассматриваемые изделия удовлетворяется изделиями с выбранными значениями параметра с наименьшими суммарными затратами в сферах производства и потребления.

Для формальной записи задачи введем следующие обозначения:

$I = \{1, \dots, m\}$ – множество образцов изделий с всевозможными допустимыми значениями основного параметра, упорядоченных по возрастанию значений этого параметра;

φ_j – величина потребности в изделиях образца j ;

f_i – величина начальных затрат, связанных с организацией производства изделий образца i ;

c_i – затраты на использование потребителями одного изделия образца i .

Рассмотрим также переменные $z_i \in \{0, 1\}$, $i \in I$, и переменные $x_{ij} \in \{0, 1\}$, $i \in I, j \in I$, которым придадим следующий содержательный смысл:

z_i – переменная, показывающая входит или нет образец i в состав ряда; $z_i = 1$, если входит и $z_i = 0$, если не входит.

x_{ij} – переменная, показывающая используется или нет при удовлетворении потребностей изделия образца i вместо изделий образца j ; $x_{ij} = 1$, если используется и $x_{ij} = 0$, если нет.

С использованием введенных обозначений задача выбора оптимального ряда изделий записывается следующим образом:

$$\min_{(z_i), (x_{ij})} \left\{ \sum_{i \in I} c_i z_i + \sum_{i \in I} \sum_{j=1}^i \varphi_j c_i x_{ij} \right\},$$

$$\sum_{i=j}^n x_{ij} = 1, \quad j \in I,$$

$$z_i \geq x_{ij}, \quad i \in I, j \in I,$$

$$z_i, x_{ij} \in \{0,1\}, \quad i \in I, j \in I.$$

Понятно, что это есть задача FL , у которой матрица $C=(c_{ij})$ размера $m \times m$ имеет элементы вида

$$c_{ij} = \begin{cases} \infty, & \text{если } j > i, \\ \varphi_j c_i, & \text{если } j \leq i. \end{cases}$$

Понятно также, что матрица C данной задачи обладает некоторыми важными свойствами, связанными со спецификой задачи выбора оптимального ряда, влияющими на строение оптимального решения задачи FL . В частности, если $((z_i^*), (x_{ij}^*))$ – оптимальное решение задачи, то для всякого $i \in I$ такого, что $z_i^* = 1$, множество $J(i) = \{j \in I \mid x_{ij}^* = 1\}$, называемое областью использования изделий образца i , имеет достаточно простое строение в виде некоторого целочисленного сегмента $\{k, k+1, \dots, i\}$.

3 Задача минимизации полиномов от булевых переменных

Представленные в предыдущем параграфе задачи являют собой различные математические формулировки задачи размещения средств обслуживания. Среди этих эквивалентных формулировок выделим две: Задачу FL и задачу $MINF_0$, первая из которых есть задача целочисленного линейного программирования, а вторая — задача минимизации функции множества. Исходные данные для этих задач задаются парой матриц (F_0, C) , а сами задачи отличаются одна от другой используемыми переменными. При этом по оптимальным значениям переменных одной задачи легко определяются оптимальные значения переменных другой задачи. Поэтому задачи FL и $MINF_0$ следует рассматривать не как разные математические задачи, а как разные формы записи одной и той же задачи.

Представляют интерес эквивалентные формулировки задачи размещения средств обслуживания в виде других известных экстремальных задач. К числу таких задач относится, прежде всего, такая широко известная и хорошо изученная экстремальная задача, как задача о покрытии множества системой подмножеств. Однако прежде чем перейти к исследованию связи между задачей FL и задачей о покрытии множества введем в рассмотрение еще одну важную для дальнейшего экстремальную задачу — задачу минимизации полиномов от булевых переменных. Эта задача будет играть роль связующего звена между задачей FL и задачей о покрытии множествами.

Как уже отмечалось, внимание к задаче минимизации функций от булевых переменных, названных псевдобулевыми, было привлечено работами [166, 167], в которых предложен универсальный способ построения алгоритмов минимизации псевдобулевых функций, названный методом псевдобулева программирования. Ниже приводится общая схема алгоритма, построенного на основе этого метода. Описание алгоритма

следует работе [167] и при этом основное внимание уделяется идейной стороне вопроса, а некоторые детали формальных обоснований опущены.

Представление задачи FL в виде задачи минимизации псевдобулевой функции предложено в [169]. Независимо от этого в [13] дано представление задачи FL в виде задачи минимизации полиномов от булевых переменных с неотрицательными коэффициентами при нелинейных членах. Существенных различий между указанными псевдобулевой функцией и полиномом от булевых переменных не имеется и несложными преобразованиями псевдобулева функция может быть приведена к полиному от булевых переменных. В этом смысле полученные в указанных работах результаты по представлению задачи FL в виде полинома от булевых переменных равносильны. Существенное различие состоит в том, что в [13, 17, 19] показана и обратная сводимость, то есть, показано, что задача минимизации полинома от булевых переменных с неотрицательными коэффициентами сводится к задаче FL . Это означает, что данная задача есть еще одна математическая формулировка задачи размещения средств обслуживания, эквивалентная рассмотренным ранее задачам FL и $MINF_0$.

Использование задачи минимизации полинома от булевых переменных позволяет ввести понятие эквивалентных матриц [17]. Эквивалентные матрицы обладают тем свойством, что задачи FL с такими матрицами в качестве матриц затрат на обслуживание имеют одинаковые оптимальные решения. В классе эквивалентных матриц в качестве представителя этого класса можно выделить так называемую матрицу в канонической форме. Такая матрица обладает замечательным свойством, состоящим в том, что ненулевые элементы каждого столбца этой матрицы одинаковые. Понятие эквивалентных матриц полезно в плане построения эффективно разрешимых частных случаев задачи FL . В силу отмеченного свойства эквивалентных матриц достаточно, чтобы в классе эквивалентных матриц нашлась хотя бы одна (например, матрица в канонической форме), обладающая полезным свойством, позволяющим построить эффективный алгоритм оптимизации.

Для эквивалентных матриц можно указать некоторый набор операций, последовательное применение которых позволяет по данной матрице построить любую ей эквивалентную. Впервые такой набор операций был предложен в [17]. Позднее возможность нетривиальных преобразований над матрицей затрат на обслуживание задачи FL , не изменяющих оптимального решения задачи, была замечена другими авторами. Например, в [211] рассматриваются операции объединения потребителей и разъединения потребителей, аналогичные так называемой операции переноса подстолбца, рассмотренной в [17].

Уже отмечалось, что задача минимизации полиномов от булевых переменных эквивалентна задачам FL и $MINF_0$. Однако взаимосвязь этих трех задач можно считать более тесной, чем просто эквивалентность. Дело в том, что в качестве исходных данных для задач FL и $MINF_0$ можно рассматривать не пару матриц (F_0, C) , а пару матриц (F_0, \tilde{C}) , где \tilde{C} – каноническая форма матрицы C . А исходными данными для задачи минимизации полиномов от булевых переменных с неотрицательными коэффициентами является пара (F_0, \tilde{C}) , где \tilde{C} – матрица, в каждом столбце которой все ненулевые элементы одинаковые. Таким образом, для всех трех рассматриваемых задач переход

от одной к соответствующей другой, то есть задаче, по оптимальному решению которой строится оптимальное решение первой, не требует изменения исходных данных, а само построение оптимального решения исходной задачи осуществляется по оптимальному решению соответствующей задачи очевидным образом. Поэтому три указанные задачи можно считать разными формами записи одной и той же задачи.

3.1 Полиномы от булевых переменных. Псевдобулево программирование

Псевдобулевой функцией называют вещественную функцию $p(y_1, \dots, y_m)$ от переменных, принимающих значение 0 и 1. Поскольку всякую такую функцию можно представить в виде полинома

$$p(y_1, \dots, y_m) = \sum_{\alpha \subset I} c_\alpha \prod_{i \in \alpha} y_i,$$

где $I = \{1, \dots, m\}$, а c_α — действительные числа, то псевдобулеву функцию $p(y_1, \dots, y_m)$, для которой задано указанное выше представление, будем называть *полиномом от булевых переменных*. Выражение $c_\alpha \prod_{i \in \alpha} y_i$ назовем *членом полинома*, число

c_α — *коэффициентом* при этом члене, а величину $|\alpha|$ — *степенью* данного члена полинома. Член полинома степени $|\alpha| \geq 1$ назовем *линейным*, если $|\alpha|=1$, и *нелинейным* в противном случае.

Задача минимизации полинома от булевых переменных, то есть задача отыскания булева вектора (y_1, \dots, y_m) , доставляющего минимум полиному $p(y_1, \dots, y_m)$, записывается следующим образом:

$$\min \{p(y_1, \dots, y_m) \mid y_i \in \{0, 1\}, i \in I\}.$$

Эту задачу далее будем обозначать *MINP*.

Рассмотрим общую схему универсального алгоритма решения задачи *MINP*, построенного на основе упомянутого выше метода псевдобулева программирования. В основе метода лежит соображение о возможности редукции задачи *MINP* для произвольного полинома к такой же задаче, но для полинома с числом переменных на единицу меньше, чем у исходного полинома. Для осуществления такой редукции используется линейность исходного полинома $p(y_1, \dots, y_m)$ по переменной y_1 , позволяющая представить полином $p(y_1, \dots, y_m)$ следующим образом:

$$p_1(y_1, \dots, y_m) = y_1 g_1(y_2, \dots, y_m) + h_2(y_2, \dots, y_m).$$

Несложно понять, что существует оптимальное решение (y_1^*, \dots, y_m^*) задачи *MINP* для полинома $p(y_1, \dots, y_m)$ такое, что

$$y_1^* = \begin{cases} 1, & \text{если } g_1(y_2^*, \dots, y_m^*) \leq 0, \\ 0, & \text{если } g_1(y_2^*, \dots, y_m^*) > 0. \end{cases}$$

В связи с этим возникает идея построить полином от булевых переменных $y_1(y_2, \dots, y_m)$, обладающий свойством

$$y_1(y_2, \dots, y_m) = \begin{cases} 1, & \text{если } g_1(y_2^*, \dots, y_m^*) \leq 0, \\ 0, & \text{если } g_1(y_2^*, \dots, y_m^*) > 0. \end{cases}$$

Эта идея о возможности представить переменную y_1 как полином от других переменных составляет суть метода псевдобулева программирования. Используя полином $y_1(y_2, \dots, y_m)$, получаем полином

$$p_2(y_2, \dots, y_m) = y_1(y_2, \dots, y_m)g_1(y_2, \dots, y_m) + h_1(y_2, \dots, y_m),$$

который по построению обладает следующим свойством. Если (y_2^*, \dots, y_m^*) – оптимальное решение задачи *MINP* для полинома $p_2(y_2, \dots, y_m)$, то $(y_1^*, y_2^*, \dots, y_m^*)$, где $y_1^* = y_1(y_2^*, \dots, y_m^*)$, – оптимальное решение задачи *MINP* для исходного полинома $p_1(y_1, \dots, y_m)$.

Отсюда, с учетом того, что задача *MINP* для полинома $p_m(y_m) = y_m g_m + h_m$ от одной переменной является просто решаемой, ясно, как работает алгоритм, построенный на основе метода псевдобулева программирования.

Алгоритм состоит из двух этапов. Первый (основной) этап включает m шагов. На k -м шаге, $k=1, \dots, m-1$, строятся полиномы $p_k(y_k, \dots, y_m)$ и $y_k(y_{k+1}, \dots, y_m)$. На m -м шаге строится полином $p_m(y_m) = y_m g_m + h_m$ и полагается $y_m=1$, если $g_m \leq 0$, и $y_m=0$, если $g_m > 0$.

После этого начинается второй этап — восстановление оптимального решения (y_1^*, \dots, y_m^*) . Он, так же как и первый включает в себя m шагов. На первом шаге полагается $y_m^* = y_m$. Далее на k -м, $k=2, \dots, m$, шаге с использованием уже найденных на предыдущих шагах величин $y_{m-k+2}^*, \dots, y_m^*$ вычисляется значение $(m+k-1)$ -й компоненты оптимального решения по формуле

$$y_{m-k+1}^* = y_{m-k+1}(y_{m-k+2}^*, \dots, y_m^*).$$

Из сказанного ясно, как выглядит общая схема алгоритма псевдобулева программирования. Однако о конкретных алгоритмах решения задачи *MINP* для полиномов того или иного вида можно говорить только после конкретизации процедуры построения полиномов $y_k(y_{k+1}, \dots, y_m)$, $k=1, \dots, m-1$, обладающих необходимыми свойствами. К сожалению, построение каждого полинома $y_k(y_{k+1}, \dots, y_m)$ связано с отысканием всех решений нелинейного неравенства $g_k(y_{k+1}, \dots, y_m) \leq 0$, что в общем случае представляет собой задачу сравнимую по сложности с исходной задачей минимизации полинома $p_1(y_1, \dots, y_m)$. Предложенный в [166, 167] универсальный способ решения задачи

поиска всех решений неравенства $g_k(y_{k+1}, \dots, y_m) \leq 0$ представляет собой весьма громоздкую и трудоемкую процедуру. Эта процедура предполагает отыскание всех решений некоторого линейного неравенства с числом переменных, определяемых количеством членов полинома $g_k(y_{k+1}, \dots, y_m)$, которое может быть существенно большим, чем число переменных.

В силу сказанного, в конечном счете, алгоритм псевдобулева программирования, как универсальный способ решения задачи *MINP* оказывается малоприменимым с практической точки зрения. Вместе с тем идея, лежащая в основе этого метода является продуктивной и может быть использована при исследовании специальных классов полиномов, специфика которых позволяет эффективно строить полиномы $y_k(y_{k+1}, \dots, y_m)$, $k=1, \dots, m-1$.

3.2 Полиномы с неотрицательными коэффициентами

Полином $p_0(y_1, \dots, y_m)$, не имеющий отрицательных коэффициентов при линейных членах, назовем *полиномом с неотрицательными коэффициентами*.

Полином $p_0(y_1, \dots, y_m)$ с неотрицательными коэффициентами назовем заданным в *канонической форме*, если он представлен в виде

$$p_0(y_1, \dots, y_m) = a_0 + \sum_{i \in I} f_i (1 - y_i) + \sum_{s=1}^S b_s \prod_{i \in \beta_s} y_i,$$

где $f_i > 0$, $i \in I$; β_s , $s = 1, \dots, S$ – попарно различные подмножества множества I ; $b_s > 0$, $s = 1, \dots, S$. Понятно, что за счет линейных членов, которые могут присутствовать как в первой группе слагаемых, так и во второй, полином представим в канонической форме неоднозначно. Далее, если не оговорено противное, рассматриваемые полиномы будем считать заданными в канонической форме, то есть в форме, при которой члены полинома разделены на две группы: к первой группе принадлежат линейные члены с отрицательными коэффициентами, а ко второй – члены с положительными коэффициентами.

Для заданного в канонической форме полинома $p_0(y_1, \dots, y_m)$ с неотрицательными коэффициентами определим матрицу $H=(h_{is})$ размера $m \times S$, называемую *характеристической*, следующим образом:

$$h_{is} = \begin{cases} 0, & \text{если } i \in \beta_s, \\ 1, & \text{иначе.} \end{cases}$$

Если матрица $H=(h_{is})$ размера $m \times S$ является характеристической для полинома $p_0(y_1, \dots, y_m)$, то в канонической форме данный полином записывается следующим образом:

$$p_0(y_1, \dots, y_m) = a_0 + \sum_{i \in I} f_i(1 - y_i) + \sum_{s=1}^S b_s \prod_{i|h_{is}=0} y_i.$$

Задача минимизации полинома с неотрицательными коэффициентами $p_0(y_1, \dots, y_m)$ формально записывается следующим образом:

$$\min \{p_0(y_1, \dots, y_m) \mid y_i \in \{0,1\}, i \in I\}.$$

Далее такую задачу будем обозначать $MINP_0$. Исходными данными задачи $MINP_0$ считаем пару матриц (F_0, \tilde{C}) , где $F_0 = (f_i)$ – вектор-столбец длины m и $\tilde{C} = (b_s h_{is})$ – матрица размера $m \times S$.

Нашей ближайшей целью будет показать, что если задача FL может быть эквивалентным образом переформулирована в виде задачи $MINP_0$.

Пусть (i_1^j, \dots, i_m^j) – перестановка номеров строк матрицы $C = (c_{ij})$, порожденная j -м столбцом этой матрицы. Напомним, что такая перестановка упорядочивает элементы столбца по возрастанию, и при этом определяются неотрицательные величины $\Delta c_0^j = c_{i_1^j j}$, $\Delta c_k^j = c_{i_{k+1}^j j} - c_{i_k^j j}$, $k = 1, \dots, m-1$, называемые коэффициентами роста элементов j -го столбца.

Справедлива следующая лемма, позволяющая записать задачу FL как задачу $MINP_0$.

Лемма 1.3 Пусть (i_1^j, \dots, i_m^j) перестановка строк матрицы C , порожденная ее j -м столбцом. Тогда для любого булева вектора (y_1, \dots, y_m) имеет место равенство

$$\min_{i|y_i=0} c_{ij} = \Delta c_0^j + \sum_{k=1}^{m-1} \Delta c_k^j y_{i_1^j} \cdot \dots \cdot y_{i_k^j}.$$

Доказательство. Если (y_1, \dots, y_m) – единичный вектор, то равенство, очевидно, выполняется. Пусть p , $1 \leq p \leq m$ – наименьший номер, для которого $y_{i_p^j} = 0$. Тогда можем написать

$$\begin{aligned} \min_{i|y_i=0} c_{ij} &= c_{i_p^j j}, \\ \Delta c_0^j + \sum_{k=1}^{m-1} \Delta c_k^j y_{i_1^j} \cdot \dots \cdot y_{i_k^j} &= \Delta c_0^j + \sum_{k=1}^{p-1} \Delta c_k^j = c_{i_p^j j}. \end{aligned}$$

Это доказывает утверждение леммы.

Теорема 1.1 Задача $MINP_0$ и задача $MINF_0$ эквивалентны.

Доказательство. Рассмотрим задачу $MINF_0$ с парой матриц (F_0, C) , где $F_0 = (f_i)$ и $C = (c_{ij})$. Используя равенство из леммы 1.3, паре матриц (F_0, C) поставим в соответствие полином $p_0(y_1, \dots, y_m)$ с неотрицательными коэффициентами, определяемый следующим образом:

$$\begin{aligned}
p_0(y_1, \dots, y_m) &= \sum_{i \in I} f_i(1 - y_i) + \sum_{j \in J} \min_{i | y_i = 0} c_{ij} = \\
&= \sum_{i \in I} f_i(1 - y_i) + \sum_{j \in J} (\Delta c_0^j + \sum_{k=1}^{m-1} \Delta c_k^j y_{i_1^j} \cdot \dots \cdot y_{i_k^j}) = \\
&= \sum_{j \in J} \Delta c_0^j + \sum_{i \in I} f_i(1 - y_i) + \sum_{j \in J} \sum_{k=1}^{m-1} \Delta c_k^j y_{i_1^j} \cdot \dots \cdot y_{i_k^j}.
\end{aligned}$$

Обозначим через β_s и b_s , $s = 1, \dots, S$, соответственно характеристические множества матрицы C и веса этих характеристических множеств. Тогда полученный полином $p_0(y_1, \dots, y_m)$ может быть записан следующим образом:

$$p_0(y_1, \dots, y_m) = \sum_{j \in J} \min_{i \in I} c_{ij} + \sum_{i \in I} f_i(1 - y_i) + \sum_{s=1}^S b_s \prod_{i \in \beta_s} y_i.$$

Полином $p_0(y_1, \dots, y_m)$ построенный указанным выше способом по паре матриц (F_0, C) , назовем *соответствующим* паре матриц (F_0, C) . Отметим, что в силу построения полинома $p_0(y_1, \dots, y_m)$ вектор (y_1, \dots, y_m) является оптимальным решением задачи $MINP_0$ для полинома $p_0(y_1, \dots, y_m)$ тогда и только тогда, когда множество $X = \{i \in I \mid y_i = 0\}$ является оптимальным решением задачи $MINF_0$ с парой матриц (F_0, C) . Отсюда вытекает, что задача $MINF_0$ сводится к задаче $MINP_0$.

Покажем обратную сводимость. Пусть задан полином $p_0(y_1, \dots, y_m)$ с неотрицательными коэффициентами. Запишем его в канонической форме

$$p_0(y_1, \dots, y_m) = a_0 + \sum_{i \in I} f_i(1 - y_i) + \sum_{s=1}^S b_s \prod_{i \in \beta_s} y_i$$

и пусть матрица $H = (h_{is})$ размера $m \times S$ будет характеристической матрицей этого полинома. Рассмотрим пару матриц (F_0, \tilde{C}) , где $F_0 = (f_i)$ – вектор-столбец длины m и $\tilde{C} = (b_s h_{is})$ – матрица размера $m \times S$. Заметим, что исходный полином $p_0(y_1, \dots, y_m)$ является соответствующим для построенной пары матриц (F_0, \tilde{C}) . Поэтому, если X – оптимальное решение задачи $MINF_0$ с парой (F_0, \tilde{C}) , то вектор (y_1, \dots, y_m) , где

$$y_i = \begin{cases} 0, & \text{если } i \in X, \\ 1, & \text{иначе} \end{cases}$$

будет оптимальным решением задачи $MINP_0$ для исходного полинома $p_0(y_1, \dots, y_m)$. Это доказывает требуемую сводимость и завершает доказательство теоремы.

Из доказательства ясно, каким образом задача FL переписывается в виде задачи минимизации полинома от булевых переменных. При построении по паре матриц (F_0, C) соответствующего полинома $p_0(y_1, \dots, y_m)$ используются перестановки множества

I , задаваемые столбцами матрицы C и упорядочивающие элементы столбцов по возрастанию.

При построении в отмеченной выше работе [169] псевдобулевой функции, соответствующей задаче FL для всякого $j \in J$ определяется матрица (u_{ijk}) размера $m \times m$, элементы которой задаются следующим образом:

$$u_{ijk} = \begin{cases} 0, & \text{если } c_{ij} < c_{kj}, \\ 0, & \text{если } c_{ij} = c_{kj} \text{ и } i < k, \\ 1, & \text{иначе.} \end{cases}$$

Несложно понять с учетом определения матрицы $(u_{ijk}), j \in J$, что существует оптимальное решение $((z_i), (x_{ij}))$ задачи FL такое, что для любых $i \in I, j \in J$ выполняется равенство

$$x_{ij} = z_i \prod_{k \in I} (1 - u_{ikj} z_k).$$

Поэтому соответствующая псевдобулева функция записывается следующим образом:

$$\sum_{i \in I} f_i z_i + \sum_{j \in J} \sum_{i \in I} c_{ij} z_i \prod_{k \in I} (1 - u_{ikj} z_k) + \Phi \prod_{i \in I} (1 - z_i),$$

где $\Phi = \sum_{j \in J} \max_{i \in I} c_{ij}$ — достаточно большая положительная величина, гарантирующая

существование ненулевого оптимального решения задачи минимизации данной функции.

Указанную псевдобулеву функцию несложно привести к полученному в ходе доказательства теоремы 1.1 полиному от булевых переменных, вид которого, как представляется, более наглядный и удобный для исследования. Действительно, для всякого $j \in J$ матрица (u_{ikj}) задает на множестве I бинарное отношение, упорядочивающее элементы j -го столбца матрицы C по убыванию, то есть определяет перестановку i_1^j, \dots, i_m^j , которая есть перестановка, задаваемая j -м столбцом матрицы C . Следовательно, можем написать

$$\begin{aligned} \sum_{i \in I} c_{ij} z_i \prod_{k \in I} (1 - z_k u_{ikj}) &= \sum_{l=1}^m c_{i_l^j j} z_{i_l^j} \prod_{k < l} (1 - z_{i_k^j}) = \\ &= \sum_{l=1}^m c_{i_l^j j} \prod_{k < l} (1 - z_{i_k^j}) - \sum_{l=1}^m c_{i_l^j j} \prod_{k \leq l} (1 - z_{i_k^j}) = \\ &= c_{i_1^j j} + \sum_{l=1}^{m-1} (c_{i_{l+1}^j j} - c_{i_l^j j}) \prod_{k \leq l} (1 - z_{i_k^j}) - c_{i_m^j j} \prod_{i \in I} (1 - z_i). \end{aligned}$$

Отсюда получаем, что рассматриваемая псевдобулева функция преобразуется в полином $p_0(y_1, \dots, y_m)$, соответствующий паре матриц (F_0, C) .

3.3 Эквивалентные матрицы

Из доказательства теоремы 1.1 видно, как устроен полином с неотрицательными коэффициентами $p_0(y_1, \dots, y_m)$, соответствующий паре матриц (F_0, C) . Члены полинома с положительными коэффициентами порождаются характеристическими множествами матрицы C . Если β – характеристическое множество с весом равным b , то этому множеству соответствует член полинома $b \prod_{i \in \beta} y_i$. Понятно поэтому, что если H – характе-

ристическая матрица для матрицы C , то H будет характеристической матрицей полинома с неотрицательными коэффициентами $p_0(y_1, \dots, y_m)$, соответствующий паре (F_0, C) .

Из сказанного несложно понять, какими свойствами должны обладать матрицы C и C' , чтобы полиномы, соответствующие парам (F_0, C) и (F_0, C') , совпадали. Пусть матрицы C и C' таковы, что их характеристические матрицы H и H' совпадают. Пусть β_1, \dots, β_S – веса столбцов матрицы H , а $\beta'_1, \dots, \beta'_S$ – веса столбцов матрицы H' и пусть $\beta_s = \beta'_s, s = 1, \dots, S$, то есть веса соответствующих столбцов матриц H и H' равны. Такие матрицы C и C' назовем эквивалентными.

Эквивалентными будут, например, следующие две матрицы

$$C = \begin{bmatrix} 6 & 3 & 3 \\ 4 & 2 & 4 \\ 3 & 3 & 1 \\ 1 & 6 & 2 \end{bmatrix} \quad \text{и} \quad C' = \begin{bmatrix} 5 & 3 & 4 \\ 3 & 2 & 5 \\ 0 & 3 & 4 \\ 1 & 6 & 2 \end{bmatrix},$$

имеющие одинаковую характеристическую матрицу

$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

и одинаковые веса столбцов этой матрицы, равные соответственно 3, 1, 2, 1, 2, 1, 2.

Понятно, что если C и C' эквивалентные матрицы, то полиномы, соответствующие парам (F_0, C) и (F_0, C') , отличаются разве что на константу. Следовательно, если C и C' – эквивалентные матрицы, то оптимальные решения задач $MINF_0$ с парой (F_0, C) и парой (F_0, C') совпадают.

Таким образом, эквивалентные матрицы образуют класс матриц, обладающих тем свойством, что задачи FL с парами (F_0, C) , где C – матрица из данного класса эквивалентности, имеют одинаковые оптимальные решения. В таком классе матриц естественно выделить матрицу $\tilde{C} = (b_s h_{is})$, где $H = (h_{is})$ – характеристическая матрица для матриц C из рассматриваемого класса, а (b_s) – вектор весов столбцов характеристической матрицы H . Матрица \tilde{C} является канонической формой для всех матриц C из данного класса эквивалентности, поэтому ее естественно считать представителем данного

класса эквивалентных матриц. Паре матриц (F_0, \tilde{C}) соответствует полином с неотрицательными коэффициентами

$$p_0(y_1, \dots, y_m) = \sum_{i \in I} f_i (1 - y_i) + \sum_{s=1}^S b_s \prod_{i | h_{is} = 0} y_i.$$

Этот полином отличается от полинома, соответствующего паре (F_0, C) , где C – матрица из данного класса эквивалентности, только на константу. Поэтому данный полином назовем *соответствующим классу эквивалентных матриц*.

Понятие эквивалентности матриц является чрезвычайно полезным в плане построения эффективно разрешимых частных случаев задачи FL . Выделение частных случаев, специфика которых позволяет строить эффективные алгоритмы, как мы увидим в дальнейшем, производится чаще всего посредством наложения некоторых дополнительных условий на матрицу затрат на обслуживание C . Эти условия обуславливают некоторые полезные свойства матрицы C , наличие которых позволяет строить эффективные алгоритмы решения задачи FL . Используя эквивалентность матриц, можно причислить к классу эффективно разрешаемых задач FL не только задачу с матрицей C , обладающей полезным свойством, но и задачу с матрицей C' , этим свойством не обладающей, но эквивалентной матрице C . Это означает, что наличие полезного свойства можно требовать не от исходной матрицы C , а, например, от канонической формы \tilde{C} матрицы C или даже от характеристической матрицы H , которые, как уже отмечалось, имеют более простое строение и поэтому легче поддаются исследованию на наличие тех или иных полезных свойств.

3.4 Операции над эквивалентными матрицами

Напомним, что матрицы C и C' называются эквивалентными, если их характеристические матрицы H и H' совпадают, а веса у соответствующих столбцов матриц H и H' равны. Другими словами, матрицы C и C' эквивалентны, если множества характеристических множеств у матриц C и C' совпадают, а веса у соответствующих характеристических множеств матриц C и C' равны. Отсюда понятно, что такая операция над исходной матрицей C как, например, перестановка столбцов не нарушает множество характеристических множеств и не изменяет веса этих множеств. Следовательно, эта операция не выводит полученную в результате матрицу C' из класса матриц эквивалентных исходной матрице C .

Однако одной операции перестановки столбцов, очевидно, не достаточно, чтобы по исходной матрице построить любую ей эквивалентную матрицу. Поэтому рассмотрим совокупность преобразований над матрицами, результатом каждого из которых является матрица, эквивалентная исходной и последовательное применение которых позволяет по исходной матрице построить любую ей эквивалентную матрицу.

Рассмотрим исходную матрицу $C = (c_{ij})$ размера $m \times n$. Кроме упомянутой выше операции *перестановки* p -го и q -го столбцов матрицы C , дающей эквивалентную мат-

рицу C' , укажем также на две другие тривиальные операции, приводящие к эквивалентным матрицам. Это операция *удаления* из матрицы C p -го столбца, имеющего одинаковые элементы, и операцию *добавления* к матрице $(m+1)$ -го вектор-столбца с одинаковыми элементами. В результате первой операции получается матрица размера $m \times (n-1)$, а в результате второго — матрица размера $m \times (n+1)$, у которой последний столбец совпадает с добавленным вектор-столбцом. Оба эти преобразования, очевидно, не изменяют множества характеристических подмножеств матрицы C и не влияют на величину весов характеристических множеств. Поэтому данные операции приводят к матрицам эквивалентным исходной матрице C .

Определим еще две несложные операции, которые, также как и три указанные выше, приводят к эквивалентным матрицам и которые вместе с этими тремя операциями позволяют по исходной матрице построить любую ей эквивалентную матрицу.

Пусть p -й и q -й столбцы исходной матрицы C порождают одну и ту же перестановку (i_1, \dots, i_m) множества номеров строк $I = \{1, \dots, m\}$. Будем говорить, что матрица $C' = (c'_{ij})$ размера $m \times n$ получена из матрицы C в результате *сложения* p -го и q -го столбцов, $p < q$, если

$$\begin{aligned} c'_{ip} &= c_{ip} + c_{iq}, \quad i \in I; \\ c'_{iq} &= 0, \quad i \in I, \end{aligned}$$

а все остальные столбцы матрицы C совпадают с соответствующими столбцами исходной матрицы C .

Несложно увидеть, что операция сложения p -го и q -го столбцов не выводит полученную в результате матрицу C' из класса матриц, эквивалентных исходной матрице C . Действительно, заметим, прежде всего, что перестановка, порождаемая p -м и q -м столбцами матрицы C порождается так же и p -м столбцом матрицы C' . Кроме того, заметим, что при любом $k = 1, \dots, m-1$ имеет место равенство

$$\Delta c'_{kp} = \Delta c_{kp} + \Delta c_{kq}.$$

Отсюда следует, что при любом $k = 1, \dots, m-1$ множество $w_k = \{i_1, \dots, i_k\}$ одновременно является характеристическим множеством p -го столбца матрицы C' и подматрицы C_{pq} , состоящей из p -го и q -го столбцов матрицы C , и, кроме того, веса множества w_k , как характеристического множества p -го столбца матрицы C' и подматрицы C_{pq} равны. Это и доказывает эквивалентность матриц C и C' .

Пусть p -й столбец матрицы C и некоторый столбец $(c_1, \dots, c_m)^T$ порождают одну и ту же перестановку (i_1, \dots, i_m) и пусть для коэффициентов роста Δc_k , $k = 0, 1, \dots, m-1$, рассматриваемого вектор-столбца и коэффициентов роста Δc_{kp} , $k = 0, 1, \dots, m-1$, p -го столбца матрицы C выполняются неравенства

$$\Delta c_k \leq \Delta c_{kp}, \quad k = 0, 1, \dots, m-1.$$

Будем говорить, что матрица $C' = (c'_{ij})$ размера $m \times (n+1)$ получается из матрицы C в результате *вычитания* из p -го столбца вектор-столбца $(c_1, \dots, c_m)^T$, если

$$c'_{ip} = c_{ip} - c_i, \quad i \in I;$$

$$c'_{im+1} = c_i, \quad i \in I,$$

а все остальные столбцы матрицы C' совпадают с соответствующими столбцами матрицы C .

Эта операция, как и рассмотренные выше преобразования, приводят к матрице C' , эквивалентной исходной матрице C . Действительно, так же как и в случае операции сложения, заметим, прежде всего, что p -й столбец матрицы C' порождает ту же перестановку (i_1, \dots, i_m) , что и p -й столбец матрицы C . Для этого покажем, что для всякого $k = 1, \dots, m-1$ имеем $c'_{ikp} \leq c'_{i_{k+1}p}$. В самом деле, поскольку $\Delta c_k \leq \Delta c_{kp}$, то можем написать

$$c'_{ikp} = c_{ikp} - c_{i_k} \leq c_{i_{k+1}p} - c_{i_{k+1}} = c'_{i_{k+1}p}.$$

Кроме того, поскольку $\Delta c_0 \leq \Delta c_{0p}$, то $c'_{i_1p} \geq 0$. Заметим также, что для любого $k = 1, \dots, m-1$ справедливо равенство

$$\Delta c'_{kp} + \Delta c'_k = \Delta c_{kp}.$$

Из сказанного следует, что при любом $k = 1, \dots, m-1$ множество $w_k = \{i_1, \dots, i_k\}$ одновременно является характеристическим множеством p -го столбца матрицы C и подматрицы C'_{pn+1} , состоящей из p -го и $(n+1)$ -го столбцов матрицы C' , и, кроме того, вес характеристического множества w_k p -го столбца матрицы C и вес характеристического множества w_k матрицы C'_{pn+1} равны. Это доказывает эквивалентность матриц C и C' .

Таким образом, рассмотрены пять операций над матрицами: перестановка столбцов, добавление и исключение столбца с одинаковыми элементами, сложение столбцов и вычитание из столбца вектор-столбца. Применение каждой из этих операций приводит к матрице, эквивалентной исходной. Покажем, что этих пяти операций достаточно, чтобы по данной матрице построить любую ей эквивалентную.

Для этого покажем, прежде всего, что с использованием данных операций по матрице C можно построить ее каноническую форму \tilde{C} . Для этого рассмотрим процедуру разложения первого столбца матрицы C на столбцы с одинаковыми ненулевыми элементами.

Пусть исходная матрица C имеет размерность $m \times n$ и пусть (i_1, \dots, i_m) – перестановка, задаваемая первым столбцом, а Δ_k , $k = 0, 1, \dots, m-1$, – коэффициенты роста элементов этого столбца. Процедура разложения первого столбца матрицы C на столбцы с одинаковыми ненулевыми элементами состоит из $m-1$ основных шагов и заключительного шага.

На первом шаге рассматривается исходная матрица C . Если $\Delta_1 > 0$ производится операция вычитания из первого столбца матрицы C вектор-столбца $(c_1, \dots, c_m)^T$, определяемого следующим образом:

$$c_{i_k} = \begin{cases} 0, & \text{если } k \leq 1, \\ \Delta_1, & \text{иначе.} \end{cases}$$

Полученная в результате матрица обозначается через C_2 и начинается второй шаг.

Пусть к очередному l -му шагу, $l \leq m - 1$ получена матрица C_l . Если $\Delta_l > 0$, из первого столбца матрицы C_l вычитается вектор-столбец $(c_1, \dots, c_m)^T$, определяемый следующим образом:

$$c_{i_k} = \begin{cases} 0, & \text{если } k \leq l, \\ \Delta_l, & \text{иначе.} \end{cases}$$

Полученная в результате матрица обозначается через C_{l+1} . Если $l < m$, то начинается следующий основной шаг, в противном случае начинается заключительный шаг.

Несложно понять, что в результате l -го основного шага получается матрица C_{l+1} такая, что, по крайней мере, $l + 1$ элемент первого столбца этой матрицы совпадает с величиной Δ_0 , равной наименьшему элементу первого столбца исходной матрицы C . Поэтому к началу заключительного шага имеется матрица C_m , у которой первый столбец состоит из одинаковых элементов равных Δ_0 . На заключительном шаге этот столбец удаляется из матрицы C_m , после чего получается матрица C'_m и процедура разложения первого столбца матрицы C считается законченной.

Из построения матрицы C'_m видно, что она получена из матрицы C с помощью двух из пяти рассмотренных операций и, следовательно, она эквивалентна матрице C . Кроме того, отметим, что матрица C'_m отличается от исходной матрицы C тем, что из нее удален первый столбец матрицы C и добавлены столбцы с номерами j , $j \geq m$, такие, что ненулевые элементы каждого из этих столбцов одинаковые.

Используя рассмотренную процедуру разложения первого столбца, покажем теперь, как с помощью рассмотренных операций по исходной матрице C размера $m \times n$ построить каноническую форму \tilde{C} этой матрицы. Соответствующий алгоритм состоит из n основных этапов и заключительного этапа.

На первом этапе рассматривается исходная матрица C , к которой применяется процедура разложения первого столбца. Полученная в результате матрица обозначается C_2 и начинается второй этап.

Пусть к началу p -го этапа, $p \leq n$, построена матрица C_p . Первый столбец этой матрицы совпадает с p -м столбцом исходной матрицы C . К матрице C_p применяется процедура разложения первого столбца, в результате которой получается матрица C_{p+1} . Если $p < n$, то начинается следующий основной этап, в противном случае осуществляется переход к заключительному этапу.

На заключительном этапе рассматривается матрица C_{n+1} , каждый столбец которой имеет одинаковые ненулевые элементы. К данной матрице последовательно применяется операция сложения двух столбцов, имеющих одинаковое множество строк с ненулевыми элементами. Этот процесс продолжается до тех пор, пока не останется хотя бы одна пара столбцов с одинаковыми множествами ненулевых строк. После этого последовательно исключаются образовавшиеся в результате операции сложения нулевые столбцы.

Полученную в результате указанных действий матрицу обозначим C'_{n+1} . Из описания процедуры ее построения ясно, что, во-первых, матрица C'_{n+1} построена из исходной матрицы C в результате применения рассматриваемых операций и, следовательно, она эквивалентна матрице C . Во-вторых, столбцы матрицы C'_{n+1} попарно различны и каждый из них имеет нулевые и одинаковые ненулевые элементы. Поэтому, если в завершение заключительного этапа рассматриваемого алгоритма операциями перестановки столбцов упорядочить столбцы матрицы C'_{n+1} в соответствии с порядком столбцов характеристической матрицы для матрицы C , то полученная в результате матрица \tilde{C} будет канонической формой исходной матрицы C .

Таким образом, с помощью набора из пяти рассматриваемых операций по исходной матрице C может быть построена ее каноническая форма. Заметим далее, что каждую из рассматриваемых пяти операций можно обратить последовательным применением этих же операций. То есть заметим, что если матрица C' получена из матрицы C в результате применения какой-либо из рассмотренных пяти операций, то последовательное применение этих пяти операций позволяет по матрице C' построить исходную матрицу C .

Относительно первых трех из рассмотренных пяти операций: перестановки столбцов, удаления и добавления столбцов с одинаковыми элементами, это утверждение, очевидно справедливо.

Пусть матрица C' получена из матрицы C размера $m \times n$ в результате сложения p -го и q -го столбцов. Применим последовательно к матрице C' следующие операции. Вычтем из p -го столбца матрицы C' вектор-столбец, совпадающий с q -м столбцом исходной матрицы C . Далее переставим q -й и $(m + 1)$ -й столбцы полученной матрицы и, наконец, удалим $(m + 1)$ -й столбец полученной в результате перестановки столбцов матрицы. Итоговая матрица будет, очевидно, матрицей C .

Пусть теперь матрица C' получена из матрицы C в результате вычитания из p -го столбца этой матрицы некоторого вектор-столбца. Применим к матрице C' последовательно следующие операции. Сложим p -й и $(m + 1)$ -й столбцы матрицы C' и удалим $(m + 1)$ -й нулевой столбец. Полученная в результате матрица будет исходной матрицей C .

Сделанные замечания позволяют убедиться в справедливости приведенного выше предложения о том, что рассмотренных операций над матрицами достаточно для того, чтобы по матрице C построить любую ей эквивалентную матрицу C' .

Действительно, последовательным применением указанных операций можно по матрице C построить ее каноническую форму \tilde{C} , а по матрице C' — ее каноническую форму \tilde{C}' . Если обратить последовательность операций, которые по матрице C' строят матрицу \tilde{C}' , то получим, что с использованием рассматриваемых пяти операций по матрице \tilde{C}' можно построить матрицу C' . Но поскольку матрицы C и C' эквивалентны, то матрицы \tilde{C} и \tilde{C}' совпадают. Отсюда получаем, что последовательное применение

рассматриваемых операций позволяет по матрице C построить матрицу \tilde{C} , а затем по матрице \tilde{C} получить матрицу C' .

Разумеется, указанная выше последовательность операций, преобразующая матрицу C в матрицу C' не единственно возможная. Чаще всего существуют другие, более короткие последовательности операций, строящие матрицу C' по матрице C и не предусматривающие при этом построения канонической формы \tilde{C} .

В качестве примера рассмотрим последовательность операций, связывающую две рассмотренные ранее эквивалентные матрицы C и C' ,

$$C = \begin{pmatrix} 6 & 3 & 3 \\ 4 & 2 & 4 \\ 3 & 3 & 1 \\ 1 & 6 & 2 \end{pmatrix}, \quad C' = \begin{pmatrix} 5 & 3 & 4 \\ 3 & 2 & 5 \\ 0 & 3 & 4 \\ 1 & 6 & 2 \end{pmatrix}.$$

Вычтем последовательно из первого столбца матрицы C вектор-столбец, $(3, 3, 3, 1)^T$ и из третьего столбца — вектор-столбец $(1, 2, 1, 1)^T$. В полученной в результате этих операций матрице

$$\begin{pmatrix} 3 & 3 & 2 & 3 & 1 \\ 1 & 2 & 2 & 3 & 2 \\ 0 & 3 & 0 & 3 & 1 \\ 0 & 6 & 1 & 1 & 1 \end{pmatrix}$$

сложим первый и третий столбцы, а затем четвертый и пятый. В результате получим следующую матрицу

$$\begin{pmatrix} 5 & 3 & 0 & 4 & 0 \\ 3 & 2 & 0 & 5 & 0 \\ 0 & 3 & 0 & 4 & 0 \\ 1 & 6 & 0 & 2 & 0 \end{pmatrix},$$

Эта матрица после удаления третьего и пятого нулевых столбцов превращается в требуемую матрицу C' .

3.5 Три эквивалентные задачи

Из теоремы 1.1 вытекает, что три рассматриваемые выше задачи: FL , $MINF_0$ и $MINP_0$ взаимно сводимы. Более того, приведенные в ходе доказательства теоремы построения соответствующих задач указывают на взаимосвязь этих задач более тесную, чем эквивалентность.

Напомним, что если матрица C имеет характеристическую матрицу $H = (h_{ij})$ размера $m \times S$ с весами столбцов b_s , $s = 1, \dots, S$, то задача FL с исходными данными (F_0, C) или задача $MINF_0$ с парой (F_0, C) сводятся к задаче $MINP_0$ для полинома

$$p_0(y_1, \dots, y_m) = \sum_{i=1}^m f_i(1 - y_i) + \sum_{s=1}^S b_s \prod_{i|h_{is}=0} y_i$$

с характеристической матрицей H . С другой стороны, задача $MINP_0$ для полинома $p_0(y_1, \dots, y_m)$ с характеристической матрицей $H = (h_{ij})$ и вектором неотрицательных коэффициентов (b_s) сводится к задаче FL с исходными данными (F_0, \tilde{C}) и сводится к задаче $MINF_0$ с парой (F_0, \tilde{C}) , где $\tilde{C} = (b_s h_{is})$ – каноническая форма матрицы C .

Таким образом, поскольку понятие эквивалентности матриц позволяет ограничиться рассмотрением задач FL только с парами (F_0, \tilde{C}) , где \tilde{C} – матрица в канонической форме, то для всех трех рассматриваемых задач исходными данными можно считать пары (F_0, \tilde{C}) , то есть вектор-столбец F_0 , булеву матрицу $H = (h_{ij})$ и вектор весов столбцов (b_s) матрицы H . При этом для данных задач переход от одной задачи к соответствующей другой, то есть такой, по оптимальному решению которой строиться оптимальное решение первой, не требует изменения исходных данных. Кроме того, построение оптимального решения любой задачи по оптимальному решению другой представляет собой простую процедуру вычисления значений переменных одной задачи по значению переменных другой задачи. В этом смысле рассматриваемые три задачи представляют собой не три различные задачи, а одну задачу, имеющую разные формы записи.

Понятно в силу сказанного, что если для одной из трех задач обнаружено некоторое полезное свойство исходных данных (матрицы \tilde{C} или матрицы H), позволяющее построить полиномиальный алгоритм, то тем самым автоматически выделены эффективно разрешимые частные случаи и двух других задач. В частности, если найден некоторый класс полиномов с неотрицательными коэффициентами, специфика которых позволяет построить эффективный алгоритм минимизации, то это означает, что выделен некоторый класс матриц такой, что задача FL с матрицей затрат на обслуживание из этого класса будет эффективно разрешима.

В заключение отметим, что поскольку в качестве исходных данных задачи FL можно рассматривать пару (F_0, \tilde{C}) , где \tilde{C} – каноническая форма эквивалентных матриц, число различных конкретных задач FL определяется числом различных пар (F_0, \tilde{C}) . Если же предположить дополнительно, что свойство конкретной задачи FL быть трудно или легко решаемой определяется, прежде всего, строением характеристической матрицы H и в меньшей степени зависит от значений элементов вектор-столбца F_0 и вектора весов столбцов матрицы H , то получаем, что число существенно различных конкретных задач FL столько же, сколько различных характеристических матриц. Число же таких матриц поддается прямому подсчету. Число булевых матриц размера

$m \times n$ равняется, очевидно, величине $C_{2^m}^n$, а количество булевых матриц, имеющих m

строк, равняется $\sum_{n=1}^{2^m} C_{2^m}^n = 2^{2^m} - 1$.

4 Задача о покрытии множества системой подмножеств

В предыдущих параграфах установлена эквивалентность трех задач: FL , $MINF_0$, и $MINP_0$, и отмечено, что связь между этими задачами является более тесной, чем просто взаимосводимость, поскольку у соответствующих задач исходные данные задаются одной и той же парой матриц (F_0, \tilde{C}) , а построение оптимального решения сводится к нетрудоемкому вычислению оптимальных значений переменных одной задачи по оптимальным значениям переменных другой задачи. Это означает, в частности, что если для одной из трех задач выделена некоторая подзадача, специфика исходных данных которой позволяет построить эффективный алгоритм оптимизации, то тем самым автоматически найдены эффективно разрешимые частные случаи и для двух других задач. При этом «мощность» эффективно разрешимого подкласса относительно всего класса задач будет одинаковой для всех трех рассматриваемых задач.

В настоящем параграфе к трем указанным взаимосводимым задачам добавляется еще две хорошо известные и изученные NP-трудные задачи: задача о покрытии множествами [40, 49] и ее частный случай так называемая задача о вершинном покрытии. Связь этих задач с задачами FL , $MINF_0$, и $MINP_0$ исследована в работах [2, 6]. Указанные задачи составляют пятерку взаимосводимых задач, оптимальное решение каждой из которых достаточно просто строится по оптимальному решению другой соответствующей ей задачи. Однако связь задачи о покрытии множествами с тремя ранее рассмотренными задачами не позволяет считать ее еще одной формой записи задачи размещения средств обслуживания. Это выражается, в частности, в следующем. Если выделен эффективно разрешимый подкласс задачи FL , то тем самым выделен и эффективно разрешимый подкласс задачи о покрытии множествами с такой же относительной мощностью, что и относительная мощность выделенного подкласса задачи FL . С другой стороны, если имеется эффективно разрешимый подкласс задачи о покрытии множествами, то не всегда можно рассчитывать на то, что удастся указать эффективно разрешимый подкласс задачи FL такой же относительной мощности, как рассматриваемый подкласс задач о покрытии множествами. В этом смысле можно говорить, что задача о покрытии множествами более простая, чем три ранее рассмотренные задачи.

Следует подчеркнуть также, что сводимость задач FL , $MINF_0$, и $MINP_0$ к задачам о покрытии множествами и о вершинном покрытии имеют важное вспомогательное

значение и в дальнейшем будут использованы как при построении эффективных алгоритмов, так и при доказательстве NP-трудности.

4.1 Эквивалентность задачи о покрытии множествами и задачи выбора строк пары матриц

Пусть заданы конечные множества $J=\{1, \dots, n\}$, $I=\{1, \dots, m\}$ и булева матрица $A=(a_{ij})$ размера $m \times n$, строка с номером i которой определяет подмножество $J_i = \{j \in J \mid a_{ij} = 1\}$ множества J . Считаем, что матрица A не имеет нулевых столбцов и, следовательно, объединение множеств J_i , $i \in I$, покрывает множество J . Пусть заданы величины f_i , $i \in I$, определяющие «веса» рассматриваемых подмножеств. Задача о покрытии множества J системой подмножеств J_i , $i \in I$, состоит в выборе такого множества подмножеств, которые бы в совокупности покрывали множество J и имели бы наименьший суммарный вес.

Если ввести переменные $x_i \in \{0, 1\}$, $i \in I$, такие что $x_i = 1$, когда множество J_i используется для покрытия множества J , и $x_i = 0$, в противном случае, то *задача о покрытии множествами* формулируется в виде задачи целочисленного линейного программирования следующим образом:

$$\begin{aligned} \min_{(x_i)} \sum_{i \in I} f_i x_i; \\ \sum_{i \in I} a_{ij} x_i \geq 1, \quad j \in J; \\ x_i \in \{0, 1\}, \quad i \in I. \end{aligned}$$

Эту задачу далее будем обозначать как *SC*. Исходные данные задачи *SC* представляют собой пару матриц (F_0, A) , где F_0 – диагональная матрица с диагональными элементами f_i , $i \in I$, или вектор столбец (f_i) длины m и $A=(a_{ij})$ – матрица ограничений. В случае, когда $f_i = 1$ для всякого $i \in I$, задача *SC* называется *задачей о покрытии минимальной мощности*. Такую задачу будем обозначать далее через *SC1*.

Некоторые частные случаи задачи *SC* имеют специальные названия. Если для всякого $j \in J$ имеем

$$\sum_{i \in I} a_{ij} = 2,$$

то задача *SC* называется *задачей о вершинном покрытии* и обозначается далее через *VC*. В задаче *VC* матрицу ограничений $A=(a_{ij})$ можно рассматривать как матрицу инцидентий некоторого графа $G=(I, J)$ с множеством вершин $I=\{1, \dots, m\}$ и множеством ребер $J=\{1, \dots, n\}$.

Если для всякого $i \in I$ имеем

$$\sum_{j \in J} a_{ij} = 2,$$

то задача SC называется *задачей о реберном покрытии* и обозначается далее через EC . В этой задаче матрицу ограничений можно рассматривать как транспонированную матрицу инцидентий графа $G = (I, J)$ с множеством вершин $I = \{1, \dots, m\}$ и множеством ребер $J = \{1, \dots, n\}$.

Если задан неориентированный граф $G = (V, E)$ с множеством вершин V и дуг E , то задачу VC для данного графа можно сформулировать, не используя матрицы инцидентий. Произвольным образом выбрав направление ребер, сделаем этот граф ориентированным, чтобы определить для каждого ребра $e \in E$ его начальную вершину $v_1(e)$ и его конечную вершину $v_2(e)$. Задача VC для графа G записывается следующим образом:

$$\begin{aligned} \min_{(x_v)} \sum_{v \in V} f_v x_v; \\ x_{v_1(e)} + x_{v_2(e)} \geq 1, \quad e \in E; \\ x_v \in \{0, 1\}, \quad v \in V. \end{aligned}$$

Покажем, что задача SC эквивалентна каждой из рассмотренных ранее задач FL , $MINF_0$ и $MINP_0$. Удобней всего доказать эквивалентность задачи SC и задачей $MINF_0$.

Теорема 1.2 Задача $MINF_0$ и задача SC эквивалентны.

Доказательство. Рассмотрим задачу SC , исходным данными которой (F_0, A) , где $A = (a_{ij})$ – матрица размера $m \times n$, поставим в соответствие пару матриц (F_0, C) такую, что $C = (c_{ij})$ – матрица размера $m \times n$ с элементами $c_{ij} = \Phi(1 - a_{ij})$, где $\Phi > \sum_{i \in I} f_i$. Опти-

мальному решению X^* задачи $MINF_0$ с построенной парой матриц (F_0, C) поставим в соответствие решение (x_i) исходной задачи SC , определенное следующим образом:

$$x_i = \begin{cases} 1, & \text{если } i \in X^*, \\ 0, & \text{иначе.} \end{cases}$$

Воспользовавшись критерием оптимальности (лемма 1.1), покажем, что (x_i) – оптимальное решение. Поскольку X^* – оптимальное решение, то в силу выбора величины Φ имеем $X^* \cap \{i \in I \mid a_{ij} = 1\} \neq \emptyset$. Поэтому (x_i) – допустимое решение. По той же причине справедливы равенства

$$f_0(X^*) = \sum_{i \in X^*} f_i + \sum_{j \in J} \min_{i \in X^*} \Phi(1 - a_{ij}) = \sum_{i \in X^*} f_i = \sum_{i \in I} f_i x_i^*,$$

то есть значение целевой функции задачи SC на построенном решении (x_i) равняется оптимальному значению целевой функции $f_0(X)$ задачи $MINF_0$. Кроме того, заметим, что если (x_i^*) – оптимальное решение задачи SC , то для множества $X = \{i \in I \mid x_i^* = 1\}$ справедливы равенства

$$f_0(X) = \sum_{i \in X} f_i + \sum_{j \in J} \min_{i \in X} \Phi(1 - a_{ij}) = \sum_{i \in X} f_i = \sum_{i \in I} f_i x_i^*.$$

Из сказанного в силу признака оптимальности получаем, что решение (x_i) задачи SC , построенное по оптимальному решению X^* задачи $MINF_0$ является оптимальным и, следовательно, задача SC сводится к задаче $MINF_0$.

Имеет место и обратная сводимость. Рассмотрим задачу $MINF_0$ с парой матриц (F_0, C) . Пусть $H=(h_{is})$ – характеристическая матрица размера $m \times S$ для матрицы C и пусть $\tilde{C}=(b_s h_{is})$ – каноническая форма матрицы C . Задаче $MINF_0$ с парой (F_0, C) поставим в соответствие задачу SC вида

$$\begin{aligned} \min_{(x_i), (v_s)} \{ & \sum_{i \in I} f_i x_i + \sum_{s=1}^S b_s v_s \}; \\ & \sum_{i \in I} (1 - h_{is}) x_i + v_s \geq 1, \quad s = 1, \dots, S; \\ & x_i, v_s \in \{0, 1\}, \quad i \in I, \quad s = 1, \dots, S. \end{aligned}$$

Оптимальному решению $((x_i^*), (v_s^*))$ данной задачи поставим в соответствие множество $X = \{i \in I \mid x_i^* = 1\}$ и покажем, что это оптимальное решение задачи $MINF_0$ с парой матриц (F_0, \tilde{C}) . Для этого, так же как и выше, воспользуемся критерием оптимальности. Заметим, что в силу оптимальности решения $((x_i^*), (v_s^*))$ имеем

$$v_s^* = 1 - \max_{i \in I} (1 - h_{is}) x_i^* = 1 - \max_{i \in X} (1 - h_{is}) = \min_{i \in X} h_{is}, \quad s = 1, \dots, S.$$

Отсюда получаем

$$f_0(X) = \sum_{i \in X} f_i + \sum_{s=1}^S \min_{i \in X} b_s h_{is} = \sum_{i \in I} f_i x_i^* + \sum_{s=1}^S b_s v_s^*,$$

то есть значение целевой функции $f_0(X)$ задачи $MINF_0$ на построенном решении X равняется оптимальному значению целевой функции задачи SC . Кроме того, для оптимального решения X^* задачи $MINF_0$ с парой (F_0, \tilde{C}) укажем допустимое решение $((x_i), (v_s))$ задачи SC , на котором значение целевой функции совпадает с величиной $f_0(X^*)$. Такое решение определяется следующим образом:

$$\begin{aligned} x_i &= \begin{cases} 1, & \text{если } i \in X^*, \\ 0, & \text{иначе.} \end{cases} \\ v_s &= \min_{i \in X^*} h_{is}. \end{aligned}$$

Это решение является допустимым поскольку

$$v_s = \min_{i \in X^*} h_{is} = 1 - \min_{i \in I} (1 - h_{is}) x_i \geq 1 - \sum_{i \in I} (1 - h_{is}) x_i, \quad s = 1, \dots, S.$$

Кроме того, значение целевой функции на данном решении по построению этого решения совпадает с величиной $f_0(X^*)$. Из сказанного, в силу критерия оптимальности (лемма 1.1), получаем, что решение X , построенное по оптимальному решению $((x_i^*), (v_s^*))$ задачи SC , является оптимальным решением задачи $MINF_0$ с парой

(F_0, \tilde{C}) и, следовательно, – задачи $MINF_0$ с исходной парой (F_0, C) . Таким образом, задача $MINF_0$ сводится к задаче SC и требуемая эквивалентность задач $MINF_0$ и SC показана.

Из теоремы 1.3 получаем, что задачи FL , $MINF_0$, $MINP_0$ и SC взаимосводимы. Отсюда вытекает, что эти задачи являются NP-трудными. Действительно, задача $SC1$, являясь частным случаем задачи SC , сводится к каждой из этих задач, а распознавательный вариант задачи $SC1$ является одной из наиболее известных задач класса **NPC** [40].

Из проводимых в ходе доказательства теоремы построений ясно, какой вид имеет задача SC , соответствующая задаче FL и, наоборот, как выглядит задача FL , соответствующая задаче SC . Если матрица C имеет характеристическую матрицу $H=(h_{is})$ размера $m \times S$ с весами столбцов b_s , $s=1, \dots, S$, то задача FL с исходными данными (F_0, C) сводится к задаче SC вида

$$\begin{aligned} \min_{(x_i), (v_s)} \{ & \sum_{i \in I} f_i x_i + \sum_{s=1}^S b_s v_s \}; \\ \sum_{i \in I} (1 - h_{is}) x_i + v_s & \geq 1, \quad s = 1, \dots, S; \\ x_i, v_s & \in \{0, 1\}, \quad i \in I, \quad s = 1, \dots, S. \end{aligned}$$

С другой стороны, задача SC с исходными данными (F_0, A) , где $A=(a_{ij})$ матрица ограничений размера $m \times n$ сводится к задаче FL с парой (F_0, C) , где $C=(c_{ij})$ матрица размера $m \times n$ с элементами вида $c_{ij} = \Phi(1 - a_{ij})$.

Отсюда видно, что если для характеристической матрицы $H=(h_{is})$ найдено полезное свойство, позволяющее за полиномиальное время решать задачу FL , то тем самым найден подкласс эффективно разрешимых задач SC . Этот подкласс определяется матрицами ограничений $A=(a_{ij})$ такими, что матрица $\bar{A}=(1 - a_{ij})$ обладает нужным свойством. При этом заметим, что поскольку число матриц A и \bar{A} в множестве матриц ограничений задачи SC одинаковое, как и число матриц H и \bar{H} , то относительная мощность построенного класса эффективно разрешимых задач SC такая же, как и относительная мощность эффективно разрешимых (по рассматриваемому свойству характеристической матрицы) задач FL . В этом смысле можно утверждать, что задача SC не сложнее задачи FL .

С другой стороны, при сведении задачи FL к задаче SC исходным данным (F_0, C) задачи FL соответствует матрица A ограничений задачи SC , устроенная следующим образом. Если $H=(h_{is})$ – характеристическая матрица размера $m \times S$ для матрицы C и $\bar{H}=(1 - h_{is})$, то матрица \tilde{A} имеет вид $\tilde{A} = \begin{pmatrix} \bar{H} \\ E \end{pmatrix}$, где E – единичная диагональная матрица размера $S \times S$. Отсюда видно. Что если для матрицы ограничений A найдено некоторое полезное свойство, позволяющее эффективно решать задачу SC , то это может не дать никакого результата применительно к задаче FL . Действительно, матрица \tilde{A} , уст-

роенная специальным образом, может не обладать требуемым полезным свойством, и, следовательно, класс характеристических матриц H , для которого матрица \tilde{A} обладает рассматриваемым полезным свойством, может оказаться вырожденным.

Таким образом, в отношениях задач FL и SC нет симметрии, наблюдаемой в отношениях задач FL , $MINF_0$ и $MINP_0$. Если выделен эффективно разрешимый класс задач FL , то тем самым найден эффективно разрешимый подкласс задачи SC той же относительной мощности. Но обратное не имеет места и, если найден эффективно разрешимый подкласс задачи SC , то не обязательно получен аналогичный результат для задачи FL . В этом смысле можно утверждать, что задача FL более трудная, чем задача SC .

4.2 Задача о вершинном покрытии

Выше показана взаимосводимость между задачами FL , $MINF_0$, $MINP_0$ и задачей SC , позволяющая достаточно просто строить по оптимальному решению одной задачи оптимальное решение другой. Аналогичная взаимосводимость имеет место между задачами FL , $MINF_0$, $MINP_0$ и частным случаем задачи SC – задачей VC .

Теорема 1.3 Задача $MINP_0$ сводится к задаче VC .

Доказательство. Задаче $MINP_0$ для полинома

$$p_0(y_1, \dots, y_m) = \sum_{i \in I} f_i(1 - y_i) + \sum_{s=1}^S b_s \prod_{i \in \beta_s} y_i$$

поставим в соответствие задачу VC вида

$$\min \left\{ \sum_{i \in I} f_i x_i + \sum_{s=1}^S b_s \left(\sum_{i \in \beta_s} t_{is} + |\beta_s| - 1 \right) \right\} \quad (1.4.1)$$

$$x_i + t_{is} \geq 1, \quad s = 1, \dots, S, \quad i \in \beta_s; \quad (1.4.2)$$

$$t_{is} + t_{ks} \geq 1, \quad s = 1, \dots, S, \quad i \in \beta_s, \quad k \in \beta_s, \quad i \neq k; \quad (1.4.3)$$

$$x_i, t_{is} \in \{0, 1\}, \quad i \in I, \quad s = 1, \dots, S. \quad (1.4.4)$$

Отметим, что эта задача действительно является задачей VC , в целевой функции которой для удобства последующих рассуждений присутствует константа, не влияющая на выбор оптимального решения. Поскольку по теореме 1.2 задача $MINP_0$ сводится к задаче SC вида

$$\min_{(x_i), (w_s)} \left\{ \sum_{i \in I} f_i x_i + \sum_{s=1}^S b_s w_s \right\};$$

$$\sum_{i \in \beta_s} x_i + w_s \geq 1, \quad s = 1, \dots, S;$$

$$x_i, w_s \in \{0, 1\}, \quad i \in I, \quad s = 1, \dots, S,$$

то для доказательства требуемой сводимости достаточно показать сводимость этой задачи к рассмотренной задаче VC .

Пусть $((x_i), (t_{is}))$ – допустимое решение задачи VC . Поставим ему в соответствие решение $((x_i), (w_s))$ задачи SC такое, что

$$w_s = \sum_{i \in \beta_s} t_{is} - |\beta_s| + 1. \quad (1.4.5)$$

Это допустимое решение. Действительно, по условию (1.4.3) величина $\sum_{i \in \beta_s} t_{is}$ равняется

либо $|\beta_s|$, либо $|\beta_s| - 1$. Следовательно, $w_s \in \{0, 1\}$. Кроме того, в силу условия (1.4.2) имеем

$$\sum_{i \in \beta_s} x_i + w_s = \sum_{i \in \beta_s} (x_i + t_{is}) - |\beta_s| + 1 = |\beta_s| - |\beta_s| + 1 = 1.$$

Допустимые решения $((x_i), (t_{is}))$ и $((x_i), (w_s))$ соответственно задач VC и CS , для которых выполняется равенство (1.4.5) назовем соответствующими. Легко видеть, что на соответствующих решениях значения целевых функций равны. Поэтому для завершения доказательства, согласно признаку оптимальности (лемма 1.1), остается показать, что по допустимому решению $((x_i), (w_s))$ задачи SC можно построить допустимое решение $((x_i), (t_{is}))$ задачи VC таким образом, чтобы эти решения были соответствующими, то есть, чтобы выполнялось равенство (1.4.5)

Пусть $((x_i), (w_s))$ – допустимое решение задачи SC . Заметим, что если $w_s = 0$, то множество $\{i \in \beta_s \mid x_i = 1\}$ не пусто. Поэтому если $w_s = 0$, то через $i(s)$ обозначим произвольно выбранный элемент указанного множества. Рассматриваемому решению $((x_i), (w_s))$ поставим в соответствие решение $((x_i), (t_{is}))$ такое, что

$$t_{is} = \begin{cases} 1, & \text{если } w_s = 1; \\ 1, & \text{если } w_s = 0 \text{ и } i \neq i(s); \\ 0, & \text{если } w_s = 0 \text{ и } i = i(s). \end{cases}$$

Это допустимое решение, поскольку если $t_{is} = 0$, то $x_i = 1$ и, следовательно, неравенство (1.4.2) выполняется. Неравенство (1.4.3) справедливо по определению величин t_{is} , $i \in I$. Заметим также, что непосредственно из определения величин t_{is} , $i \in I$, вытекает справедливость равенства

$$w_s = \sum_{i \in \beta_s} t_{is} - |\beta_s| + 1.$$

Таким образом, рассмотренные решения являются соответствующими, что завершает доказательство теоремы.

В ходе доказательства теоремы 3.1 получена задача VC , по оптимальному решению которой строится оптимальное решение задачи $MINP_0$. Учитывая приведенные в этом и предыдущем параграфах построения при доказательстве сводимости задач FL и SC к задаче $MINP_0$, построим задачи VC , соответствующие задачам FL и SC .

Пусть матрица C имеет характеристическую матрицу $H = (h_{is})$ размера $m \times S$ с весами столбцов b_s , $s = 1, \dots, S$. Тогда задача FL с исходными данными (F_0, C) сводится к задаче VC вида

$$\begin{aligned}
& \min_{(x_i), (t_{is})} \left\{ \sum_{i \in I} f_i x_i + \sum_{i \in I} \sum_{s=1}^S b_s (1 - h_{is}) t_{is} \right\}; \\
& x_i + t_{is} \geq 1 - h_{is}, \quad i \in I, s = 1, \dots, S; \\
& t_{is} + t_{ks} \geq (1 - h_{is})(1 - h_{ks}); \quad i, k \in I, i \neq k, s = 1, \dots, S; \\
& x_i, t_{is} \in \{0, 1\}, \quad i \in I, s = 1, \dots, S.
\end{aligned}$$

Аналогично, задача SC с матрицей ограничений $A=(a_{ij})$ размера $m \times n$ сводится к задаче VC вида

$$\begin{aligned}
& \min_{(x_i), (t_{ij})} \left\{ \sum_{i \in I} f_i x_i + \sum_{i \in I} \sum_{j=1}^n \Phi a_{ij} t_{ij} \right\}; \\
& x_i + t_{ij} \geq a_{ij}, \quad i \in I, j = 1, \dots, n; \\
& t_{ij} + t_{kj} \geq a_{ij} a_{kj}; \quad i, k \in I, i \neq k, j = 1, \dots, n.
\end{aligned}$$

Приведенные задачи действительно являются задачами VC , поскольку правые части ограничений этих задач равняются либо 0, либо 1.