В. Н. БУРКОВ, И. А. ГОРГИДЗЕ, С. Е. ЛОВЕЦКИЙ

ПРИКЛАДНЫЕ ЗАДАЧИ ТЕОРИИ ГРАФОВ

под редакцией А. Я. Горгидзе

ТБИЛИСИ
1974

5I8
5I8.4 УДК 62I.383.93
П 759

В книге рассматривается применение методов теории графов для решения различных задач исследования операций. Основная группа излагаемых задач имеет экономическую интерпретацию (задачи календарного планирования, распределение ресурсов на сетях, размещение складов, заводов, выбор маршрутов и др.). Книга предназначена для широкого круга читателей-инженеров, научных и руководящих работников, аспирантов и просто лиц, интересующихся прикладной математикой.

Б 20204
М-607 (03)-74

П Р Е Д И С Л О В И Е

Возьмите лист бумаги, нарисуйте несколько кружков и соедините их произвольным образом линиями со стрелками или без стрелок. Вы получите то, что называется геометрическим способом задания графа или просто графом. Итак, граф это любая система или структура, которая может быть абстрактно рассмотрена как множество кружков и множество соединяющих их линий. Переходя на язык теории графов, будем называть кружки - вершинами графа, линии со стрелками - дугами графа, а линии без стрелок - ребрами графа. Более строго, граф определен, если задано множество вершин и множество дуг или ребер, соединяющих пары вершин. Язык графов оказывается очень удобным при описании многих физических, биологических и технических систем и структур самого различного характера. Если вершины графа принять за города нашей страны, а дуги за связывающие их дороги, то получится карта дорог (шоссейных, железных). Примите вершины за узлы электрической цепи, дуги - за соединяющие эти узлы провода, и вы получите некоторую электрическую схему. Если вас интересует построение календарного плана работ, естественно, например принять работы за вершины сети и показать дугами зависимость между работами (технологию производства). Полученная технологическая схема очень наглядна и безусловно поможет вам в построении плана работ. Подобных примеров можно привести много. Задача книги и заключается, собственно говоря, в иллюстрации многообразия применений теории графов, как самостоятельной дисциплины используемой в качест-

ве метода решения различных задач математической экономики, исследования операций, теории игр, кибернетики, теории информации, проектировании информационных систем управления, социологии и других областях.

Ограниченный объем книги не позволяет дать полное изложение приложений графов в перечисленных областях.

Основная группа излагаемых задач имеет экономическую интерпретацию (задачи календарного планирования, распределения ресурсов на сетях, размещения складов, заводов, выбор маршрутов и др.). Это сделано не случайно. Подавляющее число приложений теории графов связано так или иначе с задачами экономического характера. С другой стороны, именно широкое применение методов теории графов при решении экономических и организационных задач способствовало бурному развитию самой теории за последние годы.

Выбор задач определялся существованием достаточно интересных приложений, причем не последнюю роль сыграли личные вкусы авторов.

Характер книги в значительной мере определил и стиль изложения. Авторы старались избегать излишней формализации выводов и доказательств, предпочитая "метод убеждения" и предоставляя читателям возможность самостоятельного творчества.

Читатели найдут в книге ряд оригинальных результатов, полученных авторами. Это относится к понятиям величины циркуляции и потенциала перестановок, результатам, связанным с задачами распределения ресурсов в сетях определенного вида и в сетях с упорядоченными событиями.

Необходимо отметить введение нового понятия псевдопотенциального графа и использовании его при решении задачи обработки партий деталей и задачи редактора, а также предложенные алгоритмы определения экстремальных путей. Стремясь сделать различные главы по возможности независимыми, авторы избрали следующий принцип изложения материала. В первой главе рассматриваются основные понятия теории графов, необходимые для понимания всех последующих глав. В каждой следующей главе приводятся необходимые сведения из теории графов, используемые при решении задач данной главы. Поскольку большинство из рассматриваемых в книге экстремальных задач носит комбинаторный характер, а для понимания алгоритмов решения некоторых из них требуется знание основ линейного программирования, в приложении к книге кратко изложены основные методы решения экстремальных комбинаторных задач и основы линейного программирования.

Книга "Прикладные задачи теории графов" предназначена для широкого круга читателей - инженеров, научных и руководящих работников, аспирантов и просто любителей прикладной математики.

Пользуемся случаем выразить искреннюю благодарность всем тем, кто своим вниманием помогли появлению этой книги, академику А.А.Воронову, профессорам А.Я.Горгидзе, Ю.Н.Иванову, Д.А.Квеселава.

Глава I

ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ
(основные понятия)

§ I. Определение графа

Рассмотрим некоторое множество X из n элементов. Природа этих элементов может быть самой различной (люди, города, страны, дома, книги, числа, функции, черты характера и т.д.). Пусть далее задано отображение Γ множества X в X элементы множества X будем изображать кружками, а пары элементов x_i и x_j соединять непрерывной линией, если $x_j \in \Gamma x_i \Leftrightarrow x_i \in \Gamma x_j$. Вид отображения естественно,зависит как от природы элементов, так и от характера решаемой задачи. Множество линий между элементами обозначим V . Пара множеств (X, V) и определяет граф. Элементы множества X называются вершинами графа, а элементы множества V- ребрами графа. Будем изображать ребра - отрезками линий, соединяющими соответствующие вершины. Если число элементов в X конечно, то граф называется конечным, в противном случае - бесконечным. Далее будем рассматривать только конечные графы. Из приведенного определения графа следует,что пары вершин могут соединяться не более, чем одним ребром.

Вершины графа будем нумеровать числами от I до n , а ребра обозначать $[i,j]$, где i и j номера соответствующих вершин (иногда для удобства будем обозначать ребра буквой с индексом внизу, например , V_1 , V_2 , V_j и т.д.).

Пример I.I. Иван, Лена и Володя знакомы друг с другом, Александр знаком с Иваном. Как отобразить эту ситуацию в виде графа? Присвоим Ивану, Лене, Володе и Александру соответственно номера I, 2, 3 и 4 . Будем считать, что два элемента связаны, если соответствующие лица знакомы. Имеем

$X = \{1, 2, 3, 4\}$, $V = \{[1, 2]$, $[1, 3]$, $[2, 3]$, $[4, 1]\}$

Граф (X, V) изображен на рис. I.I. Здесь имеется небольшая тонкость, считать ли человека знакомым с самим собой. Если да, то нужно нарисовать петлю у каждой вершины. В дальнейшем примем, что граф петель не имеет, если это не оговорено особо.

Часть графа, образованная подмножеством вершин вместе со всеми ребрами, соединяющими вершины этого подмножества, называется подграфом.

Если из графа удалить часть ребер, то получим частичный граф. В примере, когда вершины графа X соответствуют городам нашей страны, а ребра соответствуют железнодорожным и воздушным сообщениям между городами, то карта железных дорог, авиалиний Московской области – это подграф, а карта железных дорог нашей страны (при сохранении на ней вершин X) частичный граф.

§ 2. Смежные вершины

Вершины i , j называются смежными, если они соединены ребром. Смежные вершины называются также граничными вершинами соответствующего ребра, а это ребро называется инцидентным соответствующим вершинам. Будем обозначать

V_x - множество ребер, инцидентных вершине x, и d_x - число ребер, индидентных вершине x. Число d_x называется степенью вершины x. Очевидно, $d_x \leq n-1$ Граф, степени всех вершин которого равны $n-1$, называется полным.

Упражнение I.I. Докажите тождество

$$\sum_{i=1}^{n} d_i = 2m, \qquad (I.I)$$

где m - число ребер графа.

Вершина X графа называется висячей, если существует только одно инцидентное ей ребро, то есть $d_x = I$ (вершина 4 на рис.I.I).

Вершина X называется изолированной, если не существует инцидентных ей ребер, то есть $d_x = 0$.

Упражнение I.2. Докажите тождество

$$\sum_{i:n_i>0} i n_i = 2m, \qquad (I.2)$$

где n_i - число вершин графа, имеющих степень i. Действительно, для графа рис.I.I. имеем $n_1 = I$, $n_3 = I$, $n_2 = 2$ $m = 4$,

$$\sum_i i n_i = I \cdot I + 2 \cdot 2 + 3 \cdot I = 8 = 2m .$$

Зададимся вопросом для каких наборов степеней $(d_1, d_2, \ldots, d_n)$ существуют соответствующие графы? Пусть d_i упорядочены таким образом, что $d_1 \leq d_2 \leq \ldots \leq d_n$.

Заметим сначала, что если $d_1 = d_2 = \ldots = d_n = S$, то есть степени всех вершин равны (такие графы называются однородными), то при выполнении условия $nS = 2m$ всегда можно построить соответствующий граф (убедитесь

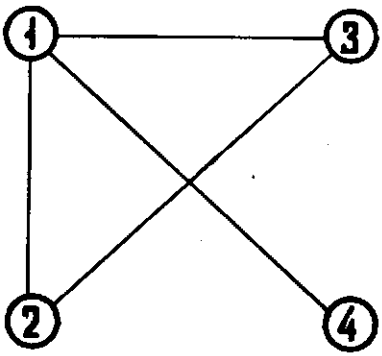


Рис. 1.1

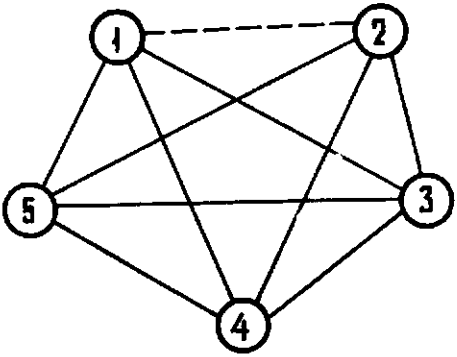


Рис. 1.2

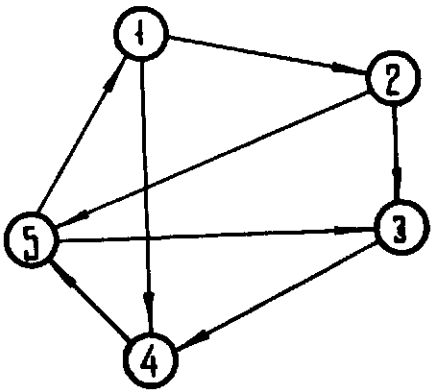


Рис. 1.3

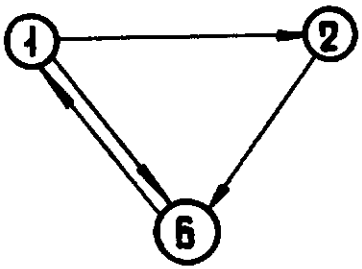


Рис. 1.4

в этом самостоятельно). А теперь докажите более сложный результат: для любых положительных n_s и n_{s+1} удовлетворяющих условиям

$$Sn_s + (S+1)n_{s+1} = 2m, \quad n_s + n_{s+1} = n \qquad (I.3)$$

существует соответствующий граф. Для доказательства сначала постройте однородный граф степени S (либо $S+1$) и затем добавляя (удаляя) ребра, получите требуемый.

Пример I.2. Пусть $m = 9$, $n = 5$, $S = 3$.

Из уравнений (I.3) получаем

$$n_4 = 2m - nS = 3,$$

$$n_3 = n - n_{s+1} = 2.$$

Построим однородный граф степени 4 (рис.I.2). Удалив теперь любое ребро, получим требуемый граф.

Для ответа на вопрос существования графа со степенями $d_1, d_2, \ldots, d_n$, удовлетворяющими (I.I.), опишем процедуру выравнивания степеней. Описание проведем на примере. Пусть задан следующий набор степеней ($n = 8$, $m = 18$)

i	1	2	3	4	5	6	7	8
d_i	2	3	3	4	5	5	7	7

Исключаем d_1 и вычитаем по единице из d_7 и d_8 (всего вычитаем $d_1 = 2$ единицы).

i	2	3	4	5	6	7	8
d_i'	3	3	4	5	5	6	6

Исключаем d_2 и вычитаем по единице из d_6, d_7, d_8 (всего вычитаем $d_2 = 3$ единицы).

i	3	4	5	6	7	8
d_i^2	3	4	5	4	5	5

Исключаем d_3 и вычитаем по единице из d_5, d_7 и d_8 (всего вычитаем $d_3 = 3$ единицы).

i	4	5	6	7	8
d_i^3	4	4	4	4	4

Получили набор степеней, отличающихся не более чем на I. Следовательно, граф существует, и его легко получить, построив сначала граф со степенями, полученными на последнем шаге процедуры, и добавив затем исключенные степени.

Упражнение I.3. Существует ли граф со следующими степенями

i	I	2	3	4	5	6	7
d_i	3	3	5	5	6	6	6

§ 3. Цепи. Циклы

Ц е п ь - это последовательность ребер (V_1, V_2, ...), в которой у каждого ребра V_k $(k \neq 1)$ одна из граничных вершин является граничной для ребра V_{k-1}, а другая - граничной для ребра V_{k+1}. Цепь называется простой, если все ее ребра различные, и составной - в противном случае. Цепь будем также обозначать номерами последовательных вершин цепи, так цепь (I,2,3,4) состоит из ребер (I,2),(2,3) и (3,4). Цепь называется элементарной, если все ее вершины различны.

Ц и к л - это конечная цепь, начинающаяся в некоторой вершине X и оканчивающаяся в той же вершине. Цикл называется простым, если все его ребра различны, и составным - в противном случае. Цикл, при обходе которого ни одна вершина не встречается дважды, называется элементарным. Элементарная цепь, проходящая через все вершины графа, на-

зывается гамильтоновой цепью, а элементарный цикл, проходящий через все вершины графа - гамильтоновым циклом.

§ 4. Связные графы

Если любые две вершины графа можно соединить цепью, то граф называется связным. Если граф не является связным, то его можно разбить на связные подграфы. Такие связные части графа называются компонентами. Проверить связность графа просто. Нужно пометить произвольную вершину графа и затем пометить все вершины, смежные с уже помеченными и т.д. Если таким путем будут помечены все вершины, то граф связный. В противном случае множество помеченных вершин определяет компоненту графа.

Задача I.4. Докажите, что описанная в п.I.2 процедура выравнивания степеней графа решает также вопрос о существовании связного графа со степенями $d_1, d_2, \ldots, d_n$, если $m \geq n-1$.

Связностью ν графа называется минимальное число ребер, после удаления которых граф становится несвязным. Связность графа не может быть больше, чем $\left[\frac{2m}{n}\right]$ ($[X]$ целая часть числа X). Действительно, в графе всегда найдется вершина со степенью меньше или равной $\left[\frac{2m}{n}\right]$. Удалив инцидентные ей ребра, получим несвязный граф. Можно доказать, что существуют графы с n вершинами и m ребрами, имеющие связность $\left[\frac{2m}{n}\right]$.

§ 5. Ориентированные графы

Во многих задачах значение имеет не только факт соединения элементов множества X линиями, но и направление

этих линий. Ребро, соответствующее упорядоченной паре вершин i,j называется дугой (i,j) и изображается линией со стрелкой, направленной от i к j .

Граф, определяемый множеством X вершин и множеством U дуг, называется ориентированным. Однако, для краткости слово "ориентированный" в дальнейшем будем опускать. При этом будем иметь в виду, что если речь идет о дугах, то граф ориентированный, а если о ребрах, то граф неориентированный. Дуги будем обозначать также буквой U , возможно, с индексами (u_1 , u_2 , u_j и т.д.). Вершины i и j называются граничными вершинами дуги (i , j), причем i - начало дуги, j - конец дуги. Две вершины называются смежными, если они различны и существует дуга, соединяющая эти вершины.

Дуга U исходит из вершины X , если X является началом дуги, дуга U заходит в X , если X является концом дуги. В обоих случаях дуга U называется инцидентной вершине X . Число дуг, исходящих из вершины X , называется полустепенью исхода этой вершины и обозначается d_x^+ . Число дуг, заходящих в вершину X , называется полустепенью захода вершины X и обозначается d_x^- . Так, для графа, показанного на рис.I.3

$$d_1^+ = 2, \quad d_2^+ = 2, \quad d_3^+ = 1, \quad d_4^+ = 1, \quad d_5^+ = 2,$$
$$d_1^- = 1, \quad d_2^- = 1, \quad d_3^- = 2, \quad d_4^- = 2, \quad d_5^- = 2.$$

В дальнейшем, если это не будет оговорено особо, рассматриваются графы без петель, то есть без дуг вида (i , i).

В этом случае

$$\sum_{i=1}^{n} d_i^+ = \sum_{i=1}^{n} d_i^- = m \qquad (I.4)$$

Граф, между любыми двумя вершинами которого существует хотя бы одна дуга, называется полным.

Если из $(i,j) \in U$ следует $(j,i) \in U$, то граф называется симметрическим.

Если из $(i,j) \in U$ следует, что $(j,i) \notin U$, то граф называется антисимметрическим.

§ 6. Пути. Контуры

П у т ё м в графе (X, U) называется такая последовательность $(u_1, u_2 \ldots)$ дуг, что конец каждой предыдущей дуги совпадает с началом следующей. Путь является простым, если все его дуги различны, и составным – в противном случае. Путь, состоящий из последовательности вершин $x_1, x_2, \ldots, x_q$, будем обозначать символом $\mu = (x_1, x_2, \ldots x_q)$. Путь, в котором все вершины различны, называется элементарным.

К о н т у р – это конечный путь, у которого начальная вершина совпадает с конечной. Контур называется элементарным, если все его вершины различны.

Элементарный путь, проходящий через все вершины графа, называется гамильтоновым путем. Элементарный контур, проходящий через все вершины графа, называется гамильтоновым контуром.

Длиной пути называется число дуг пути. Соответственно длина контура равна числу дуг контура.

Длина гамильтонова пути равна (n - I). а длина гамильтонова контура равна n . Во многих задачах в соответствие каждой дуге (i , j) ориентированного графа ставят некоторое число l_{ij} , называемое длиной дуги. В этих случаях под длиной пути (контура) понимается сумма длин дуг этого пути. Эти два понятия длин пути и контура совпадают, если l_{ij} = I для всех дуг. В случае неориентированного графа соответственно имеем длину цепи и цикла.

В графе, показанном на рис.I.3:

(5, I, 2, 5, I, 4) - составной путь,

(5, I, 2, 5, 3) - простой путь,

(5, I, 2, 3) - элементарный путь,

(5, I, 2, 5) - элементарный контур,

(I, 2, 3, 4, 5) - гамильтонов путь,

(I, 2, 3, 4, 5, I) - гамильтонов контур.

§ 7. Сильно связные графы

Граф, для любых двух вершин i , j которого существует путь из вершины i в вершину j , называется сильно связным.

Заметим, что граф, имеющий гамильтонов контур, сильно связный. Опишем простой способ проверки сильной связности графа.

Берем произвольную вершину в качестве начальной и двигаемся по дугам графа, отмечая встречающиеся вершины. Возможны два случая:

а) дальнейшее движение невозможно (попали в вершину, из которой нет выхода). В этом случае граф не является сильно связным;

б) попадаем в уже отмеченную вершину. Соответствующая замкнутая часть пути определяет контур. Стягиваем вершины контура в одну.

Обозначим X_μ -множество вершин выделенного контура μ, U_μ — множество дуг подграфа, образованного множеством X_μ, U_μ —вершину, соответствующую контуру μ. Дуги множества U_μ при стягивании контура μ в вершину X_{n+1} удаляются. Дуга (i, X_{n+1}) проводится в том и только в том случае, если существует хотя бы одна дуга (i, j) где $j \in \mu$, $i \bar{\in} \mu$. Дуга (X_{n+1}, i) проводится в том и только в том случае, если существует хотя бы одна дуга (j, i), где $j \in \mu$, $i \bar{\in} \mu$. Повторяем процедуру. Если в результате граф стягивается в одну вершину, то он является сильно связным.

Применим описанный алгоритм для проверки сильной связности графа рис.I.3. Возьмем вершину I за начальную. Двигаясь по дугам выписываем путь (I,4,5,3,4, ...). Вершина 4 встретилась дважды. Выделяем контур (4,5,3,4) и стягиваем его в одну вершину 6 (рис.I.4). Повторяя процедуру, находим контур (I,2,6,I), содержащий все вершины нового графа. Следовательно, граф рис.I.3 сильно связен.

§ 8. Матрица смежности графа

Определим квадратную матрицу $n \times n$ следующим образом. На пересечении i-ой строки и j-го столбца запишем I, если $[i,j] \in V$ и 0, если $[i,j] \bar{\in} V$. Определенная таким образом матрица называется матрицей смежности графа. Заметим, что матрица смежности всегда симметрическая. Аналогично можно определить матрицу смежности для ориентированного графа. Эта матрица уже не является симметрической в общем случае. Для графа рис.I.3, например, матрица смежности имеет вид

$$A=\begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Обозначим $A^{\lambda}=A\cdot A\cdots A$ λ-ю степень матрицы A, т.е. матрицу A^{λ} полученную в результате булева произведения λ матриц A. Приведем без доказательства следующий результат.

Теорема I.I. Элемент p_{ij}, стоящий на пересечении i-й строки и j-го столбца матрицы A^{λ}, равен числу различных путей длины λ, идущих из i в j. Элемент p_{ii} матрицы A^{λ} равен числу различных контуров длины λ, содержащих вершину i.

Для графа рис.I.3 имеем

$$A^2=\begin{pmatrix} 0 & 0 & 1 & 0 & 2 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 \end{pmatrix} \qquad A^3=\begin{pmatrix} 2 & 0 & 2 & 1 & 0 \\ 0 & 1 & 0 & 2 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 3 \end{pmatrix}$$

Элемент p_{55} матрицы A^3 равен 3. Действительно, имеются три контура $[5, 1, 4, 5]$, $[5, 1, 2, 5]$ и $[5, 3, 4, 5]$, длины 3, содержащие вершину 5. Элемент p_{15} матрицы A^2 равен 2. Проверьте, что существуют два пути длины 2 из вершины I в вершину 5.

§ 9. Матрицы инциденций графа

Пусть $v_1, v_2, \ldots, v_m$ - ребра графа.

Определим матрицу R с n строками и m столбцами следующим образом. Элемент r_{ij} этой матрицы равен I, если вершина i инцидентна ребру V_j и равен 0, в противном случае $(i=1,2,\ldots,n;\ j=1,2,\ldots,m)$.

Полученная матрица называется матрицей инциденций для ребер графа. Так если граф рис.I.3 считать неориентированным, то матрица инциденций его ребер $V_1 = (1,2)$, $V_2 = (1,4)$, $V_3 = (2,3)$, $V_4 = (2,5)$, $V_5 = (3,4)$, $V_6 = (4,5)$, $V_7 = (5,1)$, $V_8 = (5,3)$ имеет вид

$$R = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Аналогично определяется матрица инциденций для дуг графа. Элемент r_{ij} этой матрицы равен +I, если дуга U_j исходит из вершины i, - I, если дуга U_j заходит в вершину i и равен 0, в остальных случаях. Для графа рис.I.3 матрица инциденций дуг имеет вид (номера дуг совпадают с номерами соответствующих ребер)

$$R = \begin{pmatrix} +1 & +1 & 0 & 0 & 0 & 0 & -1 & 0 \\ -1 & 0 & +1 & +1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & +1 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 & -1 & +1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & +1 & +1 \end{pmatrix}$$

Каждый столбец матрицы инциденций для ребер графа со-

держит ровно 2 единицы, а число единиц строки равно степени соответствующей вершины. В свою очередь каждый столбец матрицы инциденций для дуг графа содержит ровно одну (+I) и одну (-I). Число положительных единиц строки равно степени исхода соответствующей вершины, а число отрицательных - степени ее захода.

§ IO. Деревья

Д е р е в о м называется связный граф без простых циклов, имеющий не менее двух вершин. Так как дерево не имеет простых циклов, то у него обязательно должны быть висячие вершины. Если удалить висячую вершину вместе с инцидентным ей ребром, то оставшийся граф также будет деревом (если $n > 2$). Продолжая таким образом, получим простейшее дерево, состоящее из двух вершин и соединяющего их ребра. Отсюда следует, что число вершин дерева на единицу больше числа его ребер, то есть

$$m = n - 1. \qquad (I.5)$$

Подставляя (I.5) в тождество (I.2), получаем

$$\sum_i i n_i - 2(n-1) = 2\sum_i n_i - 2,$$

или

$$n_1 = 2 + \sum_{i>2} (i-2) n_i . \qquad (I.6)$$

Формула (I.6) позволяет определить число висячих вершин дерева. Так, например, для дерева рис.I.5 имеем

Поэтому

$$n_1 = 2 + n_3 + 2n_4 = 5 .$$

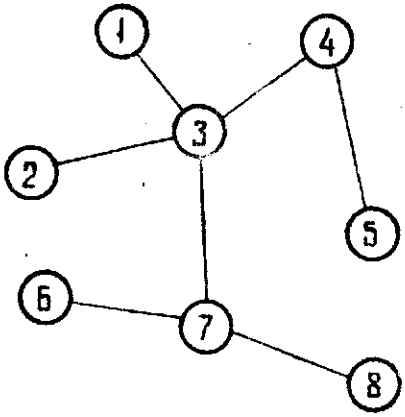


Рис 1.5.

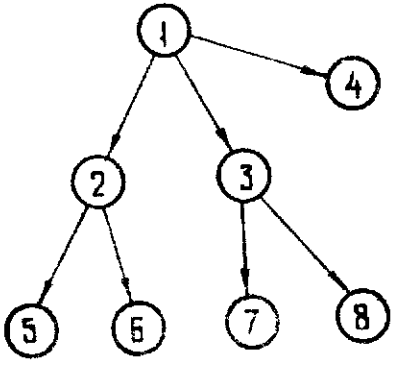
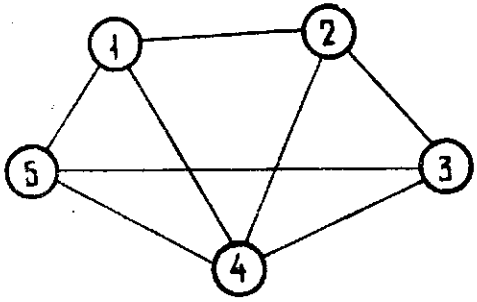
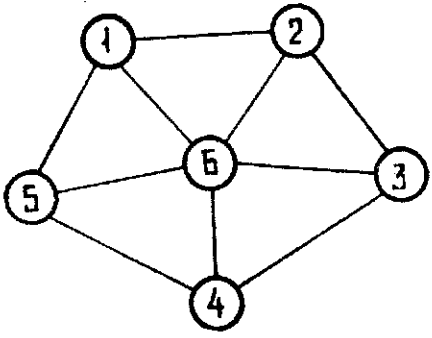


Рис. 1.6.



а



б

Рис 1.7.

Упражнение I.4. Докажите, что для любого набора положительных чисел ($n_1, n_2, \ldots$), удовлетворяющего (I.6), существует дерево.

Для ориентированного графа часто используется понятие прадерева. Прадерево - это ориентированное дерево, у которого одна из вершин, называемая корнем, не имеет заходящих дуг, а степени захода остальных вершин равны I. Пример прадерева приведен на рис.I.6.

§ II. Плоские графы

Граф называется плоским, если его можно изобразить на плоскости так, что различным вершинам соответствуют различные кружки и никакие два ребра не имеют общих точек, отличных от их границ. Изображение графа на плоскости, отвечающее этим условиям, называется плоским топологическим графом. Оба графа на рис.I.7 являются плоскими, хотя граф рис.I.7а не является плоским топологическим графом.

Гранью плоского топологического графа называется область плоскости, ограниченная ребрами и не содержащая внутри себя ни вершин, ни ребер. Особо выделяется внешняя грань (грань, ограниченная ребрами (I,2), (2,3), (3,4), (4,5), (5,I) для графа рис.I.7б). Например, дерево (заметим, что дерево всегда плоский граф) имеет всего одну внешнюю грань - всю плоскость. Будем обозначать грани буквой τ , возможно, с индексами, например, τ_1 , τ_2 , ..., τ_k и т.д.

Край грани - это цикл, образованный граничными ребрами грани. Две грани называются смежными, если их края имеют хотя бы одно общее ребро. Обозначим p - число граней

плоского графа, p_i – число его граней, имеющих степень i (<u>степенью грани</u> называется число ее граничных ребер, причем ребра, граничные только для данной грани, считаются дважды). Для любого плоского топологического связного графа, имеющего n вершин, m ребер и ρ граней

$$n - m + \rho = 2 \quad \text{(формула Эйлера)}. \qquad (I.7)$$

Действительно, для дерева формула очевидна, так как $n = m + 1$, а $\rho = 1$. Однако, любой связный граф можно свести к дереву, удаляя ребра, не нарушающие связности. Удаление одного такого ребра уменьшает m на единицу и ρ на единицу. Это доказывает справедливость формулы. Следовательно, верны соотношения

$$\sum_i i n_i = 2m; \quad \sum_i i p_i = 2m; \quad \sum_i p_i + \sum_i n_i = m + 2. \qquad (I.8)$$

Эти соотношения выражают необходимые условия того, что набору чисел $\{n_i\}$ и $\{p_i\}$ соответствует плоский граф. Однако, эти условия недостаточны, что видно из следующего примера.

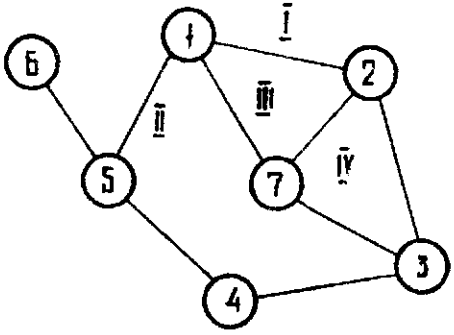
<u>Пример I.3.</u> Существует ли плоский граф, у которого все грани степени 3, $n = 13$ и степень каждой вершины больше или равна 5?

Имеем $3p_3 = 2m$, $p_3 = \frac{2m}{3}$.

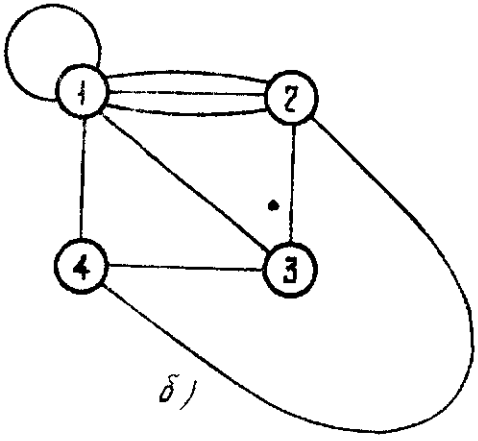
Подставляя в формулу Эйлера, получаем

$$\frac{2m}{3} + 13 = m + 2; \qquad m = 33.$$

Единственное решение системы (I.8) имеет вид $n_6 = 1$, $n_5 = 12$. Тем не менее такого плоского графа не существует



а)



б)

Рис. 1.8

тересно отметить, что плоские графы с треугольными гранями и степенями вершин не менее 5 существуют для любого числа вершин $n \geq 12$, исключая $n = 13$ (не зря 13 считается плохим числом!).

Любому связному плоскому топологическому графу G можно поставить в соответствие связный плоский топологический граф G^*, получаемый следующим образом:

а) каждой грани графа G соответствует вершина графа G^*.

б) каждому ребру V графа G, являющемуся граничным для граней z_1 и z_2, соответствует ребро V^* графа G^*, соединяющее соответствующие граням z_1 и z_2 вершины. На рис.I.8б приведен граф G^* для графа G на рис.I.8а (номера вершин графа G^* совпадают с номерами граней графа G, показанными римскими цифрами).

Заметим, что граф G^* имеет петли, если граф G имеет висячие вершины. Кроме того, в графе G^* две вершины могут соединяться более чем одним ребром (такие графы называются мультиграфами).

Граф G^* называется двойственным для графа G. Понятие двойственного графа тесно связано с понятием двойственности в линейном программировании.

Глава П.

ЭКСТРЕМАЛЬНЫЕ ПУТИ И КОНТУРЫ НА ГРАФАХ

Задачи нахождения кратчайших и длиннейших путей в графах возникают в различных прикладных областях науки и техники, включая расчет временных параметров в системах сетевого планирования и управления (СПУ), поиск оптимальных маршрутов в транспортных сетях и сетях связи, при раскладке и монтаже электрических цепей сетей, вычислительных машин и пр.

При поиске экстремальных путей в графе (X, U) с $(n+1)$ вершинами предполагается, что каждому ребру $(i,j) \in U$ отнесено рациональное число l_{ij} или "длина" дуги (длина дуги может быть и отрицательной). Как уже отмечалось выше под длиной пути понимается сумма длин входящих в него дуг. В этом случае возникают задачи о кратчайших (длиннейших) путях в (X, U):

(I) Определение кратчайшего пути из заданной вершины скажем 0 до другой заданной вершины, скажем n;

(2) Определение длиннейшего пути от вершины 0 до вершины n;

Предлагаемые далее алгоритмы позволяют находить также кратчайшие (длиннейшие) пути между любыми парами вершин в графе (X, U).

§ I. Формулировка прикладных задач

Многие прикладные задачи на оптимум могут быть сведены к поиску экстремальных путей на графах определенного вида. Рассмотрим некоторые из этих задач, которые сформу-

лируем как задачи о поиске экстремальных путей (кратчайших и длиннейших путей, путей максимальной эффективности).

Задача о ранце

Предположим существует n предметов, j -ый предмет имеет вес a_j и ценность p_j (ценность положительна) Требуется найти наиболее ценное подмножество предметов, при условии, что их общий вес не должен превышать вместимости ранца b . Задача может быть, естественно, сформулирована как задача целочисленного линейного программирования в виде

$$\sum_j p_j x_j \to max$$

при условии

$$\sum_j a_j x_j \leq b ,$$

$x_j = 1$, если предмет j выбран,

$x_j = 0$, в противном случае.

К задаче о ранце сводятся многие интересные проблемы выбора при ограничениях на общий вес, площадь, общие средства и т.д. [2]

Такого сорта задачи приходится, например, решать при распределении бюджета организации по ряду заключаемых контрактов. После некоторых преобразований задача сводится к поиску ожидаемой максимальной прибыли организации, когда p_j - ожидаемый средний доход, получаемый от заключения j -ого контракта, a_j - средства отпускаемые под j -ый контракт и b - средства, которыми располагает организация.

Задача формулируется как задача поиска длиннейшего пути в соответствующей сети.

Пусть сеть имеет вершины обозначенные упорядоченными парами (j,k), $j=0,1,\ldots,n$, $k=0,1,2,\ldots,b$.

В вершину (j,k) входят две дуги - одна из $(j-1,k)$ другая из $(j-1,k-q)$, тем самым определена вершину (j,k). Длина первой дуги равна нулю, а второй D_j Вводится также начальная вершина, которая соединяется со всеми вершинами $(0,k)$ дугами нулевой длины. Тогда пути от начальной вершины к (j,k) соответствуют подмножествам первых j объектов, чей общий вес самое больше k, длина пути соответствует ценности подмножества.

Необходимо отметить, что сеть для задачи о ранце не содержит контуров, как это видно на примере показанном на рис.2.I. $(n=4, b=4)$

Задача коммивояжера

Задача коммивояжера для данного полного симметрического ориентированного графа (X, U) состоит в поиске кратчайшего Гамильтонного контура,(т.е. контура проходящего через каждую из вершин только один раз), когда известны длины дуг $l_{ij} \geqslant 0$. Таким образом интерпретация задачи коммивояжера в терминах экстремальных путей вполне очевидна.

Предположим, что мы заменили некоторую вершину графа, скажем вершину n двумя новыми вершинами p и q, где p соединяется со всеми вершинами инцидентными с вершиной n дугами направленными от p и q соединяется с теми же вершинами дугами направленными к q. Тогда задача

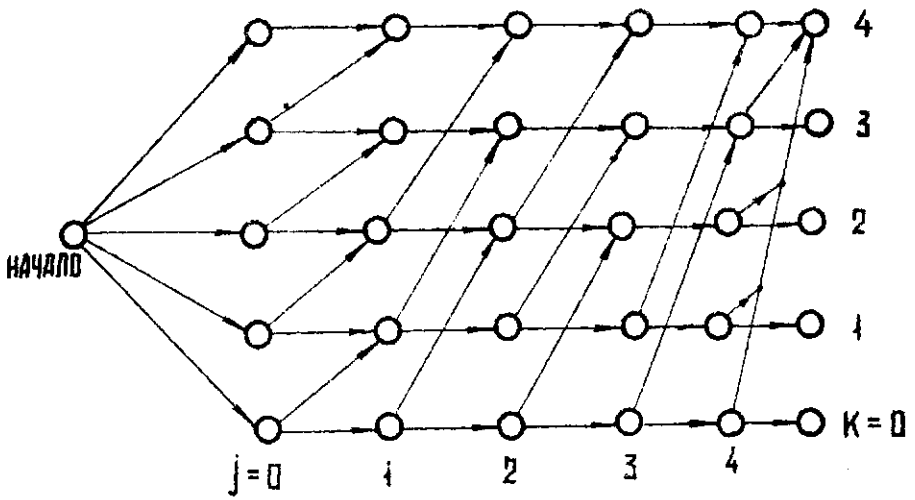


Рис. 2.1

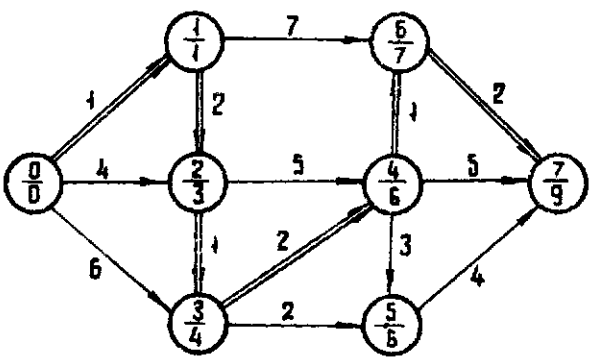


Рис. 2.2

коммивояжера состоит в нахождении кратчайшего пути из p в q, при условии, что путь проходит через каждую из вершин 0, I, 2, ..., $n-1$ только один раз.

Теперь предположим, что мы заменили l_{ij} на длину $K - l_{ij}$, где K – достаточно большое положительное число. Теперь задача состоит в нахождении длиннейшего пути от p до q, при условии, что вершина не повторяется, т.е. путь элементарный.

Задача нахождения длиннейшего элементарного пути может быть решена различными способами.

Путь максимальной надежности

В сетях связи, обычно определяется вероятность p_{ij} того, что связь от i к j существует. Следовательно, вероятность того, что связь осуществлена по любому данному пути есть произведение вероятностей дуг пути. Требуется найти путь максимальной надежности.

Задача сводится к поиску кратчайшего пути в сети, если заменить вероятности p_{ij} "длинами" $l_{ij} = -\log p_{ij}$ Наиболее надежный путь тогда будет соответствовать кратчайшему пути.

Путь максимальной эффективности

В графе (X, U) с $n+1$ вершинами каждой дуге (i,j) отнесены два числа продолжительность дуги t_{ij} и эффективность S_{ij}. Под эффективностью любого пути,

соединяющего вход 0 с выходом n будем понимать сумму эффективностей дуг пути, под продолжительностью, как и раньше - сумму продолжительностей дуг пути под средней эффективностью - отношение эффективности пути к его продолжительности. Возникают задачи поиска пути максимальной средней эффективности, пути максимальной эффективности при дополнительных ограничениях на продолжительности. Прежде чем переходить к описанию алгоритмов поиска экстремальных путей исследуем важный вопрос о разрешимости задачи, под которой понимается существование кратчайшего пути конечной длины. Будем предполагать, что для любой вершины $i \neq 0$ и n существует хотя бы один путь из 0 в n, проходящий через нее (в противном случае эту вершину можно удалить из графа без ущерба для задачи).

Теорема 2.I. Для разрешимости задачи о кратчайшем пути необходимо и достаточно, чтобы в графе отсутствовали контуры отрицательной длины.

Необходимость. Если в графе имеется контур отрицательной длины, то кратчайший путь будет иметь длину ($-\infty$) и будет включать бесконечное число обходов контура отрицательной длины.

Достаточность. Если все контуры графа имеют неотрицательную длину, то существует кратчайший путь, который является элементарным. Так как каждый элементарный путь состоит из конечного числа дуг, длины которых конечны, то и кратчайший элементарный путь имеет конечную длину.

Предполагая в дальнейшем условия теоремы 2.I. выполненными, рассмотрим основные алгоритмы определения кратчайшего пути.

§ 2. Пути минимальной длины

Алгоритм Форда

I. Помечаем каждую вершину i индексом λ_i ; первоначально берем

$$\lambda_0 = 0, \quad \lambda_i = +\infty, \quad i \neq 0$$

2. Ищем такую дугу (i , j), для которой $\lambda_j - \lambda_i > \ell_{ij}$ затем заменяем индекс λ_j индексом $\lambda_j' = \lambda_i + \ell_{ij} < \lambda_j$. Продолжаем таким образом до тех пор, пока остается хотя бы одна вершина i , для которой можно уменьшить λ_i .

3. После того как индексы установятся, находим такую вершину i_1 , что $\lambda_n - \lambda_{i_1} = \ell_{i_1 n}$. Далее, находим такую вершину i_2 , что $\lambda_{i_1} - \lambda_{i_2} = \ell_{i_2 i_1}$ и т.д. Если все контуры графа положительной длины, то последовательность $\lambda_{i_s}, \lambda_{i_{s-1}}, \ldots, \lambda_{i_2}, \lambda_{i_1}, \lambda_n$ будет обязательно элементарным путем (докажите самостоятельно) и значит в некоторый момент будет $\lambda_{i_s} = \lambda_0$. Путь $(0, i_s, i_{s-1}, \ldots, i_2, i_1, n)$ длины λ_n и будет искомым кратчайшим путем. Процедура определения кратчайшего пути сложнее, если граф имеет контуры нулевой длины, так как в этом случае кратчайший путь не обязательно должен быть элементарным.

Если граф является сетью (то есть не имеет контуров), алгоритм существенно упрощается. Перенумеруем вершины таким образом, что не существует пути, идущего из вершины с большим номером в вершину с меньшим номером (такая нумерация вершин сети называется правильной).

Полагаем последовательно

$$\lambda_0 = 0$$
$$\lambda_1 = l_{01}$$
$$\lambda_2 = \min [\lambda_0 + l_{02}, \lambda_1 + l_{12}],$$
$$\vdots$$
$$\lambda_j = \min_{i<j} [\lambda_i + l_{ij}],$$
$$\vdots$$
$$\lambda_n = \min_{0 \le i < n} [\lambda_i + l_{ij}],$$

где $l_{ij} = +\infty$, если $(i,j) \notin U$. Значение λ_n равно длине кратчайшего пути.

Пример 2.I. На рис.2.2. приведен граф с правильной нумерацией вершин. Числа, поставленные у дуг, равны их длинам. Для удобства вычислений каждый кружок разделен на две части. В верхней записан номер вершины, а в нижней - ее индекс.

Имеем последовательно

$$\lambda_0 = 0,$$
$$\lambda_1 = l_{01},$$
$$\lambda_2 = \min (0+4, 1+2) = 3,$$
$$\lambda_3 = \min (0+6, 3+1) = 4,$$
$$\lambda_4 = \min (3+5, 4+2) = 6,$$
$$\lambda_5 = \min (4+2, 6+3) = 6,$$
$$\lambda_6 = \min (6+1, 1+7) = 7,$$
$$\lambda_7 = \min (7+2, 6+5, 6+4) = 9.$$

Двигаясь от конца к началу, определяем дуги, входящие в

кратчайший путь, как дуги, для которых $\lambda_j - \lambda_i = l_{ij}$. Кратчайший путь выделен на рис.2.2 двойными линиями. Алгоритм Форда, удобный в машинной реализации, не совсем удобен при ручных вычислениях, если граф не является сетью или если нумерация вершин не является правильной.

Алгоритм Данцига

Будем предполагать, что длины всех дуг неотрицательны.

I. Помечаем вершину 0 индексом $\lambda_0 = 0$.

2. Пусть уже помечено некоторое множество вершин. Обозначим Q множество непомеченных вершин, смежных с помеченными. Для каждой вершины $i \in Q$ вычисляем величину $\mu_i = \min(\lambda_i + l_{ij})$, где минимум берется по всем помеченным вершинам j , смежным с i . Помечаем вершину i , для которой μ_i минимальна, индексом $\lambda_i = \mu_i$. Подобную процедуру повторяем пока не будет помечена вершина n .

3. Длина кратчайшего пути равна индексу λ_n , а сам путь определяется аналогично тому,как это делается в предыдущем алгоритме.

Упражнение 2.I. Докажите, что описанный алгоритм действительно определяет кратчайший путь, если длины всех дуг неотрицательны.

Пример 2.2. Граф приведен на рис.2.3.

I шаг $\lambda_0 = 0$, $Q = \{1,2,3\}$. Вычисляем

$\mu_1 = l_{01} = 1$, $\mu_2 = 4$, $\mu_3 = 5$

Помечаем вершину I индексом $\lambda_1 = 1$.

2 шаг $Q = \{2,3,4\}$, $\mu_2 = 4$, $\mu_3 = 5$, $\mu_4 = 2$

Помечаем вершину 4 индексом $\lambda_4 = 2$.

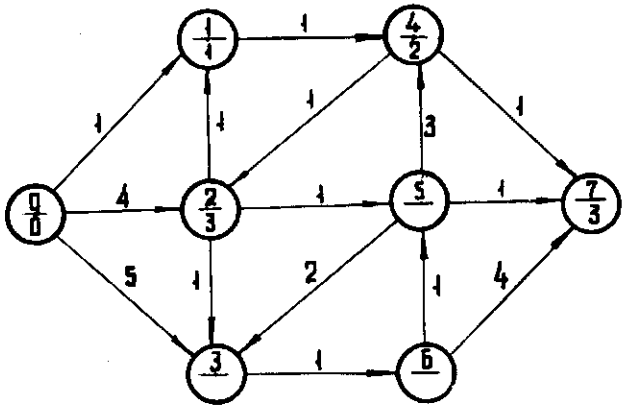


Рис. 2.3

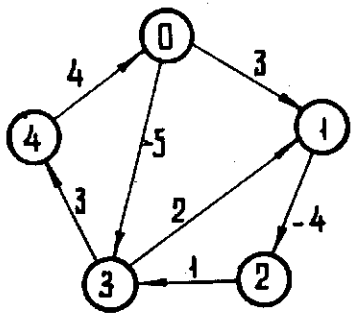


Рис. 2.4

3 шаг $Q = \{2,3,7\}$, $\mu_2 = 3$, $\mu_3 = 5$, $\mu_7 = 3$

Помечаем вершину 7 индексом $\lambda_7 = 3$.

Поскольку вершина 7 помечена, то кратчайший путь найден. Это путь (0,I,4,7) длины 3. Заметим, что в данном случае нам не потребовалось даже рассматривать все вершины графа.

§ 3. Контур минимальной длины

По-прежнему предполагаем, что в графе отсутствуют контуры отрицательной длины. Если известно, что искомый контур содержит вершину i, то задача решается просто. Нужно определить кратчайший путь от вершины i к вершине i, применив описанные выше алгоритмы. Поскольку в общем случае контур минимальной длины может проходить через любую вершину графа, то находятся контуры минимальной длины, проходящие через каждую вершину, и среди них выбирается кратчайший.

§ 4. Контур отрицательной длины

Как мы видели выше, существенным требованием к алгоритмам поиска кратчайших путей и контуров является отсутствие в графе контуров отрицательной длины. Если же такие контуры есть, то алгоритм Данцига вообще неприменим, а алгоритм Форда никогда не закончится. Как же убедиться, что в графе имеются контуры отрицательной длины?

Простейший способ продолжать изменять индексы, пока число шагов алгоритма не превысит максимально необходимое для установления всех индексов. Оценим это максимально необходимое число шагов. Для этого заметим, что при каждом просмотре m дуг графа, хотя бы одна вершина получит

окончательный индекс (то есть индекс, который больше не будет меняться). Поскольку всего вершин n (не считая начальной), то потребуется самое большое $m \cdot n$ шагов для установления всех индексов. Поэтому если число шагов алгоритма превысило $m \cdot n$, то в графе, наверняка, имеется контур отрицательной длины. К сожалению, этот способ при большом числе вершин и дуг графа не очень экономен. Возможен другой путь. Ограничим потенциал λ_i в каждой вершине числом d_i. Если на некотором шаге λ_i оказалось меньше d_i, то проверяем, действительно ли полученное значение λ_i соответствует длине некоторого пути, или в данном случае имеется контур отрицательной длины. Такую проверку легко осуществить, двигаясь от вершины, для которой нарушилось условие $\lambda_i \geq d_i$ в обратном направлении (то есть в направлении, обратном ориентации дуг), причем только по дугам, удовлетворяющим условию $\lambda_j - \lambda_i = l_{ij}$; при этом запрещается проходить дважды по одной и той же дуге.

Такое движение обязательно приведет в вершину 0, если λ_i соответствует длине простого пути из 0 в i. В противном случае будет обнаружен контур. Выбор чисел d_i во многом будет определять эффективность обнаружения контуров отрицательной длины. Действительно, если взять d_i маленькими, то придется сделать много шагов алгоритма прежде чем нарушится условие $\lambda_i \geq d_i$. Если же взять d_i большими, то это условие будет нарушаться слишком часто, и потребуется много проверок на наличие контура.

Пример 2.3. Применим описанный подход для определения контуров отрицательной длины в графе, показанном на рис.2.4. Примем все $d_i = 5$, $i = 1, 2, 3, 4$.

I. Полагаем $\lambda_0 = 0$, $\lambda_i = +\infty$, $i = 1,2,3,4$,

для дуги (0,3) имеем $\lambda_0 - 5 = -5 < +\infty$, $\lambda_3 = -5 = d_3$

-"- (3,1) -"- $\lambda_3 + 2 = -3 < +\infty$, $\lambda_1 = -3 > d_1$

-"- (3,4) -"- $\lambda_3 + 3 = -2 < +\infty$, $\lambda_4 = -2 > d_4$

-"- (1,2) -"- $\lambda_1 - 4 = -7 < +\infty$, $\lambda_2 = -7 < d_2$

Условие $\lambda_2 \geq d_2$ нарушилось. Однако, двигаясь в обратном направлении, определяем последовательность $2 \to 1 \to 3 \to 0$, соответствующую элементарному пути (0,3,1,2) длины -7. Поэтому полагаем $d_2 = 27$. Продолжаем дальше. Для дуги (2,3) имеем $\lambda_2 + 1 = -6 < -5$, $\lambda_3 = -6 < d_3$. Теперь нарушилось условие $\lambda_3 \geq d_3$. Двигаясь в обратном направлении, получаем последовательность $3 \to 2 \to 1 \to 3 \to$, в которой вершина 3 встречается дважды. Следовательно, контур (3,1,2,3) имеет отрицательную длину.

Можно предложить другой способ определения контура отрицательной длины. Он основан на следующей идее. Пусть мы уже убедились, что не существует контура отрицательной длины в подграфе с вершинами $0,1,2,\ldots, i$. Для проверки существования контура отрицательной длины в подграфе $0,1,2,\ldots,i,i+1$ достаточно найти контур кратчайшей длины, проходящий через вершину $(i+1)$, применяя алгоритм Форда. Начиная с $i = 2$, повторяем эту процедуру и либо находим контур отрицательной длины, либо убеждаемся в его отсутствии (при этом, мы определяем контур минимальной неотрицательной длины). Так для предыдущего примера подграфы с вершинами $\{0,1\}$, $\{0,1,2\}$ не имеют контуров отрицательной длины. В подграфе, образованном вершинами $\{0, 1, 2, 3\}$, контур минимальной длины $\{3, 1, 2, 3\}$ имеет отрицательную длину.

§ 5. Пути максимальной длины

Итак, мы умеем определять пути минимальной длины. Изменим теперь знаки длин дуг на противоположные и определим кратчайший путь в полученном графе. Очевидно, этот путь будет иметь максимальную длину в первоначальном графе. Для того, чтобы такой путь существовал, необходимо и достаточно отсутствие в графе контуров положительной длины (что соответствует отсутствию контуров отрицательной длины в графе с измененными знаками длин всех дуг). Можно не менять знаки длин дуг, а немного изменить сами алгоритмы. Так в алгоритме Форда начальные индексы следует взять $\lambda_0 = 0$, $\lambda_i = -\infty$, $i = 1,2,\ldots, n$. Индексы вершины изменяются, если найдется дуга (i, j) такая, что $\lambda_j - \lambda_i < l_{ij}$. Новый индекс в этом случае $\lambda_j = \lambda_i + l_{ij}$ (напоминаем, что при определении кратчайшего пути индекс менялся всякий раз, когда нарушалось условие $\lambda_j - \lambda_i > l_{ij}$).

Упражнение 2.3. Как будет выглядеть алгоритм Данцига при определении путей максимальной длины.

Заметим, что если в графе имеется хотя бы один контур ненулевой длины, то не существуют либо пути минимальной длины (если контур имеет отрицательную длину), либо пути максимальной длины (если контур имеет положительную длину). В графе существуют и кратчайшие, и длиннейшие пути только в том случае, если граф не имеет контуров, либо все контура имеют нулевую длину.

Пример 2.4. Определим длиннейший путь в графе, показанном на рис.2.2, применяя алгоритм Форда. Имеем последовательно

$$\lambda_0 = 0,$$
$$\lambda_1 = 1,$$
$$\lambda_2 = \max(4, 1+2) = 4,$$
$$\lambda_3 = \max(6, 4+1) = 6,$$
$$\lambda_4 = \max(4+5;\ 6+2) = 9,$$
$$\lambda_5 = \max(6+2;\ 9+3) = 12,$$
$$\lambda_6 = \max(1+7;\ 9+1) = 10,$$
$$\lambda_7 = \max(9+5;\ 12+4;\ 10+2) = 16.$$

Длина максимального пути равна 16, а соответствующий путь (0,2,4,5,7). Длиннейший путь в графе без контуров получил название критического пути (это понятие имеет принципиальное значение в теории сетевого планирования и управления). Действительно, если дуги графа соответствуют работам некоторого проекта, а сам граф отражает технологическую схему выполнения работ, то длина критического пути определяет минимальный срок осуществления проекта.

Если в графе имеются контуры положительной длины, то конечного пути максимальной длины не существует. Способы обнаружения контуров положительной длины аналогичны способам обнаружения контуров отрицательной длины с соответствующими изменениями.

§ 6. Контур минимальной средней длины

Средней длиной дуг контура называется отношение длины контура к числу его дуг. Опишем алгоритм определения контура с минимальной средней длиной дуг, существенно использующий алгоритм определения контуров отрицательной длины.

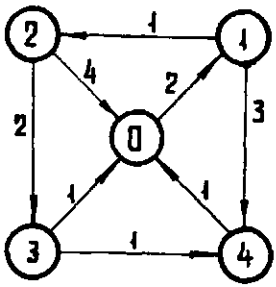
1. Вычисляем $l_0 = \max\limits_{i,j} l_{ij}$.

и полагаем длины всех дуг равными

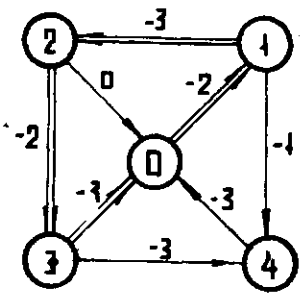
$$l'_{ij} = l_{ij} - l_0$$

2. Определяем контур отрицательной длины. Пусть $\mathcal{L}$ длина этого контура, κ - число его дуг. Вычисляем $l_{ср} = \mathcal{L}/\kappa$ и добавляем ($-l_{ср}$) к длинам l'_{ij} всех дуг. Затем снова определяем контур отрицательной длины и т.д. до тех пор, пока на очередном шаге таких контуров не найдется. Так как на каждом шаге алгоритма длины всех дуг изменялись на одно и то же число, то на последнем шаге длина каждой дуги равна $l_{ij} - \gamma$, где γ суммарное изменение длины каждой дуги на всех шагах. Значение γ равно минимальной средней длине дуг контуров графа. При этом контуром минимальной средней длины дуг является контур, определенный на предпоследнем шаге.

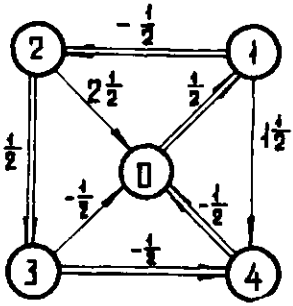
Пример 2.5. Определим контур C минимальной средней длиной дуг в графе, показанном на рис.2.5а, $l_0 = 4$. Вычитая 4 из всех длин дуг, получаем граф с длинами дуг, показанный на рис.2.5б. Определяем контур отрицательной длины (0,I,2,3,0). Длина этого контура равна (-I0), число дуг - 4. Добавляем к длине каждой дуги по $\frac{I0}{4} = 2{,}5$. Получаем граф, показанный на рис.2.5в. В этом графе имеется единственный контур (0,I,2,3,4,0) отрицательной длины $\mathcal{L} = -\frac{I}{2}$ и числом дуг = 5. Проверьте самостоятельно, что при увеличении длин всех дуг графа рис.2.5в на $\frac{I}{I0}$, контура полученного графа будут иметь положительную длину, а контур (0,I,2,3,4,0) будет иметь нулевую длину. Следовательно, контур (0,I,2,3,4,0) имеет минимальную среднюю длину дуг, которая равна I,4.


2
1
0
3
4
1
4
2
2
3
1
1
1

а


2
1
0
3
4
-3
0
-2
-2
-1
-3
-3
-3

б


2
1
0
3
4
$-\frac{1}{2}$
$2\frac{1}{2}$
$\frac{1}{2}$
$\frac{1}{2}$
$1\frac{1}{2}$
$-\frac{1}{2}$
$-\frac{1}{2}$
$-\frac{1}{2}$

в

Рис. 2.5

§ 7. Задачи о потенциалах вершин графа

С задачами о кратчайших (длиннейших) путях и контурах тесно связаны двойственные им задачи о потенциалах вершин графа. Действительно, как мы видели в предыдущих пунктах, определение кратчайших путей эквивалентно определению некоторых чисел λ_i (потенциалов вершин), удовлетворяющих условиям $\lambda_j - \lambda_i \le l_{ij}$ для любой дуги (i, j) графа. Можно поставить самостоятельную задачу о потенциалах вершин графа.

Задача 2.2. Определить потенциалы вершин графа, удовлетворяющие условиям $\lambda_i \ (i = 0, 1, \dots, n)$

$$\lambda_j - \lambda_i \le l_{ij}, \quad (i, j) \in U \qquad (2.I)$$

и минимизирующие некоторую функцию

$$\Phi(\lambda_0, \lambda_1, \dots, \lambda_n)$$

Нетрудно показать, что для разрешимости системы (2.I) необходимо и достаточно, чтобы в графе отсутствовали контуры отрицательной длины. Действительно, пусть $(i_1, i_2, \dots, i_s, i_1)$ контур, имеющий длину $\mathcal{L} < 0$. Выпишем последовательно систему неравенств

$$\begin{aligned}
&\lambda_{i_2} - \lambda_{i_1} \le l_{i_1 i_2} \\
&\lambda_{i_3} - \lambda_{i_2} \le l_{i_2 i_3} \\
&\vdots \\
&\lambda_{i_s} - \lambda_{i_{s-1}} \le l_{i_{s-1} i_s} \\
&\lambda_{i_1} - \lambda_{i_s} \le l_{i_s i_1}
\end{aligned}$$

Складывая все эти неравенства, получаем противоречие

$$\lambda_{i_1} - \lambda_{i_1} = 0 \le l_{i_1 i_2} + l_{i_2 i_3} + \dots + l_{i_s i_1} = \mathcal{L} < 0$$

§ 8. Задача о ближайших потенциалах

Пусть G граф с n вершинами и длинами дуг l_{ij}

($i, j = 1, 2, \ldots, n$). Пусть далее задан некоторый набор $\{\lambda_i^\circ\}$, $i = 1, 2, \ldots, n$ потенциалов вершин графа. Требуется определить потенциалы λ_i, $i = 1, 2, \ldots, n$, наиболее близкие к набору $\{\lambda_i^\circ\}$. Близость будем оценивать величиной наибольшей разности $|\lambda_i - \lambda_i^\circ|$, то есть

$$\Phi = \max_i |\lambda_i - \lambda_i^\circ|. \qquad (2.2)$$

Алгоритм. Для решения задачи применяем алгоритм Форда, начиная со значений индексов λ_i°. Когда индексы установятся (если в графе отсутствуют контуры отрицательной длины), вычисляем $\max_i |\lambda_i - \lambda_i^\circ|$ и увеличиваем все λ_i на половину полученного числа.

Упражнение 2.6. Докажите, что алгоритм дает оптимальное решение задачи о ближайших потенциалах.

Пример 2.6. Найдем потенциалы вершин графа, показанного на рис.2.6, ближайшие к набору $\lambda_1^\circ = 1, \lambda_2^\circ = 5, \lambda_3^\circ = 10, \lambda_4^\circ = 4, \lambda_5^\circ = 2$. Установившиеся значения индексов в результате применения алгоритма Форда приведены в нижних половинах вершин.

Вычисляем

$$\max(1-1;\ 5-3;\ 10-6;\ 4-4;\ 2-2) = 4.$$

Оптимальные значения потенциалов вершин $\lambda_1 = 3, \lambda_2 = 5, \lambda_3 = 8, \lambda_4 = 6, \lambda_5 = 4$

При этом $\Phi = \max_i |\lambda_i - \lambda_i^\circ| = 2$.

Упражнение 2.7. Решите задачу о ближайших потенциалах с критериями

$$\Phi = \sum_{i=1}^{n} |\lambda_i - \lambda_i^\circ|$$

$$\Phi = \sum_{i=1}^{n} (\lambda_i - \lambda_i^\circ)^2$$

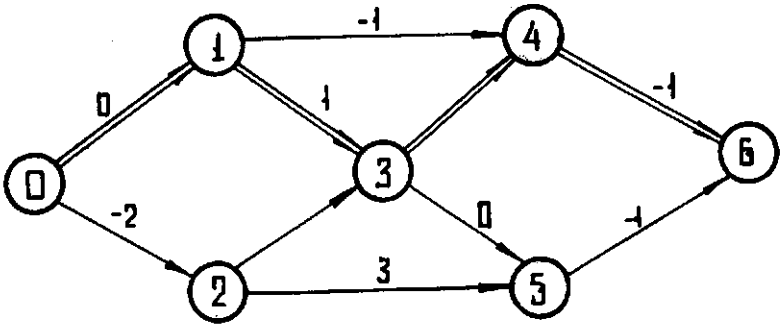


Рис. 2.7

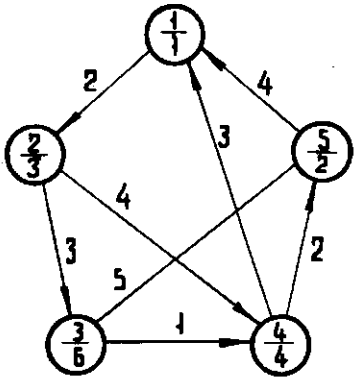


Рис. 2.6

§ 9. Определение пути максимальной средней эффективности

Рассмотрим сеть с $(n+1)$ вершинами. Отнесем каждой дуге сети два числа продолжительность дуги t_{ij} и эффективность S_{ij}. Под эффективностью любого пути, соединяющего вход 0 с выходом n будем понимать сумму эффективностей дуг пути, под продолжительностью - сумму продолжительностей дуг пути, под средней эффективностью - отношение эффективности пути к его продолжительности.

Задача I. Определить путь максимальной средней эффективности.

Задача II. Определить путь максимальной эффективности при условии, что продолжительность пути не превышает заданной величины T.

Задача I. решается аналогично задаче определения контура минимальной средней длины. Пусть λ - максимальная средняя эффективность. Тогда для любого пути μ

$$\frac{\sum\limits_{(i,j)\in\mu} S_{ij}}{\sum\limits_{(i,j)\in\mu} t_{ij}} \leq \lambda$$

или

$$\sum_{(i,j)\in\mu} (S_{ij} - \lambda t_{ij}) \leq 0$$

Следовательно, для решения задачи достаточно определить λ, при которой длина максимального пути в сети (при длинах дуг $\ell_{ij} = S_{ij} - \lambda t_{ij}$) будет равна 0. Отсюда

сразу следует алгоритм решения задачи I.

I. Полагаем длины l_{ij} дуг равными $l_{ij} = S_{ij}$ и определяем длиннейший путь μ_0. Вычисляем

$$S_0(\mu_0) = \sum_{(i,j)\in\mu_0} S_{ij},$$

$$T_0(\mu_0) = \sum_{(i,j)\in\mu_0} t_{ij},$$

$$\lambda_0(\mu_0) = \frac{S_0(\mu_0)}{T_0(\mu_0)}$$

II. Полагаем длины дуг $l_{ij} = S_{ij} - \lambda_0(\mu_0)\, t_{ij}$ и снова определяем длиннейший путь μ. Вычисляем

$$\lambda_1(\mu_1) = \frac{S(\mu_1)}{T(\mu_1)}$$

Если $\lambda(\mu_1) = \lambda(\mu_0)$, то путь μ определяет оптимальное решение задачи. Если $\lambda(\mu_1) > \lambda(\mu_0)$, то полагаем длины дуг равными $l_{ij} = S_{ij} - \lambda(\mu_1)\, t_{ij}$ и повторяем процедуру. За конченое число шагов будет получен длиннейший путь μ_p, такой, что $\lambda(\mu_p) = \lambda(\mu_{p-1})$ который и определит оптимальное решение.

Пример 2.7. Определим путь максимальной средней эффективности для сети, изображенной на рис.2.7 с данными, приведенными ниже.

(i,j)	(01)	(02)	(13)	(14)	(23)	(25)	(34)	(35)	(46)	(56)
S_{ij}	2	4	5	7	3	9	6	4	I	3
t_{ij}	2	6	4	8	2	6	5	4	2	4

I. Проверьте самостоятельно, что при $l_{ij} = S_{ij}$ путь

μ_0 = (0,2,5,6) имеет максимальную длину $S(\mu_0) = 16$. Вычисляем $T(\mu_0) = t_{02} + t_{25} + t_{56} = 16$.

$$\lambda(\mu_0) = 1.$$

II. Полагаем $l_{ij} = S_{ij} - t_{ij}, \ (i,j) \in U$

и снова определяем путь максимальной длины (этот путь μ_1 = (0,1,3,4,6) выделен двойными дугами на рис.2.7 . Вычисляем

$$S(\mu_1) = S_{01} + S_{13} + S_{34} + S_{46} = 14$$
$$T(\mu_1) = t_{01} + t_{13} + t_{34} + t_{46} = 13,$$
$$\lambda(\mu_1) = \frac{14}{13}.$$

при длинах дуг $l_{ij} = S_{ij} - \frac{14}{13} t_{ij}$ путь μ_1 остается путем максимальной длины. Следовательно, максимальная средняя эффективность равна $1\frac{1}{13}$ и обеспечивается путем μ_1 .

§ 10. Определение пути с максимальной эффективностью

Задача II является более сложной, чем задача I . Применим для ее решения метод "ветвей и границ" (см.приложение). Способ разбиения на подмножества определим самым естественным в данном случае образом. Обозначим $Q(i_1, i_2, \ldots, i_s)$ - подмножество путей, соединяющих начальную вершину с конечной которые имеют общую часть $(i_1, i_2, \ldots, i_s, \ldots)$. Подмножество $Q(i_1, i_2, \ldots, i_s)$ разбивается на подмножества $Q(i_1, i_2, \ldots, i_s, j)$, где $(i_s, j) \in U$. Так,например, подмножество $Q(1,3)$ в сети, показанной на рис.2.7, разбивается на два подмножества $Q(1,3,4)$ и $Q(1,3,5)$. Для оценки подмножества $Q(i_1, i_2, \ldots, i_s)$

достаточно определить оценку сверху для максимальной эффективности пути из вершины i_s в конечную вершину, при условии, что продолжительность пути не превышает

$$T - \sum_{k=1}^{s} t_{i_{k-1} i_k}$$

Опишем способ оценки сверху максимальной эффективности для произвольной сети (а значит, и для произвольного подмножества решений).

Обозначим $x_{ij} = 1$, если дуга (i,j) принадлежит пути и $x_{ij} = 0$ в противном случае. Тогда задачу П можно сформулировать следующим образом: определить $x_{ij}, (i,j) \in U$ максимизирующие

$$\sum_{i,j} s_{ij} x_{ij} \qquad (2.3)$$

при ограничениях

$$\sum_{i} x_{0i} = \sum_{i} x_{in} = 1$$
$$\sum_{j} x_{ij} = \sum_{k} x_{ki}, \quad i = 1,2,\ldots,n-1, \qquad (2.4)$$
$$\sum_{ij} x_{ij} t_{ij} \leqslant T$$

Заменяя дискретное условие $x_{ij} = 0$ или 1 ограничениями $0 \leq x_{ij} \leq 1$, $(i,j) \in U$, перейдем к двойственной задаче. Для этого введем двойственные переменные u_i, $i = 0,1,\ldots,n$, $\lambda \geq 0$ и $w_{ij} \geq 0$, $(ij) \in U$. Двойственная задача запишется следующим образом: определить u_i, λ и w_{ij}, минимизирующие

$$u_n - u_0 + \lambda T + \sum_{(ij) \in U} w_{ij} \qquad (2.5)$$

при ограничениях

$$u_j - u_i + \lambda t_{ij} + w_{ij} \geq s_{ij}, \quad (i,j) \in U \qquad (2.6)$$

Заметим сначала, что существует оптимальное решение задачи (2.5), (2.6), в котором все $w_{ij} = 0$

Предположим, что λ задано. Тогда задача сводится к минимизации

$$u_n - u_0$$

При ограничениях

$$u_j - u_i \geq s_{ij} - \lambda t_{ij} = l_{ij}$$

Нетрудно заметить, что мы получили задачу определения пути максимальной длины в сети с длинами дуг l_{ij}. Обозначим $M(\lambda)$ длину максимального пути как функцию λ. Задача свелась к определению λ, минимизирующего

Так как $M(\lambda) + \lambda T$

$$M(\lambda) = \max_{\mu} \sum_{(i,j)\in\mu} (s_{ij} - \lambda t_{ij})$$

(μ - произвольный путь в сети), то $M(\lambda)$ - выпуклая (книзу) функция λ. Следовательно, $M(\lambda) + T\lambda$ также выпуклая (книзу) функция λ. Обозначим $T(\lambda)$ продолжительность максимального пути в зависимости от λ. Очевидно, $T(\lambda)$, кусочно-постоянная, невозрастающая функция λ. Условия оптимальности можно записать теперь в следующем виде

$$T(\lambda - 0) \geq T \geq T(\lambda + 0) \qquad (2.7)$$

Пусть λ - оптимальное решение, μ_1 - путь максимальной длины, имеющий продолжительность $T_1 = T(\lambda - 0)$, μ_2 - путь максимальной длины, имеющий продолжительность $T_2 = T(\lambda + 0)$, S_1 и S_2 соответственно эффективности этих путей.

Тогда оценка сверху максимальной эффективности равна

$$\hat{S} = \frac{T - T_2}{T_1 - T_2} S_1 + \frac{T_1 - T}{T_1 - T_2} S_2 \qquad (2.8)$$

Пример 2.8. Получим верхнюю оценку максимальной эффективности для сети из примера 2.7 при T = I5. На рис.2.8 показана сеть с длинами дуг $l_{ij} = S_{ij} - \lambda t_{ij}$

Вычисляя потенциалы U_i вершин сети, будем также определять значение λ_i, при котором происходит изменение наклона $M(\lambda)$

$$U_1 = 2 - 2\lambda$$
$$U_2 = 4 - 6\lambda$$
$$U_3 = \max(7-6\lambda;\ 7-8\lambda) = 7-6\lambda$$
$$U_4 = \max(9-10\lambda;\ 13-11\lambda) = 13-11\lambda, \text{ при } \lambda_4 \le 4$$
$$U_5 = \max(11-10\lambda;\ 13-12\lambda) = 13-12\lambda, \text{ при } \lambda_5 \le 1$$
$$U_6 = \max(14-13\lambda;\ 16-16\lambda) = 16-16\lambda, \text{ при } \lambda_6 \le \tfrac{2}{3}.$$

При $\lambda = \frac{2}{3}$ функция $M(\lambda) = U_6$ изменяет наклон. Поэтому первый линейный участок функции $M(\lambda)$ соответствует $0 \le \lambda \le \frac{2}{3}$. При этом максимальный путь μ_1 =(0,2,5,6), его продолжительность $T(\lambda)$ = I6, эффективность $S(\lambda)$ =I6

При $\lambda = \frac{2}{3}$ появляется новый максимальный путь μ_2 = (0,I,3,4,6), продолжительности $T = (\frac{2}{3} + 0) = 13$ и эффективности $S(\frac{2}{3} + 0) = 14$.

Так как

$$16 = T(\tfrac{2}{3} - 0) > T > T(\tfrac{2}{3} + 0) = 13,$$

то $\lambda = \frac{2}{3}$ определяет оптимальное решение двойственной задачи. Имеем T_1 = I6, S_1 = I6, T_2 = I3, S_2 = I4, и верхняя оценка максимальной эффективности равна

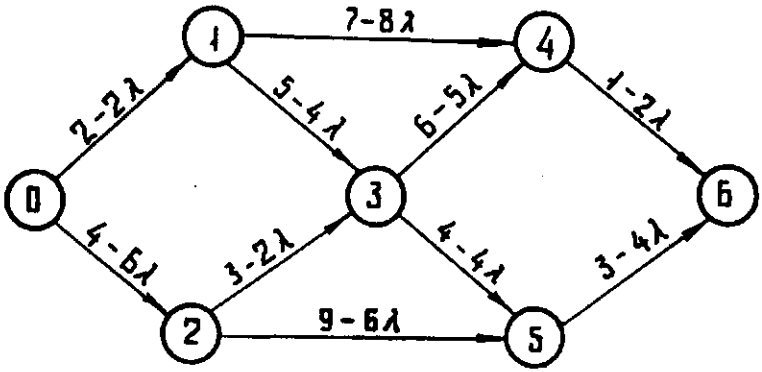


Рис. 28

$$\hat{S} = \frac{15-13}{16-13}\cdot 16 + \frac{16-15}{16-13}\cdot 14 = \frac{2\cdot 16 + 1\cdot 14}{3} = 15\frac{1}{3}.$$

Процедуру решения задачи можно существенно ускорить, если учесть ряд дополнительных условий. Рассмотрим одно из них. Если все пути, проходящие через данную дугу, имеют продолжительность больше T, то очевидно, эту дугу можно удалить из сети. Для определения таких дуг вычислим продолжительности A_i кратчайших путей, соединяющих начальную вершину с вершиной i (для всех $i = 0, 1, 2, \ldots, n-1$) и продолжительности B_i кратчайших путей, соединяющих вершину i с конечной вершиной (для всех $i = 1, 2, \ldots, n$). Если $A_i + t_{ij} + B_j > T$ то дугу (i, j) можно удалить из сети. Так, для сети из примера 2.7 имеем

i	0	I	2	3	4	5	6
A_i	0	2	6	7	I0	I0	I2
B_i	I2	I0	9	7	2	4	0

Для дуги (2,5)

$$A_2 + t_{25} + B_5 = 16 > 15$$

Поэтому дугу (2,5) можно удалить из сети. Определим теперь путь максимальной эффективности. Таких путей два: (0,2,3,4,6) и (0,2,3,5,6). Из этих путей путь (0,2,3,4,6) имеет минимальную продолжительность, равную I5. Следовательно, путь (0,2,3,4,6) определяет оптимальное решение задачи с эффективность I4.

§ II. Определение пути максимальной эффективности с учетом штрафов

Рассмотрим еще одну задачу определения пути с максималь-

ной эффективностью. Пусть продолжительность пути жестко не ограничена, но выплачивается определенный штраф при отклонении продолжительности пути от заданной T. Примем, что функция штрафа F имеет следующий вид

$$F(\mu) = \begin{cases} p[T(\mu) - T], & \text{при } T \geq T(\mu) \\ q[T(\mu) - T], & \text{при } T \leq T(\mu) \end{cases}$$

где p и q заданные коэффициенты.

Задача Ш. Определить путь μ, имеющий максимальное значение разности эффективности и штрафа, то есть величины

$$S(\mu) - F(\mu)$$

Пусть μ_1 – путь максимальной длины при длинах дуг $l_{ij} = s_{ij} - p t_{ij}$, μ_2 – путь максимальной длины при длинах дуг $l_{ij} = s_{ij} - q t_{ij}$.

При $p \geq q$ задача Ш решается элементарно. В этом случае продолжительность $T(\mu_1)$ пути μ_1 не превышает продолжительности $T(\mu_2)$ пути μ_2, то есть

$$T(\mu_1) \leq T(\mu_2)$$

Возможны три случая:

а) Если $T(\mu_2) \geq T(\mu_1) \geq T$, то путь μ_2 определяет оптимальное решение задачи.

б) Если $T(\mu_1) \leq T(\mu_2) \leq T$, то путь μ_1 определяет оптимальное решение задачи.

в) Если $T(\mu_1) < T < T(\mu_2)$, то оптимальное решение определяется тем из путей μ_1 и μ_2, который обеспечивает большее значение $S(\mu) - F(\mu)$

Доказательство утверждений (а), (б) и (в) предлагаем

провести самостоятельно.

Если $p < q$, то очевидно

$$T(\mu_1) \geq T(\mu_2)$$

В следующих двух случаях задача легко решается:

а) Если $T(\mu_1) \geq T(\mu_2) \geq T$, то путь μ_2 определяет оптимальное решение задачи.

б) Если $T(\mu_2) \leq T(\mu_1) \leq T$, то путь μ_1 определяет оптимальное решение задачи.

Доказательство этих двух утверждений также предоставляем читателю.

Итак, нерассмотренным остался случай

$$T(\mu_2) < T < T(\mu_1), \quad (p < q) \qquad (2.9)$$

Здесь уже не удается получить решение столь эффективно. Поэтому воспользуемся методом ветвей и границ, как это было сделано в предыдущем пункте. Покажем, что выражение (2.8) является оценкой сверху величины $S(\mu) - F(\mu)$ и для задачи Ш в условиях (2.9). Для этого разобьем задачу Ш на две:

А. Максимизировать $S(\mu) - F(\mu)$ при условии $T(\mu) \leq T$

Б. Максимизировать $S(\mu) - F(\mu)$ при условии $T(\mu) \geq T$

Задача А полностью эквивалентна задаче П, в которой эффективности дуг равны $S_{ij} - p t_{ij}$. Подставляя в формулу (2.8), получаем

$$\hat{S} = \frac{T - T_2}{T_1 - T_2}(S_1 - pT_1) + \frac{T_1 - T}{T_1 - T_2}(S_2 - pT_2) + pT =$$

$$= \frac{T - T_2}{T_1 - T_2} S_1 + \frac{T_1 - T}{T_1 - T_2} S_2 \qquad (2.10)$$

Рассмотрим задачу Б. Обозначим $x_{ij} = 1$, если дуга (ij) принадлежит пути μ, и $x_{ij} = 0$, в противном случае. Задачу Б можно сформулировать в следующем виде:

определить $x_{ij} = 0$ или 1,

$(i,j) \in U$ максимизирующие

$$\sum_{ij} x_{ij} (S_{ij} - q t_{ij}) + qT$$

при ограничениях

$$\sum_i x_{oi} = \sum x_{in} = 1$$

$$\sum_j x_{ij} = \sum_k x_{ki}, \quad i = 1, 2, \ldots, n-1,$$

$$\sum_{ij} x_{ij} t_{ij} \geq T.$$

Заменяя дискретное условие $x_{ij} = 0$ или 1 непрерывным $0 \leq x_{ij} \leq 1$ для всех $(ij) \in U$ и переходя к двойственной задаче, получаем, что двойственная задача сводится к определению $\lambda \geq 0$, минимизирующего

$$M(\lambda) - \lambda T + qT$$

где $M(\lambda)$ - длина максимального пути в сети при длинах дуг $l_{ij} = S_{ij} - q t_{ij} + \lambda t_{ij}$. Функция $M(\lambda)$ - неубывающая выпуклая (книзу). Поэтому необходимые и достаточные условия оптимальности можно записать в виде

$$T(\lambda - 0) \leq T \leq T(\lambda + 0) \qquad (2.11)$$

где $T(\lambda)$ - продолжительность максимального пути в зависимости от λ.

Обозначим μ_1 - путь максимальной длины, имеющий

продолжительность $T_1 = T(\lambda+0)$.

пути μ_1 и μ_2 совпадают с соответствующими путями в задаче А. Обозначаем далее

$$S_1 - q T_1 = \sum_{(ij)\in\mu_1} (S_{ij} - q t_{ij})$$

$$S_2 - q T_2 = \sum_{(ij)\in\mu_2} (S_{ij} - q t_{ij})$$

Получаем, что оценка сверху величины $S(\mu) - F(\mu)$ для задачи Б равна

$$\hat{S} = \frac{T - T_2}{T_1 - T_2}(S_1 - q T_1) + \frac{T_1 - T}{T_1 - T_2}(S_2 - q T_2) + q T =$$

$$= \frac{T - T_2}{T_1 - T_2} S_1 + \frac{T_1 - T}{T_1 - T_2} S_2 ,$$

что полностью совпадает с (2.10).

Пример.

Рассмотрим на примере использование изложенных в предыдущем пункте результатов. На рис.2.9 изображена сеть, данные об эффективностях и продолжительностях дуг которой приведены ниже

(ij)	(01)	(02)	(03)	(13)	(15)	(23)	(24)	(26)
S_{ij}	15	20	11	14	12	10	18	21
t_{ij}	7	9	5	6	3	5	8	7

(ij)	(34)	(35)	(45)	(46)	(47)	(57)	(67)
S_{ij}	16	13	26	19	17	20	15
t_{ij}	5	3	10	8	5	7	3

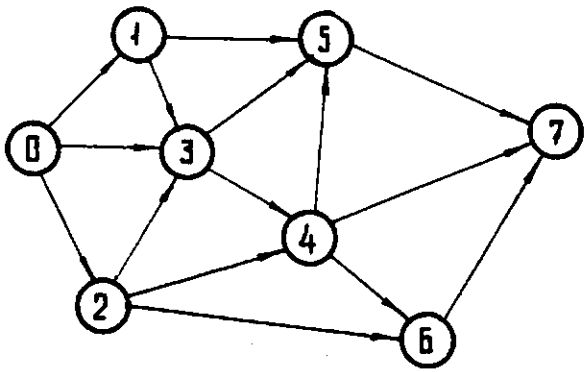


Рис. 2.9.

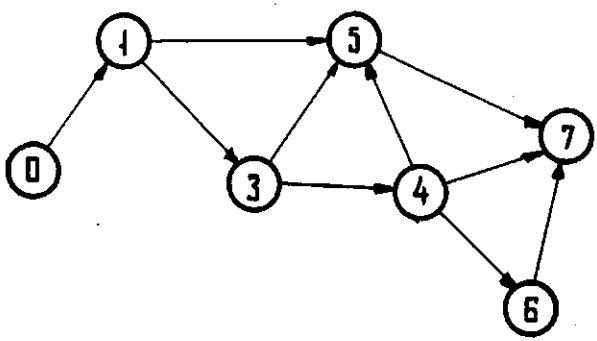


Рис. 2.10

Пусть $p = 1$, $q = 3$, $T = 20$.

Предварительно вычислим длину $L_i(p)$ максимального пути из вершины i в конечную вершину при длинах дуг

$$l_{ij} = S_{ij} - p t_{ij} = S_{ij} - t_{ij}, \quad (ij) \in U$$

длину $L_i(q)$ максимального пути из вершины i в конечную вершину при длинах дуг $l_{ij} = S_{ij} - q t_{ij} = S_{ij} - 3t_{ij}$; $(ij) \in U$ и соответственно продолжительности $T_i(p)$ и $T_i(q)$ этих путей. Проверьте самостоятельно результаты вычислений

i	0	1	2	3	4	5	6
$L_i(p)$	56	48	45	40	29	13	12
$T_i(p)$	35	28	27	22	17	7	3
$L_i(q)$	-1	2	6	3	2	-1	6
$T_i(q)$	19	10	10	10	5	7	3

при вычислении может встретиться случай равных $L_i(p)$ для нескольких путей. Так $L_0(p) = 56$ имеют два пути (0,1,3,4,5,7) и (0,2,3,4,5,7). В этом случае предпочтительнее в качестве $T_i(p)$ брать минимальную из продолжительностей этих путей (в нашем примере $T_0(p) = 35$ имеет путь (0,1,3,4,5,7). Наоборот, если два или более путей имеют равные $L_i(q)$, то предпочтительнее брать в качестве $T_i(q)$ максимальную из продолжительностей путей. Так $L_0(q) = -1$ имеют пути (0,3,5,7) (0,3,4,7) и (0,2,6,7). Максимальную продолжительность $T_0(q) = 19$ имеет путь (0,2,6,7). При таком выборе более вероятно получить $T_0(p) \le T$ или $T_0(q) \ge T$, что соответствует легко разрешимым случаям задачи. Например, если бы

T = 19, то путь (0,2,6,7) определил бы оптимальное решение задачи с величиной $S(\mu) - F(\mu) = 1 + 3 \cdot 19 = 56$. Так как $T_0(p) > T > T_0(q)$, то имеем случай (2.9). Поэтому применим метод ветвей и границ. Разобьем множество всех решений на три подмножества в зависимости от выбора первой дуги искомого пути и определим оценки сверху каждого из подмножеств. Будем обозначать $M_i(\lambda)$ длину максимального пути из вершины i в конечную вершину в зависимости от λ при длинах дуг $l_{ij} = S_{ij} - q t_{ij} + \lambda t_{ij}$ Оценим подмножество решений, в которых $x_{01} = 1$. Для этого определим $M_1(\lambda)$. Очевидно, $M_1(0) = \mathcal{L}_1(q)$, а начальный участок функций $M_1(\lambda)$ имеет вид $\mathcal{L}_1(q) + \lambda T_1(q) = 2 + 10\lambda$. Найдем ближайшую точку, в которой функция $M_1(\lambda)$ меняет наклон. На рис.2.10 изображена часть сети с длинами дуг $(S_{ij} - q t_{ij} + \lambda t_{ij})$. Вычисляем последовательно потенциалы U_i вершин в зависимости от λ

$$U_7 = 0$$
$$U_6 = 6 + 3\lambda,$$
$$U_5 = -1 + 7\lambda,$$
$$U_4 = \max(-5 + 17\lambda;\ 2 + 5\lambda;\ 1 + 11\lambda) = 2 + 5\lambda$$

при $\lambda \le \min\left(\frac{17}{2};\ \frac{1}{6}\right) = \frac{1}{6}$.

$$U_3 = \max(3 + 10\lambda;\ 3 + 10\lambda) = 3 + 10\lambda,$$
$$U_1 = \max(2 + 10\lambda;\ -1 + 16\lambda) = 2 + 10\lambda$$

при $\lambda \le \frac{1}{2}$

При $\lambda \ge \frac{1}{6}$ потенциал четвертой вершины будет вычисляться по формуле $U_4 = \max(-5 + 17\lambda,\ 1 + 11\lambda) = 1 + 11\lambda$ при

$$\frac{1}{6} \le \lambda \le 1$$

Далее

$$U_3 = \max(3+10\lambda;\ 2+16\lambda) = 2+16\lambda \quad \text{при } \tfrac{1}{6} \le \lambda$$

$$U_4 = \max(2+10\lambda;\ -2+22\lambda) = 2+10\lambda \quad \text{при } \tfrac{1}{3} \le \lambda$$

Итак, $M_1(\lambda) = U_4$ меняет наклон впервые при $\lambda = \frac{1}{3}$ Соответствующий новый путь максимальной длины (I,3,4,6,7) имеет продолжительность 22 и эффективность 64. Так как $22 > T - t_{01} = 13$, то условия (2.I2) выполнены. Обозначив μ_1 путь (I,5,7) и μ_2 - путь (I,3,4,6,7), имеем $T_1 = 10$, $S_1 = 32$, $T_2 = 22$, $S_2 = 64$ и по формуле (2.II) получаем оценку $\hat{S}(01)$ подмножества $X_{01} = I$:

$$\hat{S}(01) = \frac{22-13}{22-10} \cdot 32 + \frac{13-10}{22-10} \cdot 64 + 15 = 55.$$

П. Оценим подмножество решений, в которых $X_{02} = I$. Поскольку потенциалы U_3, U_4, U_5, U_6 и U_7 были вычислены на предыдущем шаге, то сразу вычисляем

$$U_2 = \max(-2+15\lambda,\ -4+13\lambda,\ 6+10\lambda) = 6+10\lambda$$

при $\lambda \le \min(1{,}6,\ 3\tfrac{1}{3}) = 1{,}6$.

При $\lambda \ge \frac{1}{6}$ потенциал четвертой вершины будет

$$U_4 = 1+11\lambda, \quad \text{при } \tfrac{1}{6} \le \lambda \le 1$$

а третий $U_3 = 2+16\lambda$

При этом

$$U_2 = \max(-3+21\lambda,\ -4+13\lambda,\ 6+10\lambda) = 6+10\lambda$$

$$\text{при } \tfrac{1}{6} \le \lambda \le \min\left(\tfrac{9}{11},\ 3\tfrac{1}{3}\right) = \tfrac{9}{11}$$

Следовательно, при $\lambda = \frac{9}{11}$ ломаная $M_2(\lambda) = U_2$ меняет

наклон. Новый путь максимальной длины (2,3,4,6,7) имеет продолжительность $21 > T - t_{02} = 11$ и эффективность 60. Условия оптимальности (6.I.9) выполнены, поэтому получаем

$$\hat{S}(0,2) = \frac{21-11}{21-10} \cdot 36 + \frac{11-10}{21-10} \cdot 60 + 20 = 58\frac{2}{11}.$$

Ш. Оценим подмножество решений, в которых $x_{03} = 1$. На основании предыдущих результатов сразу получаем, что $M_3(\lambda)$ меняет наклон впервые при $\lambda = \frac{1}{6}$. Новый путь максимальной длины (3,4,6,7) имеет продолжительность $16 > T - t_{03} = 15$ и эффективность 50. Условия (2.I2) выполнены Поэтому

$$\hat{S}(0,3) = \frac{16-15}{16-10} \cdot 33 + \frac{15-10}{16-10} \cdot 50 + 11 = 58\frac{1}{6}.$$

Выбираем подмножество с максимальной оценкой, то есть полагаем $x_{02} = 1$. Разбиваем это подмножество на три:

I. $x_{23} = 1$. При этом, $T - t_{02} - t_{23} = 6 < T_3(q) < T_3(p)$, что соответствует случаю (а). Значит путь (0,2,3,4,7) определяет оптимальное решение в рассматриваемом подмножестве. Эффективность пути (0,2,3,4,7) равна 63, продолжительность равна 24. Следовательно, оценка подмножества равна

$$S(M) = F(M) = 63 - 3 \cdot 4 = 51.$$

П. $x_{24} = 1$, $T - t_{02} - t_{24} = 3 < T_4(q) < T_4(p)$

Поэтому, аналогично предыдущему случаю, путь (0,2,4,7) определяет наилучшее решение в данном подмножестве. Эффективность пути равна 55, продолжительность - 22. Следовательно оценка подмножества равна 55 - 2·3 = 49.

Ш. X_{26} = I. В этом подмножестве существует единственный путь (0,2,6,7), эффективность которого 56, продолжительность I9, а разность эффективности и штрафа равна 56+I·I=57.

Теперь максимальная оценка равна $58\frac{1}{6}$ Выбираем подмножество с максимальной оценкой, то есть полагаем X_{03} = I. Разбиваем это подмножество на два. В первом полагаем X_{34} = I, а во втором X_{35} = I. Заметим, что оценка подмножества решений, в которых X_{34} = I, по-прежнему равна $58\frac{1}{6}$. Подмножество решений, в которых X_{35} = I, состоит всего из одного пути (0,3,5,7) продолжительности I5 и эффективности 44. Для этого подмножества

$$S(M) - F(M) = 44 + 5 = 49 < 58\frac{1}{6}.$$

Поэтому выбираем подмножество X_{03} = I, X_{34} = I. Разбивая это подмножество на три, непосредственно проверяем: путь (0,3,4,5,7) имеет эффективность 73 и продолжительность 27.

Поэтому $$S(M) - F(M) = 73 - 3 \cdot 7 = 52$$

путь (0,3,4,7) имеет эффективность 44 и продолжительность I5.

Поэтому $$S(M) - F(M) = 44 + 5 = 49.$$

Путь (0,3,4,6,7) имеет эффективность 6I и продолжительность 2I.

Поэтому $$S(M) - F(M) = 61 - 3 \cdot 1 = 58.$$

Получили решение (0,3,4,6,7), которое является оптимальным,

так как оценки остальных подмножеств меньше 58.

Упражнение 2.8. Решите пример при следующих значениях p и q

N	I	2	3
p	3	2	I
q	I	3	6

Упражнение 2.9. Определите максимальную эффективность пути в сети из предыдущего примера, при условии, что продолжительность пути не превышает $T = 20$.

Упражнение 2.IO. Определите путь максимальной средней эффективности в сети из предыдущего примера.

Упражнение 2.II. Решите задачу Ш для частного случая $S_{ij} = ct_{ij}$ для всех $(ij) \in U$ (с - положительная величина).

Глава Ш

ЦИРКУЛЯЦИЯ МАКСИМАЛЬНОЙ ВЕЛИЧИНЫ И ПОТЕНЦИАЛ ПЕРЕСТАНОВКИ ВЕРШИН ГРАФА

В настоящей главе мы введем понятия циркуляции и потенциала перестановок вершин в ориентированных графах, используя которые рассмотрим вопросы существования гамильтоновых путей и контуров в полных графах, а также связь между величиной циркуляции и потенциалом перестановки в графах.

Сформулировав задачу о минимальном потенциале перестановки, предложим несколько интересных ее интерпретаций относящихся к задачам на разрыв контуров, возникающих при размещении информационных массивов и обнаружении неисправностей в информационных системах и наконец рассмотрим алгоритм решения задачи о минимальном потенциале и связанных с ней прикладных задач.

§ I. Циркуляция максимальной величины

В этом параграфе мы будем рассматривать ориентированные графы. Каждой дуге (i,j) отнесено действительное число $c_{ij}>0$ называемое пропускной способностью дуги. Таким образом, графу можно поставить в соответствие матрицу пропускных способностей его дуг. Будем считать, что $c_{ii}=0$ и $c_{ij}=0$, если $(i,j)\bar{\in}U$. Обозначим M - множество контуров графа, μ - произвольный контур множества M. Определим для каждого $\mu\in M$ неотрицательное число $\varphi(\mu)$, которое назовем циркуляцией по контуру μ. Обозначим также M_{ij} - множество контуров, содержащих

дугу (i, j).

Определение 3.1. Циркуляцией в графе называется множество циркуляций по контурам графа, удовлетворяющих условию

$$\sum_{\mu \in M_{ij}} \varphi(\mu) \leqslant C_{ij} \quad \text{для всех } (i,j) \in U \qquad (3.1)$$

Определение 3.2. Величиной φ циркуляции называется сумма циркуляций всех контуров графа, то есть

$$\varphi = \sum_{\mu \in M} \varphi(\mu) \qquad (3.2)$$

Пример 3.1. Рассмотрим следующую матрицу пропускных способностей

i \ j	1	2	3	4	5	6
1	0	1	0	0	0	0
2	0	0	1	0	1	0
3	0	0	0	1	0	0
4	1	0	0	0	1	0
5	0	0	0	0	0	1
6	1	0	1	0	0	0

Соответствующий граф показан на рис.3.1. Он имеет пять контуров: $\mu_1 = (1,2,3,4,1)$, $\mu_2 = (3,4,5,6,3)$, $\mu_3 = (1,2,5,6,1)$ $\mu_4 = (1,2,3,4,5,6,1)$, $\mu_5 = (1,2,5,6,3,4,1)$
Любой набор неотрицательных чисел $\varphi(\mu_1)$, $\varphi(\mu_2)$, $\varphi(\mu_3)$ $\varphi(\mu_4)$, $\varphi(\mu_5)$, удовлетворяющих матрице пропускных способностей,

определяет циркуляцию в графе. Ее величина равна $\varphi = \varphi(\mu_1) + \varphi(\mu_2) + \varphi(\mu_3) + \varphi(\mu_4) + \varphi(\mu_5)$

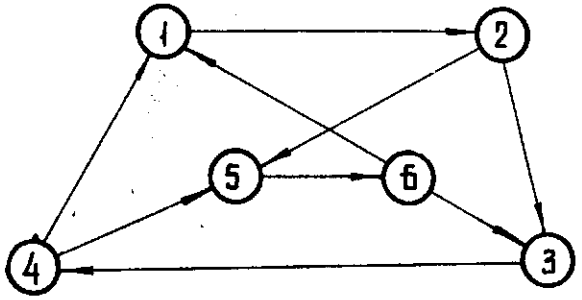


Рис 3.1

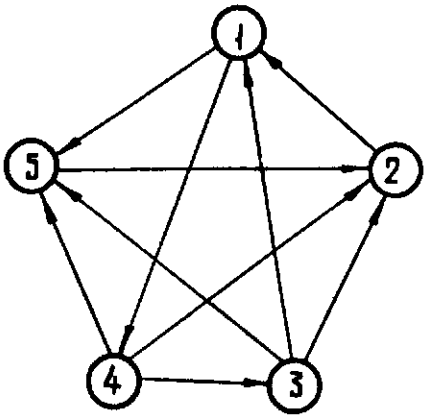


Рис. 3.2

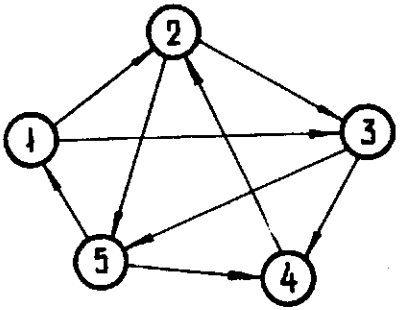


Рис. 3.3

Задача 3.I. Определить циркуляцию максимальной величины.

Двойственной к этой задаче является

Задача 3.2. Определить числа $y_{ij} \geq 0$ на дугах графа, удовлетворяющие условиям

$$\sum_{(i,j)\in\mu} y_{ij} \geq 1 \qquad \text{для всех} \quad \mu \in M \qquad (3.3)$$

и минимизирующие

$$C = \sum_{(i,j)\in U} c_{ij}\, y_{ij} \qquad (3.4)$$

Из теории двойственности линейного программирования известно, что минимальное значение C равно максимальной величине циркуляции, то есть

$$C_{min} = \varphi_{max} \qquad (3.5)$$

Пример 3.2. Для графа из примера 3.I. циркуляция максимальной величины

$$\varphi(\mu_1) = \tfrac{1}{2},\ \varphi(\mu_2) = \tfrac{1}{2},\ \varphi(\mu_3) = \tfrac{1}{2},\ \varphi(\mu_4) = 0,\ \varphi(\mu_5) = 0,\ \varphi_{max} = \tfrac{3}{2}.$$

Соответственно, оптимальное решение задачи I.3.2. имеет вид $y_{12} = y_{34} = y_{56} = \frac{1}{2}$ остальные $y_{ij} = 0$.

При этом

$$C_{min} = \tfrac{3}{2} = \varphi_{max}$$

Упражнение 3.I. Предложите алгоритм определения циркуляции максимальной величины.

§ 2. Потенциал перестановки

Пусть $(i_1, i_2, \ldots, i_n)$ некоторая перестановка номеров вершин графа. Дугу (i_κ, i_s) графа назовем неправильно ориентированной относительно перестановки $(i_1, i_2, \ldots, i_n)$, если $\kappa > s$

Определение 3.3.Потенциалом $\Pi(i_1, i_2, \ldots, i_n)$ перестановки $(i_1, i_2, \ldots, i_n)$ называется сумма пропускных способностей дуг,неправильно ориентированных относительно этой перестановки, то есть

$$\Pi(i_1, i_2, \ldots, i_n) = \sum_{k>s} c_{i_k i_s} \qquad (3.6)$$

Матрица пропускных способностей называется <u>циклической</u>, если $\sum_j c_{ij} = \sum_k c_{ki}$ для всех $i = 1, 2, \ldots, n$

В случае циклической матрицы потенциал также называется циклическим.

Рассмотрим некоторые свойства потенциалов.

<u>Свойство I.</u>

$$\Pi(i_1, \ldots, i_k, i_{k+1}, \ldots, i_n) - \Pi(i_1, \ldots, i_{k+1}, i_k, \ldots, i_n) = c_{i_k i_{k+1}} - c_{i_{k+1} i_k} \qquad (3.7)$$

то есть при транспозиции элементов перестановки потенциал изменяется на разность соответствующих элементов матрицы пропускных способностей. Для циклических потенциалов

$$\Pi(i_n, i_2, \ldots, i_{n-1}, i_1) - \Pi(i_1, i_2, \ldots, i_n) = c_{i_1 i_n} - c_{i_n i_1} \qquad (3.8)$$

<u>Свойство 2.</u>

$$\Pi(i_1, i_2, \ldots, i_n) + \Pi(i_n, i_{n-1}, \ldots, i_1) = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} \qquad (3.9)$$

то есть сумма потенциалов двух обратных перестановок равна постоянной величине (сумме всех пропускных способностей дуг графа).

<u>Упражнение 3.2.</u> Докажите свойства I и 2.

Пример I.3.3. Возьмем для графа, показанного на рис.3.I. с матрицей пропускных способностей из примера

3.I. перестановку вершин (2, I,3,6,4,5). Дуги (I,2), (4,I), (5,6), (6,3) и (6,I) являются неправильно ориентированными относительно перестановки. Следовательно

$$\mathcal{T}(2,I,3,6,4,5) = 5.$$

Если переставить местами соседние номера 3 и 6, то дуга (6,3) будет уже правильно ориентированной, и потенциал $\mathcal{T}(2,I,6,3,4,5)$ будет равен 4, то есть уменьшится на I.

Относительно обратной перестановки (5,4,6,3,I,2) неправильно ориентированными являются дуги (2,3), (2,5), (3,4) (4,5), которые были правильно ориентированными относительно перестановки (2,I,3,6,4,5).

Поэтому $\mathcal{T}(2,I,3,6,4,5) + \mathcal{T}(5,4,6,3,I,2) = 9$.

§ 3. Существование гамильтоновых путей и контуров в полных графах

Проиллюстрируем применение понятия потенциала для доказательства ряда известных результатов теории графов.

Теорема 3.I. В полном ориентированном графе всегда имеется гамильтонов путь.

Доказательство. Возьмем матрицу смежности графа в качестве матрицы пропускных способностей. Пусть $(i_1, i_2, \ldots, i_n)$ - произвольная перестановка вершин графа. Если эта перестановка не определяет гамильтонов путь, то найдется пара i_k, i_{k+1} такая, что дуга $(i_k, i_{k+1}) \in U$. Переставим i_k и i_{k+1} местами. При этом потенциал новой перестановки уменьшится на I. Продолжая таким образом, обязательно получим гамильтонов путь, так как потенциал любой перестановки неотрицателен.

Пример 3.4. Найдем гамильтонов путь в графе, показанном на рис.3.2. Возьмем начальную перестановку (I,2,3,4,5). Так как $(I,2)\in U$, то переходим к перестановке (2,I,3,4,5). Далее получаем последовательно (2,3,I,4,5), (3,2,I,4,5) - гамильтонов путь.

Теорема 3.2. В полном ориентированном сильно связном графе всегда имеется гамильтонов контур.

Доказательство. Построим произвольную циркуляцию $\varphi(\mu)$ в графе, так чтобы циркуляция по всем дугам была больше нуля (это всегда возможно, если граф сильно связен). Примем $\sum_{\mu\in M_{ij}} \varphi(\mu)$ за пропускную способность дуги (i, j). Пусть $(i_1, i_2, \dots, i_n)$ произвольная перестановка. Если она не определяет гамильтонов контур, то найдется пара i_k, i_{k+1} такая, что $(i_k, i_{k+1})\in U$ (при этом считаем $i_{n+1} = i_1$). Переставим i_k и i_{k+1} местами. Потенциал полученной перестановки будет меньше первоначальной на $C_{i_{k+1}, i_k} > 0$. Продолжая таким образом, обязательно получим гамильтонов контур, так как циклический потенциал любой перестановки неотрицателен.

Пример 3.5. Для графа, показанного на рис.3.2 в примере 3.4. была получена перестановка (3,2,I,4,5), определяющая гамильтонов путь. Поскольку дуга $(5,3)\in U$, то продолжим процедуру, переставляя 3 и 5. Полученная перестановка (5,2,I,4,3) определяет гамильтонов контур.

§ 4. Задача о минимальном потенциале

Задача 3.2. Определить перестановку, имеющую минимальный потенциал.

Обозначим минимальный потенциал P_{min}.

Теорема 3.3. Если граф не имеет контуров, то $\mathcal{T}_{min} = 0$.

Доказательство. В графе без контуров всегда возможно правильное упорядочение номеров вершин, при котором если $(i_k, i_s) \in U$, то $k < s$. Для соответствующей перестановки все дуги являются правильно ориентированными, и значит ее потенциал равен 0.

Обозначим U_c множество дуг, после удаления которых в графе не будет контуров. Из теоремы 3.3. следует, что

$$\mathcal{T}_{min} = \min_{U_c} \sum_{(i,j) \in U_c} C_{ij} \qquad (3.10)$$

Таким образом, минимальный потенциал равен минимальной сумме пропускных способностей дуг, после удаления которых в графе не будет контуров.

Теорема 3.4. Потенциал любой перестановки номеров вершин графа больше или равен величине любой циркуляции в этом графе (докажите эту теорему самостоятельно).

В частности, из теоремы 3.4 следует, что

$$\mathcal{T}_{min} \geq \varphi_{max} = C_{min} \qquad (3.11)$$

Так для графа, показанного на рис.3.1 $\varphi_{max} = 1{,}5$, $\mathcal{T}_{min} = 2$. Этот пример показывает, что равенство $\mathcal{T}_{min} = \varphi_{max}$ в общем случае не выполняется.

Определение 3.4. Величина $\psi_{ij} = \sum_{\mu \in \mu_{ij}} \varphi(\mu)$ называется потоком по дуге (i , j). Множество потоков по дугам графа образует поток в графе. Одному и тому же потоку может соответствовать, в общем случае, множество циркуляций. Если $\{\psi_{ij}\}$ поток, то для любой вершины i графа сумма входящих потоков равна сумме выходящих.

Упражнение 3.3. Пусть C_{ij} целые неотрицательные числа и $\{\varphi_{ij}\}$ целочисленный поток по графу. Существует ли целочисленная циркуляция максимальной величины? Будет ли справедливо в этом случае равенство $F_{min} = \varphi_{max}$?

Из определения 3.4. видна связь между циркуляциями и потоком в графе. В литературе очень подробно рассмотрена важная в прикладном отношении задача о потоке максимальной величины в графах.

Этому вопросу посвящена, в частности, специальная монография Л.Форда и Д.Фалкерсона "Потоки в сетях", с множеством прикладных задач; алгоритм поиска максимального потока подробно изложен в монографии К.Бержа "Теория графов и ее применения". Это позволяет авторам не останавливаться на обсуждении теории потоков в графах, ограничившись связью с циркуляцией в графах и перейти к оригинальным задачам на разрыв контуров.

§ 5. Задачи на разрыв контуров

В предыдущих параграфах мы рассмотрели понятие потенциала графа и поставили задачу определения минимального потенциала. Задача по сути дела сводится к удалению из графа дуг так, чтобы разорвать все контура и чтобы при этом сумма весов (пропускных способностей) удаленных дуг была минимальной. Цель этого параграфа - рассмотреть несколько интересных интерпретаций задачи и обсудить возможные пути ее решения.

5.I. Размещение информационных массивов

Требуется разместить n информационных массивов на ленте с однонаправленным поиском и считыванием (либо на кольцевой ленте). Известна вероятность p_i обращения к i -му массиву условная вероятность p_{ij} обращения к j - му массиву, если перед этим было обращение к i -ому массиву $i, j = 1, 2, \ldots, n$. Обозначим τ_i - время просмотра i -го массива при поиске, $T = \sum_{i=1}^{n} \tau_i$ общее время просмотра всех массивов при поиске, θ - время обратной перемотки ленты (или время просмотра свободной части ленты в случае кольцевой ленты). Задача заключается в размещении массивов на ленте таким образом, чтобы среднее время поиска было минимальным. Среднее время поиска для некоторого размещения $(i_1, i_2, \ldots, i_n)$ можно записать в следующем виде

$$T_c(i_1, i_2, \ldots, i_n) = \sum_{\kappa>s-1} q_{i_s i_\kappa} \left(\sum_{q=s-1}^{\kappa-1} \tau_{i_q} \right) +$$

$$+ \sum_{\kappa \le s} \left(T + \theta - \sum_{q=\kappa}^{s} \tau_{i_q} \right) q_{i_\kappa i_s}, \text{ где } q_{i_s i_\kappa} = p_{i_s} \cdot p_{i_s i_\kappa} \quad (3.12)$$

Выведем необходимые условия оптимальности перестановки $(i_1, i_2, \ldots, i_n)$. Для этого поменяем местами два соседних массива i_{s-1} и i_s (очевидно, это не отразится на положении остальных массивов). Вычислим разность средних времен поиска информации $T_{i_{s-1} i_s}$, $T_{i_s i_{s-1}}$ соответственно старого и нового размещения. Имеем

$$T_{i_{s-1}i_s} - T_{i_s i_{s-1}} = \sum_{j \neq s, s-1} r_{i_s} (q_{i_{s-1} i_j} - q_{i_j i_{s-1}}) + \sum_{j \neq s, s-1} r_{i_{s-1}} (q_{i_j i_s} - q_{i_s i_j}) +$$

$$+ (T+\theta - r_{i_{s-1}} - r_{i_s})(q_{i_s i_{s-1}} - q_{i_{s-1} i_s}) = r_{i_s} \sum_{j=1}^{n} (q_{i_{s-1} i_j} - q_{i_j i_{s-1}}) +$$

$$+ r_{i_{s-1}} \sum_{j=1}^{n} (q_{i_j i_s} - q_{i_s i_j}) + (T+\theta)(q_{i_s i_{s-1}} - q_{i_{s-1} i_s})$$

Обозначая $\Delta_{ij} = q_{ji} - q_{ij}$ и учитывая, что

$$\sum_{j=1}^{n} \Delta_{ij} = \sum_{i=1}^{n} \Delta_{ij} = 0$$

получаем окончательно

$$T_{i_{s-1} i_s} - T_{i_s i_{s-1}} = (T+\theta) \cdot \Delta_{i_{s-1} i_s} \qquad (3.13)$$

Из (3.13) следует, что необходимыми условиями оптимальности размещения $(i_1, i_2, \ldots, i_n)$ являются следующие

$$\Delta_{i_{s-1} i_s} \leq 0 \qquad \text{для всех} \quad s = 2, 3, \ldots, n \qquad (3.14)$$

Определим матрицу (C_{ij}) с элементами

$$C_{ij} = (T+\theta) \max (\Delta_{ij}; 0) \qquad (3.15)$$

Вычислим потенциал перестановки $(i_1, i_2, \ldots, i_n)$, связанный с матрицей (C_{ij}). Согласно определению потенциала имеем

$$\mathcal{A}(i_1, i_2, \ldots, i_n) = \sum_{k<s} C_{i_s i_k}$$

По свойству I потенциалов

$$\mathcal{A}(i_1, \ldots, i_{s-1}, i_s, i_{s+1}, \ldots, i_n) - \mathcal{A}(i_1, \ldots, i_s, i_{s-1}, \ldots, i_n) = C_{i_{s-1} i_s} - C_{i_s i_{s-1}} = \qquad (3.16)$$

$$= (T+\theta) \Delta_{i_{s-1} i_s}$$

Сравниваем (3.13) и (3.16), легко заметить, что

$$T_c(i_1,\dots,i_{s-1},i_s,\dots,i_n) - T_c(i_1,\dots,i_s,i_{s-1},\dots,i_n) =$$
$$\mathcal{A}(i_1,i_2,\dots,i_{s-1},i_s,\dots,i_n) - \mathcal{A}(i_1,\dots,i_s,i_{s-1},\dots,i_n)$$

или

$$T_c(i_1,i_2,\dots,i_{s-1},i_s,\dots,i_n) - \mathcal{A}(i_1,\dots,i_{s-1},i_s,\dots,i_n) =$$
$$T_c(i_1,\dots,i_s,i_{s-1},\dots,i_n) - \mathcal{A}(i_1,\dots,i_s,i_{s-1},\dots,i_n)$$

Отсюда непосредственно следует, что величина

$$S = T_c(i_1,i_2,\dots,i_n) - \mathcal{A}(i_1,i_2,\dots,i_n)$$

одна и та же для любой перестановки. Таким образом,

$$T_c(i_1,i_2,\dots,i_n) = S + \mathcal{A}(i_1,i_2,\dots,i_n)$$

и следовательно, задача минимизации среднего времени поиска эквивалентна задаче определения минимального потенциала.

Поскольку

$$\sum_{j=1}^{n} C_{ij} = \frac{1}{2}\sum_{j=1}^{n} |\Delta_{ij}| = \frac{1}{2}\sum_{k=1}^{n} |\Delta_{ki}| = \sum_{k=1}^{n} C_{ki},$$

то потенциалы перестановок являются к тому же циклическими.

Интересно отметить, что оптимальное решение задачи не зависит ни от времени τ_c, ни от времени θ, хотя минимальное значение среднего времени поиска конечно зависит от этих величин.

Пример 3.6. Пусть получена следующая последовательность выборки массивов (цифры указывают номера массивов)

1, 3, 2, 4, 3, 2, 1, 5, 2, 4, 1, 3, 2, 5, 4, 2, 1, 3,
2, 5, 4, 1, 3, 2, 1, 5, 4, 5, 2, 3, 1, 4, 2, 1, 3, 2,
5, 2, 5, 1, 3, 4, 2, 5, 4, 2, 5, 1, 3, 2, 1, ...

Обозначим $A=(a_{ij})$ матрицу обращений к массивам (элемент a_{ij} показывает сколько раз встретилась пара i,j в последовательности. Имеем

$$A = \begin{Vmatrix} 0 & 0 & 7 & I & 2 \\ 5 & 0 & I & 2 & 6 \\ I & 7 & 0 & I & 0 \\ 2 & 4 & I & 0 & I \\ 2 & 3 & 0 & 4 & 0 \end{Vmatrix}$$

Полное число обращений равно N = 50. Если обозначить

$$A_i = \sum_{j=1}^{5} a_{ij} = \sum_{k=1}^{5} a_{ki}, \text{ то } p_i = \frac{A_i}{N}, \; p_{ij} = \frac{a_{ij}}{A_i}, \; q_{ij} = \frac{a_{ij}}{N}$$

Пусть $T + \theta$ = 50. Тогда

$$c_{ij} = max\,(0;\, a_{jj} - a_{ij}).$$

Матрица $C=(c_{ij})$ приведена ниже

$$C = \begin{Vmatrix} 0 & 0 & 6 & 0 & 0 \\ 5 & 0 & 0 & 0 & 3 \\ 0 & 6 & 0 & 0 & 0 \\ I & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \end{Vmatrix}$$

В данном случае оптимальное решение получается сразу. Действительно, поскольку матрица C циклическая, то можно исключить любой столбец и соответствующую строку и решать задачу определения минимального потенциала (уже не циклического) для оставшейся матрицы. Исключим, например, второй столбец и вторую строку. Получим следующую матрицу

$$C' = \begin{Vmatrix} 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ I & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 \end{Vmatrix}$$

Соответствующий этой матрице граф не имеет контуров. Величина минимального потенциала равна 0 и достигается для перестановки (5, 4, I, 3). Следовательно, для исходной матрицы C величина минимального потенциала равна $\sum_{i=1}^{5} c_{i,i} = 8$ и достигается для циклической перестановки (2,5,4,I,3).

Упражнение 3.4. Определите среднее время поиска для оптимального размещения (2,5,4,I,3). Вычислите величину S. Пользуясь величиной S определите максимальную величину T_{max} среднего времени поиска. Полагая, что при случайном размещении массивов среднее время поиска равно $\frac{1}{2}(T_{max} + T_{min})$ вычислите относительное сокращение за счет оптимального размещения.

5.2. Обнаружение неисправностей

При обнаружении неисправностей в сложных схемах возникает задача наиболее рациональной последовательности проверки отдельных элементов схемы. При этом целесообразно использовать для проверки данного элемента уже проверенные элементы, выходы которых являются входами данного. Будем изображать схему в виде графа $G(X, U)$, вершины которого соответствуют элементам, а дуги - связям между элементами. Если граф не имеет контуров, то, очевидно, всегда можно организовать последовательность проверки элементов таким образом, что каждый элемент проверяется после про-

верки (и исправления неисправностей) всех предшествующих элементов. Однако большое количество обратных связей в сложных схемах не позволяет организовать такую процедуру проверки. Обозначим C_{ij} - затраты на дополнительное оборудование, если $(i,j) \in U$, а элемент j проверяется раньше элемента i.

Если $(i_1, i_2, \ldots, i_n)$ - последовательность проверки элементов, то затраты на дополнительное оборудование

$$\mathcal{K}(i_1, i_2, \ldots, i_n) = \sum_{k<s} C_{i_s i_k}$$

Легко видеть, что $\mathcal{K}$ является потенциалом перестановки при матрице пропускных способностей (C_{ij}). Поэтому задача минимизации затрат также сводится к задаче определения минимального потенциала перестановки.

§ 6. Алгоритм определения минимального потенциала

Без ограничения общности можно считать граф сильно связным (в противном случае задача решается отдельно для каждой сильно связной компоненты графа).

I шаг. Определяем все элементарные контуры графа. Это можно делать различными способами. Опишем один из простейших. Описание проведем на примере графа, изображенного на рис.3.3. Выбираем произвольную вершину, например, вершину I. Строим прадерево с корнем в вершине I, пути которого соответствуют элементарным путям и контурам графа. Процедура построения ясна из рис.3.4. и мы не будем ее описывать. Висячие вершины прадерева с номером I определяют все элементарные контуры графа, содержащие вер-

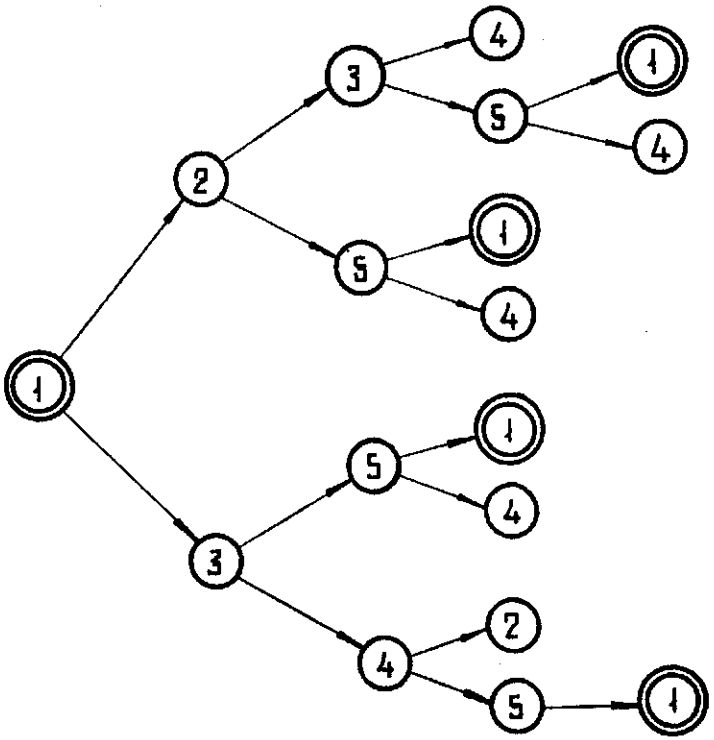


Рис. 3.4

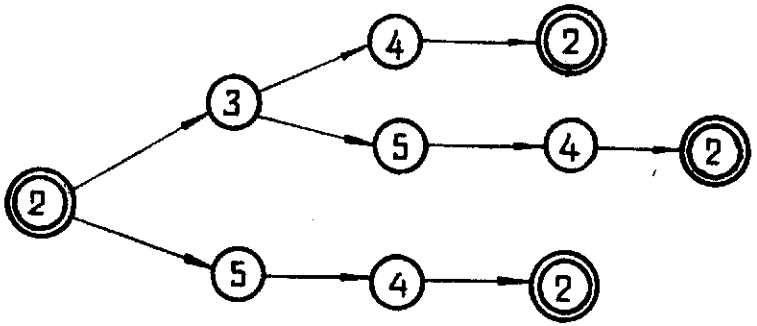


Рис. 3.5

шину I. Эти контуры

$$\mu_1 = (I,2,3,5,I),$$
$$\mu_2 = (I,2,5,I),$$
$$\mu_3 = (I,3,5,I),$$
$$\mu_4 = (I,3,4,2,5,I).$$

Удаляем вершину I, снова разбиваем граф на сильно связные компоненты (если он окажется не сильно связным) и повторяем описанную выше процедуру для каждой из полученных сильно связных компонент. После удаления вершины I граф останется сильно связным (проверьте самостоятельно). Поэтому выбираем следующую вершину, например, вершину 2, и снова строим прадерево с корнем в вершине 2 (рис.3.5). Его висячие вершины с номером 2 определяет все элементарные контуры, содержащие вершину 2 и не содержащие вершины I. Эти контуры

$$\mu_5 = (2,3,5,4,2),$$
$$\mu_6 = (2,3,4,2),$$
$$\mu_7 = (2,5,4,2).$$

Если теперь удалить вершину 2, то получим граф без контуров. Следовательно, определены все элементарные контуры графа.

П шаг. Определим двудольный граф $H(X,Y,V)$ вершины X которого соответствуют дугам исходного графа G, вершины Y - элементарным контурам графа G и дуги соединяют вершину $(i,j) \in X$ с вершиной $\mu_i \in Y$ если дуга (i,j) принадлежит контуру μ_i (3.6). Если

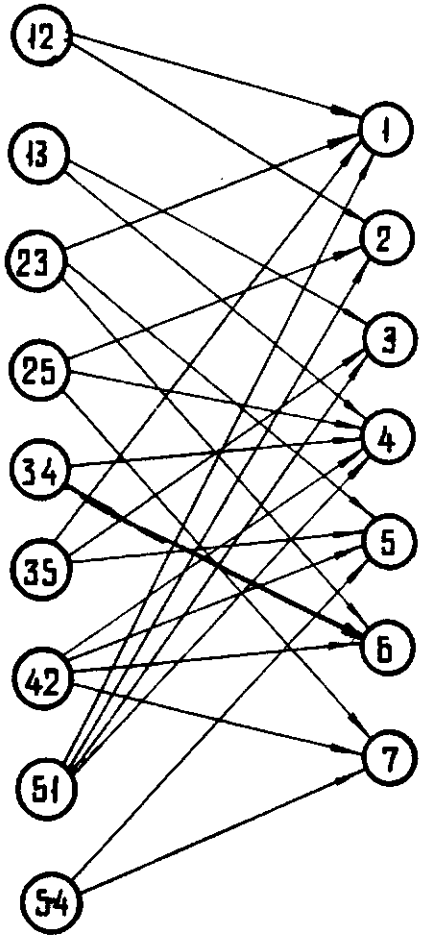


Рис. 3.6

интерпретировать вершины множества X как базы, а вершины множества Y как потребителей продукции, то получим задачу размещения баз, описанную в главе 4.

Поэтому метод ветвей и границ, применяемый для решения задачи размещения баз, можно использовать и для решения данной задачи. Решение для графа, изображенного на рис.3.6, предлагаем провести самостоятельно в качестве упражнения. Матрица (c_{ij}) пропускных способностей приведена ниже

$$C = \begin{Vmatrix} 0 & 2 & I & 0 & 0 \\ 0 & 0 & 4 & 0 & 2 \\ 0 & 0 & 0 & 6 & I \\ 0 & 3 & 0 & 0 & 0 \\ 5 & 0 & 0 & 2 & 0 \end{Vmatrix}$$

Глава IУ

РАЗБИЕНИЯ НА ГРАФАХ

Материал, излагаемый в этой главе связан с проблемой разбиения ребер и вершин конечного графа на множества некоторых определенных свойств (паросочетания,покрытия и т.д.),а также с использованием этих понятий для решения многих важных прикладных задач, в том числе,задач размещения объектов (заводов,складов и т.п.).

§ I. Разбиения ребер графа

Задачи разбиения ребер графа $G(X,V)$ рассматриваемые ниже,состоят либо в выделении из множества ребер графа V подмножества ребер $W \subset V$, в котором нет двух ребер инцидентных одной вершине $x \in X$ называемое паросочетанием, либо подмножества ребер D, такого что каждая вершина графа $x \in X$ инцидентна хотя бы одному ребру $D \subset V$, называемое реберным покрытием графа.
Не уменьшая общности будем считать, что граф $G(X,V)$связан. Если это не так, то задачи разбиения могут быть решены отдельно для каждой компоненты связности.

Очевидно, когда рассматриваются разбиения ребер графа V, имеет смысл говорить о графах или компонентах связности,содержащих по крайней мере одно ребро,т.е. множество V не пустое. Определнный класс задач на разбиения ребер графа составляют задачи поиска паросочетания W графа $G(X,V)$ с максимальным числом ребер и по-

крытия с минимальным числом ребер.

Таким образом, каждый граф $G(X,V)$ имеет паросочетание и покрытие и, очевидно, всегда существует максимальное паросочетание и минимальное покрытие, т.к. мы рассматриваем конечные графы. Интересна связь между максимальным паросочетанием и минимальным покрытием, которая будет рассмотрена позже. А сейчас мы переходим к теории паросочетаний или паросочетаний вершин ребрами и прикладным задачам решаемым на ее основе.

§ 2. Паросочетание графа

Пусть как и раньше в графе $G(X,V)$ X есть конечное множество вершин, а V - множество ребер (граф не ориентированный) и $W \subset V$ паросочетание графа. В настоящем параграфе мы будем рассматривать задачу определения максимального паросочетания, а также ее очевидное обобщение-задачу о максимальном взвешенном паросочетании (т.е. паросочетание, сумма весов которого максимальна), возникающую в случае, когда каждому ребру $v \in V$ назначен положительный вес $C(V)>0$.

Сформулированная задача может записана как задача целочисленного линейного программирования.

Пусть в дальнейшем $\sum(S,T)$, $S,T \in X$, $S \cap T \neq \Phi$ обозначает сумму $\sum(V)$ на ребрах $v \in V$, один конец которых принадлежит S, а другой T.
Тогда задача максимального (взвешенного) паросочетания определяется как:

$$\sum_{v} C(V)X(V) \to max \qquad (4.I)$$

при условиях

$$Z(i, N-i) \leq 1, \quad i \in X, \quad Z(V) = 0,1, \quad V \in V$$

В случае, когда $C(V)=1$, $V \in V$, получаем обычную задачу о поиске максимального паросочетания.

Задача 4.I. Мультипроцессорная машина с каналами передачи данных может быть представлена как граф $G(X,V)$, вершины которого X соответствуют процессорам, а ребра V - каналам передачи данных. Каждому ребру $v \in V$ приписан вес $C(V)>0$, соответствующий пропускной способности определенного канала. Считается, что каждый процессор может использовать в любой момент времени единственный канал для ввода и вывода сообщения. Требуется определить, какое максимальное число сообщений может одновременно передаваться в мультипроцессорной системе и какова будет наибольшая пропускная способность каналов при одновременных сообщениях?

Ответом на первый вопрос является максимальное подмножество ребер W в графе $G(X,V)$ - максимальное паросочетание, а ответом на второй вопрос подмножество W с максимальной суммой весов - максимальное взвешенное паросочетание.

Задачу максимального взвешенного паросочетания можно записать в более удобной форме используя матрицу инциденций $A=\|a_{ij}\|$ вершины ребра G, где

$$a_{ij} = \begin{cases} 1, \text{ если ребро } j \text{ инцидентно вершине } i; \\ 0, \text{ в противном случае} \end{cases}$$

и $x_j = \begin{cases} 1, \text{ если ребро } j \text{ войдет в паросочетание} \\ 0, \text{ в противном случае.} \end{cases}$

Нахождение максимального взвешенного **паросочетания** эквивалентно тогда максимизации

$$\sum_{j=1}^{n} c_j x_j \to max$$

при условии

$$\sum_{j=1}^{n} a_{ij} x_j \le 1, \quad i = 1,2,\dots,n \quad x_j = 0,1. \qquad (4.2)$$

Прежде чем переходить к обсуждению подходов решения к задаче максимального паросочетания, рассмотрим некоторые частные задачи паросочетания.

§ 3. Паросочетание простого графа

Будем рассматривать в этом параграфе только простые или двудольные графы $G(Y,Z,V)$, т.е. графы множество вершин которых X можно разбить на два непересекающихся подмножества Y и Z, причем ребра V соединяют вершины разных подмножеств.

Пусть n – число вершин первого подмножества, а m – число вершин второго. Обозначим c_{ij} вес или длину ребра $[i,j]$, соединяющего i-ю вершину первого подмножества с j-ой вершиной второго ($i = 1,2,\dots,n$, $j = 1,2,\dots,m$). Если $[i,j] \in V$, то примем c_{ij} равной любому отрицательному числу.

Паросочетанием простого графа называется подмножество ребер W, таких что никакие два ребра не смежны между собой. На рис.4.1. изображен простой граф. Подмножество ребер $[1,1]$, $[2,3]$, $[3,2]$, $[4,4]$, выделен-

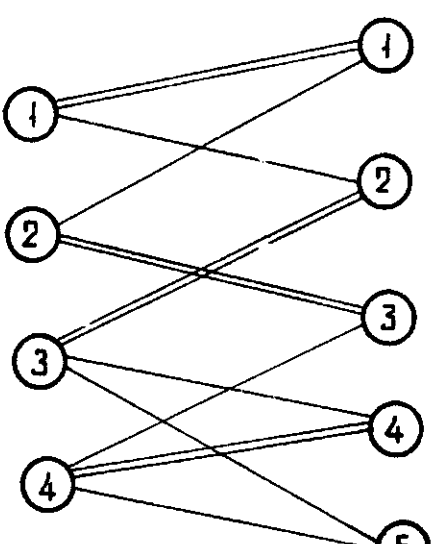


Рис. 4.1

ных двойными линиями, дает пример паросочетания. Задача формулируется как и прежде:

введем переменную x_{ij} , принимающую значение I, если ребро $[i,j]$ входит в паросочетание, и 0, в противном случае. Очевидно, значения x_{ij} определяют паросочетание в том и только в том случае, если выполняются условия

$$\sum_{j=1}^{m} x_{ij} \leq 1, \quad i = 1,2,\ldots,n$$
$$\sum_{i=1}^{n} x_{ij} \leq 1, \quad j = 1,2,\ldots,m \qquad (4.3)$$

Требуется определить $x_{ij}\ (i=1,2,\ldots,n,\ j=1,2,\ldots,m)$, удовлетворяющие (4.3) и максимизирующие

$$\mathcal{L} = \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} c_{ij} \qquad (4.4)$$

Задача назначения

В учреждении имеется n работников $x_1, x_2, \ldots, x_n$ и m должностей $z_1, z_2, \ldots, z_m$; квалификация каждого работника позволяет ему занимать некоторые из должностей, если работник Y_i может занимать должность z_j , то $(i,j) \in V$ в графе $G(Y,Z,V)$ и $(i,j) \notin V$ в противном случае. При условии, что известны эффективности c_{ij} использования Y_i на должности z_j , найти наиболее эффективное назначение работников на должности.
Если обозначить через x_{ij} переменную, принимающую значение I, если работник Y_i назначается на должность z_j и 0, в противном случае, то задача сведется к нахождению максимального взвешенного паросочетания простого графа $G(Y,Z,V)$. Задача максимального паросочетания или назначе-

ния является частным случаем общей задачи линейного программирования. Поэтому прежде, чем переходить к алгоритму решения излагаемому в следующем разделе, читателям, не знакомым с теорией двойственности в линейном программировании, необходимо изучить приложение I.

3.I. Задача двойственная к задаче паросочетания простого графа

Каждой вершине i первого подмножества поставим в соответствие двойственную переменную $U_i \geq 0$, а каждой вершине j второго подмножества двойственную переменную $V_j \geq 0$. Проверьте самостоятельно, что двойственная задача имеет вид: минимизировать:

$$B = \sum_{i=1}^{n} U_i + \sum_{j=1}^{m} V_j \qquad (4.5)$$

при ограничениях

$$U_i + V_j \geq C_{ij}; \quad i = 1,2,\ldots,n; \quad j = 1,2,\ldots,m. \qquad (4.6)$$

Заметим, что в оптимальном решении

$$V_j = \max [0; \max_i (C_{ij} - U_i)], \quad j = 1,2,\ldots,m$$

$$U_i = \max [0; \max_j (C_{ij} - V_j)], \quad i = 1,2,\ldots,n$$

Поэтому задача назначения эквивалентна задаче определения $U_i \geq 0$, $i = 1,2,\ldots,n$, минимизирующих

$$\sum_{i=1}^{n} U_i + \sum_{j=1}^{m} \max [0; \max_i (C_{ij} - U_i)]$$

или задаче определения $V_j \geq 0$, $j = 1,2,\ldots,m$ минимизирующих

$$\sum_{j=1}^{m} V_j + \sum_{i=1}^{n} \max [0; \max_j (C_{ij} - V_j)].$$

3.2. Алгоритм решения задачи

Пусть для определенности $n \le m$ и $\max_j C_{ij} > 0$ для всех i (если $\max_j C_{ij} \le 0$, то вершину i можно не рассматривать).

I шаг. Положим

$$X_{ij}^{\circ} = \begin{cases} 1, \text{ если } C_{ij} = \max_k C_{ik} \\ 0, \text{ в противном случае.} \end{cases}$$

Если при этом $\sum_{i=1}^{n} x_{ij}^{\circ} \le 1$, то получено оптимальное решение (докажите самостоятельно). Пусть для некоторой вершины j_1 имеет место $\sum_{i=1}^{n} x_{ij_1}^{\circ} > 1$.

2 шаг. Ребра (i,j), для которых $x_{ij}^{\circ} = 1$, ориентируем в направлении от j к i и полагаем $C_{ji} = -C_{ij}$; ребра, для которых $x_{ij}^{\circ} = 0$, ориентируем в направлении от i к j. Определяем путь μ максимальной длины, начинающийся в вершине j_1 и кончающийся либо в одной из вершин i, либо в вершине j, такой что $\sum_{i=1}^{n} x_{ij}^{\circ} = 0$. Полагаем

$$x_{ij}' = \begin{cases} x_{ij}^{\circ}, \text{если } (i,j) \notin \mu \\ 1 - x_{ij}^{\circ} \text{ если } (i,j) \in \mu \end{cases}$$

Если для всех j имеет место $\sum_{i=1}^{n} x_{ij}' \le 1$, то $\{x_{ij}'\}$ оптимальное решение. В противном случае повторяем шаг 2 для значений $\{x_{ij}'\}$. Очевидно, за конечное число шагов будут получены значения $\{x_{ij}\}$, для которых имеет место $\sum_{i=1}^{n} x_{ij} \le 1$ для всех j. Эти значения и определят оптимальное решение задачи паросочетания.

Пример 4.I. Значения C_{ij} для графа, изображенного на рис.4.1., приведены ниже

i \ j	I	2	3	4	5
I	5	3	x	x	x
2	4	x	I	x	x
3	x	6	x	7	4
4	x	x	7	9	3

I шаг. Определяя максимальные числа в каждой строке, получаем $x^0_{11} = I$, $x^0_{21} = I$, $x^0_{34} = I$, $x^0_{44} = I$, остальные $x^0_{ij} = 0$.

Для вершины $j_1 = I$ имеет место $\sum_{i=1}^{4} x^0_{ij} = 2 > 1$

2 шаг. Ориентируя ребра и проставляя длины дуг, получаем граф, изображенный на рис.4.2 (вершины, в которых могут оканчиваться пути выделены двойными кружками, вершина $j_1 = I$ отмечена квадратом). Применяя алгоритм Форда, определяем пути максимальной длины, оканчивающиеся в вершинах, отведенных кружками (установившиеся индексы вершин указаны у соответствующих вершин). Путь, имеющий максимальную длину, выделен двойными дугами. Получаем

$x^1_{12} = I$, $x^1_{21} = I$, $x^1_{34} = I$, $x^1_{44} = I$, остальные $x^1_{ij} = 0$.

Для вершины $j_2 = 4$ имеет место $\sum_{i=1}^{4} x_{i4} = 2 > 1$

3 шаг. Повторяем процедуру для графа, соответствующего набору $\{x^1_{ij}\}$ и изображенного на рис.4.3. Путь максимальной длины выделен двойными дугами. Получаем

$x^2_{12} = I$, $x^2_{21} = I$, $x^2_{34} = I$, $x^2_{43} = I$, остальные $x^2_{ij} = 0$.

Поскольку для всех j имеет место $\sum_{i=1}^{4} x^2_{ij} \leq 1$, то получено оптимальное решение задачи назначения. При этом $\mathcal{L}_{max} = 2I$.

3.3. Обоснование алгоритма

Обоснование алгоритма проведем в два этапа. Сначала

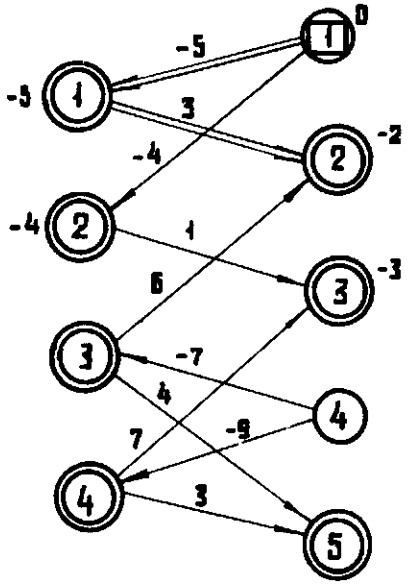


Рис. 4.2

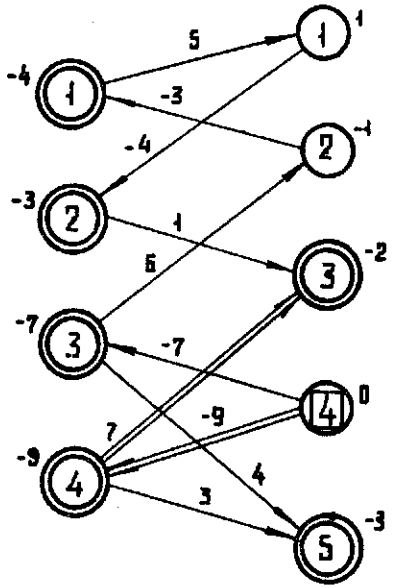


Рис. 4.3

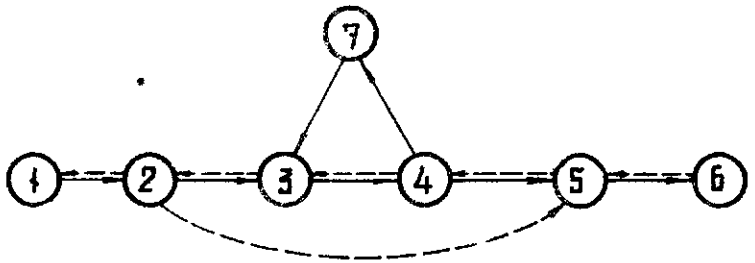


Рис. 4.4

выведем необходимые и достаточные условия оптимальности допустимого решения $\{x_{ij}\}$. Для этого ориентируем дуги графа как было описано выше, то есть в направлении от j к i, если $x_{ij} = 1$ (причем $c_{ji} = -c_{ij}$) и в направлении от i к j, если $x_{ij} = 0$. Полученный ориентированный граф обозначим H.

Теорема 4.1. Для оптимальности решения $\{x_{ij}\}$ необходимо и достаточно, чтобы граф H не имел контуров положительной длины (теорема приводится без доказательства).

Теперь осталось доказать, что для решения, полученного на последнем шаге алгоритма, условия теоремы выполняются. Для этого заметим, что граф H_1, полученный на первом шаге алгоритма, не имеет контуров положительной длины.

Докажем следующее общее утверждение. Пусть имеется произвольный граф без контуров положительной длины и пусть μ путь максимальной длины, соединяющий две разные произвольные вершины графа. Изменим направление дуг пути μ и их длины на обратные. Тогда полученный граф также не будет иметь контуров положительной длины. Для иллюстрации идеи доказательства рассмотрим рис.4.4. Пусть (1,2,3,4,5,6) путь максимальной длины. Предположим, что после изменения ориентации дуг пути и знаков длин дуг в графе появился контур (5,4,7,3,2,5) положительной длины. Заметим, что $c_{31} + c_{47} + c_{75} \leq 0$ так как первоначальный граф не имеет контуров положительной длины. Имеем

$$c_{23} + c_{34} + c_{47} + c_{73} + c_{45} \leq c_{23} + c_{45}$$

$$c_{43} + c_{73} + c_{25} > c_{23} + c_{45}$$

Следовательно,

$$c_{23} + c_{34} + c_{47} + c_{73} + c_{45} < c_{47} + c_{73} + c_{25}$$

$$c_{23} + c_{34} + c_{45} < c_{25}$$

и значит путь (I,2,3,4,5,6) не является путем максимальной длины. Таким образом, ориентированный граф, получаемый на каждом шаге алгоритма, не будет иметь контуров положительной длины, что обеспечивает получение оптимального решения на последнем шаге.

§ 4. Паросочетание произвольного графа

В § 2 мы определили понятие паросочетания W для произвольного графа и предложили формальную постановку задачи о максимальном паросочетании (4.I).

Для удобства изложения алгоритма поиска максимального паросочетания перепишем задачу (4.I) следующим образом: обозначим $x_{ij} = 1$, если ребро $[i,j] \in W$ и $x_{ij} = 0$, в противном случае. Тогда получим задачу:

$$\mathcal{L} = \sum_{(i,j)} x_{ij} c_{ij} \longrightarrow max \qquad (4.7)$$

$$\sum_{j=1}^{m} x_{ij} \leq 1, \quad i = 1, 2, \ldots, m$$

$$x_{ij} = 0,1$$

Заметим, что если $c_{ij} \leq 0$, то существует оптимальное решение, не содержащее ребра $[i,j]$. Поэтому можно положить в этом случае $c_{ij} = 0$. Аналогично, если $(i,j) \notin V$, то можно считать, что $[i,j] \in V$ и положить $c_{ij} = 0$ (в силу сказанного выше ребро $[i,j]$ не войдет в оптимальное решение). Поэтому можно считать, что граф полный и что длины всех дуг неотрицательны. Кроме того примем, что число вершин графа четно (в противном случае всегда можно добавить одну вершину с нулевыми длинами инцидентных ей дуг).

Доказательство условия оптимальности и описание алгоритма поиска максимального паросочетания проведем, используя понятия чередующихся цепей и циклов.

Чередующейся цепью (циклом) графа $G(X, V)$ называется простая цепь (цикл), в которой ребра принадлежат поочередно W и $V \setminus W$.

Длиной чередующейся цепи (цикла) называется сумма длин ребер цепи (цикла), причем длины ребер паросочетания берутся с обратным знаком.

Теорема 4.2. Необходимым и достаточным условием оптимальности паросочетания является отсутствие в графе чередующихся циклов положительной длины.

Необходимость очевидна. Действительно, если в графе имеется чередующийся цикл положительной длины, то положив $x'_{ij} = 1 - x_{ij}$ для ребер этого цикла, мы увеличим $\mathcal{L}$.

Достаточность. Пусть $\{x^0_{ij}\}$ оптимальное решение, $\{x'_{ij}\}$ некоторое решение, такое что в графе отсутствуют чередующиеся циклы положительной длины. Выделим все ребра (i,j), такие что $x^0_{ij} \neq x'_{ij}$. Очевидно, что множество таких ребер образует набор чередующихся циклов для решения $\{x_{ij}'\}$. Положив для ребер, принадлежащих этим циклам $x_{ij} = 1 - x'_{ij}$, мы очевидно получим решение $\{x^0_{ij}\}$. При этом сумма длин дуг нового паросочетания не увеличится в силу отсутствия чередующихся циклов положительной длины. Поэтому $\{x'_{ij}\}$ – оптимальное решение.

4.I. Описание алгоритма

I шаг. Определяем произвольное паросочетание $\{x_{ij}\}$.

2 шаг. Определяем чередующиеся циклы положительной

длины. Если таких циклов нет, то начальное паросочетание оптимально. В противном случае производим преобразование $x'_{ij} = 1 - x_{ij}$ для ребер чередующегося цикла положительной длины. В результате получаем новое паросочетание с большей суммой длин ребер. Продолжаем таким образом, пока не убедимся в отсутствии чередующихся циклов положительной длины.

Пример 4.2. Применим алгоритм для полного 6-вершинного графа. Данные о длинах дуг приведены ниже

i \ j	I	2	3	4	5	6
I	x	4	3	9	2	5
2	4	x	3	6	6	2
3	3	3	x	7	I	8
4	9	6	7	x	8	4
5	2	6	I	8	x	7
6	5	2	8	4	7	x

I шаг. Возьмем начальное решение $x^0_{14} = 1,\ x^0_{23} = 1,\ x^0_{56} = 1$ остальные $x^0_{ij} = 0$,

$\mathcal{L}_0 = 19$.

2 шаг. Проверим сначала наличие чередующихся циклов положительной длины в подграфе, образованном вершинами I,4, 5,6. Возможны всего два чередующихся цикла

$\mu_1 = (I,4,5,6,I)$, его длина $\mathcal{L}(\mu_1) = -3 < 0$

$\mu_2 = (I,5,6,4,I)$, его длина $\mathcal{L}(\mu_2) = -10 < 0$

Следовательно, чередующийся цикл положительной длины (если он существует) обязательно содержит ребро $[2,3]$. Сведем задачу определения чередующегося цикла положительной длины к задаче определения контура положительной длины. Для

этого заменим пару ребер $[i,j]$ и $[j,k]$ таких, что $x^0_{ij}=0$, $x^0_{jk}=1$ одной дугой (i,k) длины $c'_{ik}=c_{ij}-c_{ik}$. Примем вершину 2 за начальную (поэтому, вершину 3 можно не рассматривать). Преобразованный граф с длинами c^1_{ij} изображен на рис.4.5 (длина дуги (i,j) указана у конца j той дуги).

Определяем контур максимальной длины, проходящий через вершину 2. Установившиеся индексы вершин указаны в квадратных скобках. Контур максимальной длины $(2,6,2)$ имеет длину $4>0$. Ему соответствует чередующийся цикл $[2,5,6,3,2]$ той же длины. Изменяя значения x^0_{ij} по циклу, получаем

$$x'_{25}=1,\quad x'_{03}=1,\quad x'_{14}=1,\ \text{остальные}\ x'_{ij}=0,$$

$$\mathcal{L}_1=6+8+9=23.$$

3 шаг. Проверьте самостоятельно,что чередующихся циклов положительной длины в подграфе, образованном вершинами 1,4,2,5 нет. Следовательно, если чередующиеся циклы положительной длины имеются, то они содержат ребро $[6,3]$. Примем вершину 3 за начальную и построим преобразованный граф (рис.4.6). Контуров положительной длины нет. Поэтому решение

$$x_{25}=1,\quad x_{03}=1,\quad x_{14}=1,\ \text{остальные}\quad x_{ij}=0$$

является оптимальным.

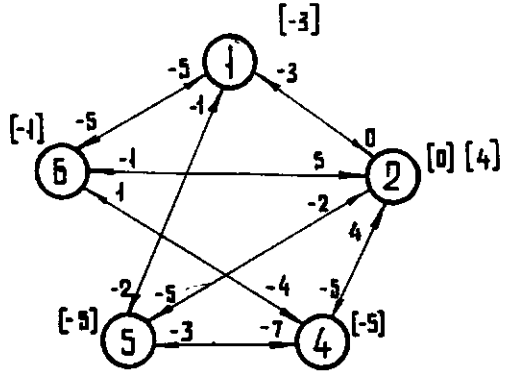

[-3]
1
-5
-3
-1
[-1]
-5
0
6
-1
5
2
[0] [4]
1
-2
4
-2
-5
-4
-5
[-5]
5
-3
-7
4

Рис. 4.5

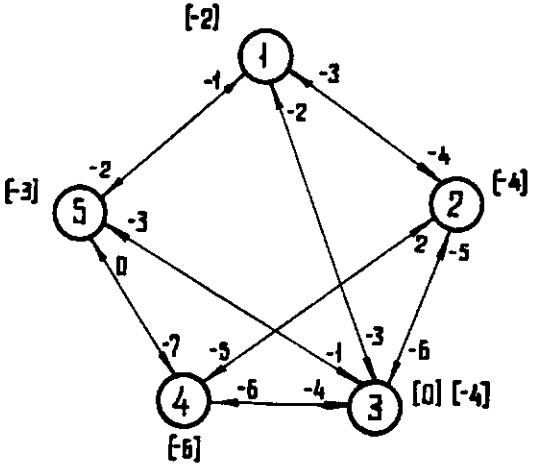

[-2]
1
-1
-3
-2
-4
[-4]
-2
[-3]
5
2
-3
2
-5
0
-7
-5
-1
-3
-6
4
-6
-4
3
[0] [-4]
[-6]

Рис. 4.6

§ 5. Реберное покрытие графа

Интересный класс комбинаторных задач на графах составляют так называемые задачи о покрытии [2].

Задача о покрытии с минимальным взвешенным числом ребер, возникающая в задачах доставки грузов, проектировании схем и пр., формулируется аналогично задаче максимального паросочетания (4.I) за исключением того, что " max " заменяется на " min " и знак неравенства меняет направление:

$$\sum_{V} C(V)\ Z(V) \to min \qquad (4.8)$$

при условиях $Z(i,x-i) \geqslant 1,\ i \in X,\ Z(V)=0,1\ \ V \in V$

Теоремы, подобные доказанным выше относительно паросочетаний, могут быть сформулированы для реберного покрытия т.к. эти задачи по существу связаны между собой.

Так, например, в простом случае $C(V) = I$ из данного максимального паросочетания W^* минимальное покрытие D^* находится добавлением в W^* ребра в каждой вершине, которая не покрыта; и наоборот. Из данного минимального покрытия D^* максимальное паросочетание W^* находится выбрасыванием ребер инцидентных одной вершине, кроме одного.

§ 6. Разбиение вершин графа

6.I. Внутреннее устойчивое множество вершин

Множество вершин $A \subset X$ графа $G(X,V)$ называется внутренне устойчивым (или независимым), если никакие две вершины в A не смежны. В частности, единственная вершина образует внутренее устойчивое множество.

Часто встречаются задачи нахождения наибольшего внутренне устойчивого множества вершин в данном графе, максимальное число вершин в A при этом называется числом внутренней устойчивости (вершиной независимости).

Задача 4.2. Необходимо сформировать комиссию из экспертов, определенные пары которых не могут быть вместе назначены в комиссию.

Задача 4.2 относится к упомянутому выше типу, в которой вершины графа X соответствуют различным экспертам, а ребра связывают несовместимых лиц.

Считая как и раньше, что граф $G(X,V)$ связный сформулируем задачу определения максимального взвешенного внутренне устойчивого множества, как задачу целочисленного программирования:

$$\sum_{i \in X} b(i)\, \mathcal{A}(i) \to \max \qquad (4.9)$$

при условиях

$$\mathcal{A}(i) + \mathcal{A}(j) \le 1, \quad (i,j) \in V \quad \mathcal{A}(i) = 0,1, \quad i \in X,$$

где $b(i)$ вес приписанный вершине $i \in X$,

$\mathcal{A}(i) = 1$, если $i \in A$ и $\mathcal{A}(i) = 0$ в противном случае.

Если в экстремальных задачах разбиения ребер (максимальном паросочетании, минимальном реберном покрытии) получены сравнительно простые необходимые и достаточные условия оптимальности для разработки "хороших" комбинаторных алгоритмов, то для экстремальных задач разбиения вершин (максимальное внутренне устойчивое множество, минимальное внешне устойчивое множество и вершинное покрытие графа, понятия определяемые ниже) не найдено аналогичных условий оптимальности.

Задачу (4.9) можно, естественно, решать используя известные методы целочисленного линейного программирования, в том числе метод "ветвей и границ" (см. приложение I). Однако, в случае, когда $b(i)=1$, $i \in X$ можно воспользоваться простой процедурой для нахождения наибольшего внутренне устойчивого множества, в общем случае не дающей оптимального решения.

I^{0}. Начинать решение с вершины $x_i \in X$ с наименьшей степенью;

2^{0}. на каждом шаге выбирать в качестве следующей вершины включаемой в A вершину, которая смежна с наименьшим числом вершин не включенных в A .

Упражнение 4.I. Постройте граф, для которого описанная процедура не дает наибольшего внутренне устойчивого множества.

Рассмотрим метод, позволяющий находить наибольшее внутренне устойчивое множество, когда граф $G(X, V)$ является деревом.

Тогда нетрудно показать, что при $n>2$ всегда существует внутренне устойчивое множество с максимальным числом вершин, содержащее все висячие вершины дерева. Действительно, пусть внутренне устойчивое множество с максимальным числом вершин не содержит некоторой висячей вершины i. Тогда, очевидно это множество содержит вершину j, смежную с i. Исключив вершину j, мы сможем включить в независимое множество вершину i. При этом, число вершин внутренне устойчивого множества не изменится. Отмеченное свойство позволяет быстро определять число внутренней устойчивости для дерева.

1 шаг. Помечаем все висячие вершины (например, обводим их).

2 шаг. Уделяем все висячие вершины и все смежные с ними. Затем повторяем шаги I и 2. Например, для дерева, показанного на рис.4.7, число вершинной независимости равно 8 (вершины соответствующего независимого множества обведены).

Упражнение 4.2. Определите число вершинной независимости для цикла, состоящего из K вершин.

6.2. Оценка сверху числа внутренней устойчивости

Определим граф G', обладающий следующим свойством: если в графе G две вершины смежны, то в G' они не смежны, и наоборот. Граф G' называется дополнительным для графа G. Заметим, что любому внутренне устойчивому множеству в графе G соответствует полный подграф в графе G'. Обозначим z_i' – степень вершины i в

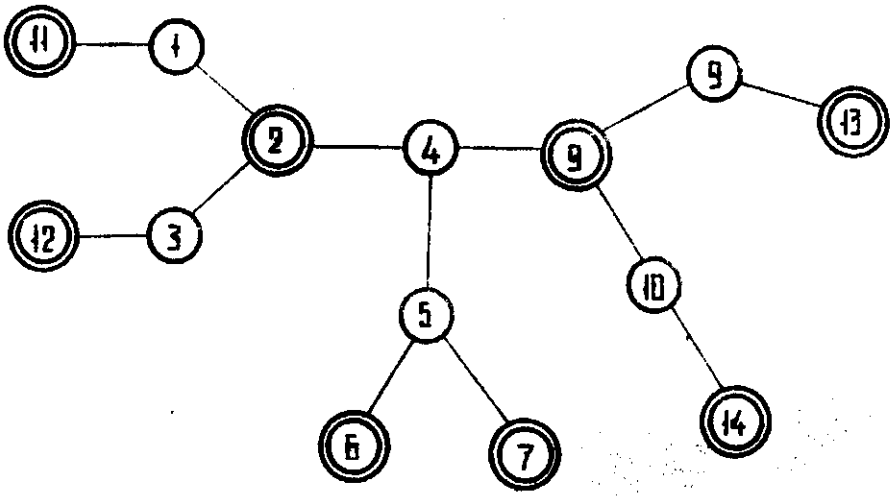


Рис. 4.7

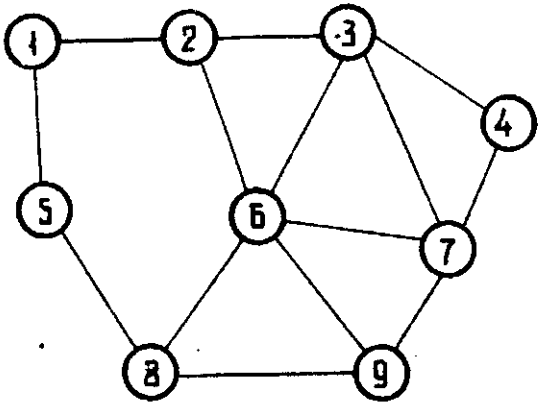


Рис. 4.8.

графе G' , m' – число ребер графа G' . Очевидно

$$r_i' = n-1-r_i \qquad 2m' = n(n-1)-2m . \tag{4.10}$$

Пусть $r_1' \geq r_2' \geq \dots \geq r_n'$

Если число внутренней устойчивости равно β , то справедливо неравенство

$$\beta(\beta-1) + \sum_{i=1}^{\beta} r_i' \leq 2m \tag{4.11}$$

Подставляя выражения r_i' и $2m'$ из (4.10) и преобразуя неравенство, получаем

$$\beta^2 \leq \sum_{i=1}^{\beta} (n - r_i) \tag{4.12}$$

Неравенство (4.12) позволяет оценить сверху число внутренней устойчивости. Граф называется k – связным, если k есть минимальное число ребер, после удаления которых граф становится несвязным. Для k – связного графа оценку (4.12) можно улучшить

$$\beta^2 \leq \sum_{i=1}^{\beta} (n - r_i) - k \tag{4.13}$$

6.3. Применение метода ветвей и границ

Рассмотрим некоторую вершину a графа и разобьем множество всех решений (под решением понимается любое независимое множество вершин) на два подмножества (см.приложение 1). В первом – все внутренне устойчивые множества содержат вершину a , а во втором – не содержат. По формуле (4.12) или (4.13) можно оценить сверху число внутренней устойчивости для каждого подмножества. Выбираем под-

множество с большей оценкой и продолжаем процедуру разбиения. Рассмотрим более подробно применение метода на примере.

Пример 4.2. Определим число внутренней устойчивости для графа, изображенного на рис.4.8. Выберем вершину 3 для разбиения множества всех решений на подмножества. Для удобства будем обозначать $x_i = 1$, если вершина i включена во внутренне устойчивое множество и $x_i = 0$, в противном случае.

I. Пусть $x_3 = 1$. Тогда $x_2 = x_4 = x_6 = x_7 = 0$
Оставшийся подграф, образованный вершинами 1,5,8,9 является деревом, и согласно результатам пункта 6.1, получаем
$x_1 = x_9 = 1$, $x_5 = x_8 = 0$, $\beta = 3$

II. Пусть $x_3 = 0$. Поскольку вершина 4 является висячей, то $x_4 = 1$, $x_7 = 0$. Оставшийся подграф двусвязен и состоит из 6 вершин. Из (4.13) получаем $\beta \leq 4$, так как при $\beta = 5$ неравенство нарушается. Выбираем подмножество с большей оценкой, то есть полагаем $x_3 = 0$, $x_7 = 0$, $x_4 = 1$. Выберем для последующего разбиения вершину 6.

I. Пусть $x_6 = 1$. Тогда сразу получаем
$$x_2 = x_7 = x_8 = x_9 = 0, \quad x_1 = 1, \quad x_5 = 0, \quad \beta = 3$$

II. Пусть $x_6 = 0$. Тогда $x_2 = x_5 = x_9 = 1$, $x_1 = x_8 = 0$, $\beta = 4$
Окончательно получаем внутренне устойчивое множество, состоящее из вершин 2,4,5,9. Число его вершин $\beta = 4$ определяет число внутренней устойчивости, так как оценки остальных подмножеств меньше 4. Дерево ветвлений показано

на рис.4.9. (оценка подмножеств указаны в соответствующих вершинах).

Упражнение 4.2. ~~Максимальное число ребер, которые образуют паросочетание, называется числом реберной независимости.~~ Докажите, что число ребер~~ной независимости~~ любого покрытия не меньше числа внутренней устойчивости.

§ 7. Внешне устойчивое множество вершин

Максимальное внутренне устойчивое множество A^* обладает свойством доминирования в графе $G(X,V)$ в том смысле, что каждая вершина, не принадлежащая A^* смежна, смежна с некоторой вершиной в A^*. Если внутренне устойчивое множество A^* не доминирует в графе, то существует по крайней мере одна вершина, не принадлежащая A^* и не смежная с ним, которую можно включить в это множество. Множество вершин, обладающих свойством доминирования, называется внешне устойчивым множеством B, вне зависимости от того является ли оно внутренне устойчивым или нет. Другими словами внешне устойчивым называется множество вершин ВСХ, если каждая не принадлежащая В вершина является конечной вершиной некоторого ребра, другая конечная вершина которого принадлежит В.

Задача, которую мы будем рассматривать далее, состоит в построении внешне устойчивого множества с наименьшим числом элементов.

Упражнение 4.3. Докажите самостоятельно, что число вершин в наибольшем внутренне устойчивом множестве, по крайней мере, не меньше, чем число вершин в наименьшем внешне

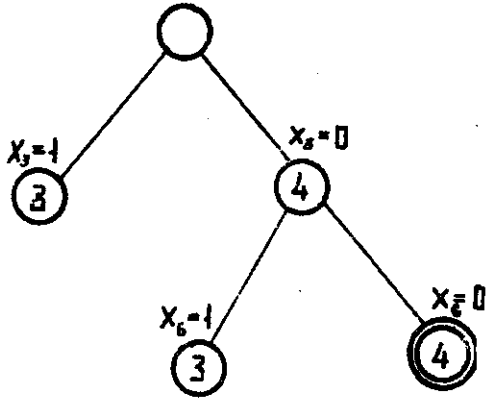


Рис. 4.9

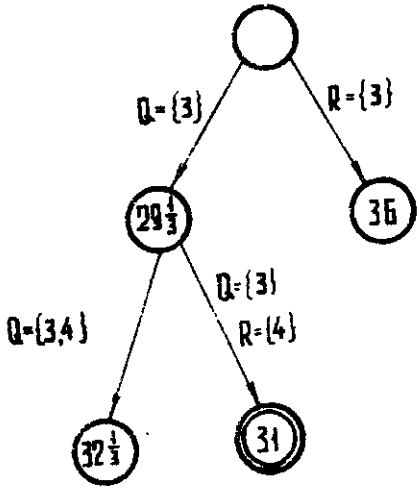


Рис. 4.10

устойчивом множестве.

Задачи нахождения наибольшего внутренне устойчивого множества и наименьшего внешне устойчивого множества, как уже отмечалось, относятся к определенному классу комбинаторных задач [6], для решения которых в общем случае, нет "регулярных" методов. Предлагаются различные комбинаторные методы решения и в том числе эвристические алгоритмы, которые были описаны выше для нахождения наибольшего внутренне устойчивого множества.

Один из алгоритмов нахождения наименьшего внешне устойчивого множества описан К.Бержом [2].

Рассмотрим связный граф $G=(X,V)$, в котором $X=\{a,b,\dots,q\}$. Поставим в соответствие G новый простой или двудольный граф, который обозначим через $G'(X,\bar{X},\bar{V})$, где $\bar{X}=\{\bar{a},\bar{b},\dots,\bar{q}\}$, а ребра $\bar{U}$ графа G' соответствуют ребрам графа G.

Задача нахождения наименьшего внутренне устойчивого множества B^* в графе G свелась тем самым к нахождению B^* в $X=\{a,b,\dots,q\}$ по отношению к $\bar{X}=\{\bar{a},\bar{b},\dots,\bar{q}\}$.

1°. Удаляем из графа G' каждую вершину $x\in X$, связанную ребром с единственной $\bar{x}\in\bar{X}$.

2°. Если в простом графе G' имеется висячее ребро $(x,\bar{y})$, то очевидно, $x\in B^*$.

3°. Исключаем из простого графа G' вершины уже вошедшие в B^*. Если упростить граф уже нельзя, то назовем его неприводимым.

4°. Относим в B^* произвольную вершину не вошедшую в B^* раньше.

5°. Возвращаемся к пункту I°, чтобы произвести упрощения графа G'.

Упражнение 4.4. Придумать граф G, для которого описанный алгоритм не дает наименьшего внешне устойчивого множества.

К задаче определения наименьшего внешне устойчивого множества сводятся многие прикладные задачи, в том числе следующая упрощенная задача размещения объектов (завод, склад и т.п.).

Задача 4.3. Дан неориентированный граф $G(X,V)$ представляющий собой транспортную сеть, вершинами которого являются города, в которых потребляется определенная продукция. Требуется построить минимальное число заводов в городах, производящих эту продукцию и снабжающих ей всех остальных потребителей по сети (движение по ребрам и сети G в двух направлениях).

Используя вышеприведенный алгоритм о минимальном внешне устойчивом множестве, можно получить решение этой задачи.

Более сложная задача размещения объектов (источников снабжения), которую можно назвать задачей о минимальном взвешенном внешне устойчивом множестве, будет рассмотрена в конце главы этой главы. А сейчас перейдем к обсуждению задачи покрытия ребер вершинами и связью ее с внутренне устойчивым множеством и паросочетанием.

§ 8. Вершинное покрытие графа

Вершинным покрытием графа $G(X,V)$ называется мно-

жество вершин ССХ графа такое, что каждое ребро графа инцидентно, по крайней мере, с одной вершиной в C .

Задача состоит в поиске покрытия C с минимальным числом вершин.

Задача минимального взвешенного вершинного покрытия (каждой вершине приписан вес $b(i)$) может быть по аналогии с (4.9) сформулирована как:

$$|\mathcal{X}| = \sum_{x} b(i)\mathcal{X}(i) \longrightarrow \min \qquad (4.14)$$

при условиях $\mathcal{X}(i) + \mathcal{X}(j) \geq 1, \quad (i,j) \in V$

$$\mathcal{X}(i) = 0,1 \quad i \in X$$

Нетрудно заметить связь между минимальным вершинным покрытием и максимальным внутренне устойчивым множеством.

Если C^* минимальное вершинное покрытие графа $G(X,V)$ то $X \setminus C^*$, дополняющее множество вершин, является максимальным внутренне устойчивым множеством A^*. И, наоборот, если A^* максимальное внутренне устойчивое множество, то $X \setminus A^*$, является минимальным вершинным покрытием C^* (докажите это утверждение самостоятельно).

Значительно более сложной и интересной является связь между максимальным паросочетанием и минимальным вершинным покрытием, выделенная Балинским.

Запишем двойственную задачу к (4.14)

$$|x| = \sum x(v) \longrightarrow \max$$
$$x(i,u) \leq b(i), \quad i \in N \qquad (4.15)$$
$$x(u) \geq 0 \text{ целые}, \quad v \in V$$

где $b(i)$ целые числа приписанные каждой вершине $i \in N$.

Задача (4.I5) состоит в определении паросочетания $x \geq 0$ как множество целых неотрицательных чисел $x(V)$, $u \in V$.

Из теории двойственности в линейном программировании следует, что $|X| \leq |\mathcal{T}|$ для любого паросочетания x и покрытия Π. Принцип оптимальности формулируется следующим образом:

Теорема 4.3. Для данного минимального покрытия $\mathcal{T}^*$ существует максимальное паросочетание X^*, когда выполняется условие

$$\mathcal{T}^*(i) = 1 \Rightarrow X^*(i, N) = b(i)$$

Рассматривая (4.I4) и (4.I5) как обычные задачи линейного программирования (отказавшаяся от целочисленности) оптимальные решения удовлетворяют соотношением:

$$\mathcal{T}^*(i)[x^*(i, N) - b(i)] = 0 \quad \text{и} \quad x^*(V)[\mathcal{T}^*(i) + \mathcal{T}^*(j) - 1] = 0,$$

где $V = (i, j)$ для всех $(i, j) \in V$

Следствие. Существуют максимальное паросочетание x^* и минимальное покрытие $\mathcal{T}^*$, удовлетворяющие:

$$|x^*| = |\mathcal{T}^*| - |x^*(C^*, C^*)|$$

где $C^* = \{i \in N, \mathcal{T}^*(i) = 1\}$

Доказательство следствия следует непосредственно из теоремы 4.3.

К сожалению невозможно обобщить теорему 4.3 в том направлении, что нельзя утверждать, что для данного максимального паросочетания X^* существует минимальное покрытие $\mathcal{T}^*$, когда выполняется либо условие $\mathcal{T}^*(i)[x^*(i, N) - b(i)] = 0$, $i \in N$, либо $x^*(V)[\mathcal{T}^*(i) + \mathcal{T}^*(j) - 1] = 0$, $(i, j) = v \in V$.

Этого нельзя утверждать даже для случая, когда $\delta(i)=1$, $i \in X$.

Таким образом, теорема 4.3 лишний раз подтверждает, что задача на покрытие более сложная, чем задача паросочетания.

§ 9. Размещение источников снабжения

9.1. Постановка задачи

Задачи размещения источников снабжения (заводов, баз, магазинов) является дальнейшим обобщением понятия внешне устойчивого множества и составляют важный класс практической задачи. Мы рассмотрим одну из наиболее известных постановок. Пусть имеется m возможных пунктов (вершин) размещения источников снабжения (будем называть их в дальнейшем базами) и n пунктов потребления. В каждом пункте потребления задана величина потребности B_j в продукции (ограничимся рассмотрением однопродуктовой задачи). Затраты, связанные с размещением и функционированием базы в i -ом пункте равны

$$\alpha_i + c_i x_i ,$$

где α_i - постоянная часть затрат, не зависящая от количества продукции x_i , поставляемого с базы, c_i - затраты на хранение единицы продукции на базе. Затраты на доставку единицы продукции с i -ой базы j -ому потребителю обозначим C_{ij} . Пусть F - множество пунктов, в которых размещены базы. Очевидно, что потребителя j целесообразно снабжать с той базы, для которой величина $c_i + C_{ij}$ минимальна. Поскольку ограничений на емкость баз мы не накладываем, то всегда существует вариант снабжения, при ко-

тором каждый потребитель снабжается с одной базы. Поэтому, общие минимальные затраты на снабжение при размещении баз в пунктах множества F равны

$$S = \sum_{i \in F} d_i + \sum_{j=1}^{n} B_j \min_{i \in F} (C_i + C_{ij}) \qquad (4.16)$$

Обозначив $B_j (C_i + C_{ij}) = S_{ij}$, получаем более простую запись

$$S = \sum_{i \in F} d_i + \sum_{j=1}^{n} \min_{i \in F} S_{ij} \qquad (4.17)$$

Задача 4.4. Определить множество F пунктов размещения баз, при котором (4.17) достигает минимальной величины.

Несмотря на простоту формулировки, задача 4.17 является довольно сложной экстремальной задачей комбинаторного типа.

Упражнение 4.5. Решите задачу 4.17 для случая

$$S_{ij} = S_i , \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, n$$

3.2. Метод ветвей и границ

Обозначим Q - множество пунктов, относительно которых уже принято решение разместить базы, R - множество пунктов, относительно которых уже принято решение не размещать базы и P - множество пунктов, относительно которых решения еще не принято.

Очевидно $P \cup Q \cup R = \{1, 2, \ldots, m\}$, $P \cap Q \cap R = \emptyset$

Задание Q и R определяет некоторое подмножество (Q, R) решений. $Q = R = \Phi$ соответствует всему множеству решений, $Q \cup R = \{1, 2, \ldots, m\}$ определяет некоторое допустимое размещение баз. Процедура разбиения любого подмножестве ре-

шений (Q,P) определяется следующим образом. Берем некоторый пункт $i \in P$ и относим его либо к Q (то есть в пункте i размещается база), либо к P (в пункте i база не размещается). Таким образом, подмножество (Q,P) разбивается на два подмножества $(Q \cup i, P)$ и $(Q, P \cup i)$. Для применения метода ветвей и границ нам осталось дать способ оценки (снизу любого подмножества (Q,P)). Заметим, во-первых, что если для двух пунктов $i \in Q$ и $k \in P$ имеет место

$$S_{ij} \leq S_{kj} \tag{4.18}$$

для некоторого j, то можно запретить снабжение потребителя j из пункта k (действительно, в пункте i база уже имеется, и в то же время затраты на снабжение потребителя j из пункта i не больше, чем из пункта k). Аналогично, если для двух пунктов $i, k \in P$ имеет место

$$S_{ij} + d_i \leq S_{kj} \tag{4.19}$$

для некоторого j, то можно запретить снабжение потребителя j из пункта k (объясните, почему?). Исключим все запрещенные связи, положив соответствующие затраты $S_{kj} = \infty$. Если при этом найдется потребитель, снабжение которого возможно только из пункта $i \in P$, то очевидно, в этом пункте следует разместить базу (то есть следует i включить в Q). После этого снова проверяем условия (4.18), исключаем связи, которые можно запретить и т.д. В результате описанной процедуры будет получена некоторая матрица возможных связей (S_{ij}). Обозначим m_i - число пунктов потребления, которые могут снабжать-

ся из пункта i (при наличии там базы). Из очевидной оценки справедливой для любого подмножества A потребителей

$$d_i + \sum_{j \in A} S_{ij} \geq \sum_{j \in A} \left(\frac{d_i}{m_i} + S_{ij} \right), \quad i \in P,$$

получаем следующую оценку $\Psi(Q,R)$ подмножества (Q,R)

$$\Psi(Q,R) = \sum_{i \in Q} d_i + \sum_{j=1}^{n} \min \left[\min_{i \in Q} S_{ij} ; \min_{i \in P} \left(\frac{d_i}{m_i} + S_{ij} \right) \right] \qquad (4.20)$$

9.3. Решение примера. Рассмотрим задачу размещения баз в 4 возможных пунктах для снабжения восьми потребителей. Данные о затратах S_{ij} и d_i приведены ниже

i	Потребители								d_i
	I	2	3	4	5	6	7	8	
I	2	5	7	4	6	3	I	7	5
2	3	5	I	9	9	2	7	6	6
3	5	2	4	3	I	8	6	3	7
4	7	8	9	4	5	6	4	2	4

Сначала проверяем условие (4.I9). Для этого определяем

$$a_j = \min_i (d_i + S_{ij}), \quad j = 1, 2, \ldots, 8$$

Имеем

j	I	2	3	4	5	6	7	8
a_i	7	9	7	8	8	8	6	6

Полагаем равными ∞ все $S_{ij} \geq a_j$

$S_{01}, S_{13}, S_{43}, S_{21}, S_{25}, S_{36}, S_{24}, S_{37}, S_{48}, S_{28}$

Получаем следующую таблицу

i	Потребители								d_i
	I	2	3	4	5	6	7	8	
I	2	5	∞	4	6	3	I	∞	5
2	3	5	I	∞	∞	2	∞	∞	6
3	5	2	4	3	I	∞	∞	3	7
4	∞	8	∞	4	5	6	4	2	4

Возьмем в качестве пункта, по которому будем производить разбиение на подмножества, третий пункт.

I. $Q = \{3\}$, то есть в пункте 3 база размещается. Проверяя условия (4.18), полагаем равными ∞ все $S_{ij} \geq S_{3j}$. Получаем следующую таблицу.

i	Потребители								d_i	m_i	$\frac{d_i}{m_i}$
	I	2	3	4	5	6	7	8			
I	2	∞	∞	∞	∞	3	I	∞	5	3	I 2/3
2	3	∞	I	∞	∞	2	∞	∞	6	3	2
3	5	2	4	3	I	∞	∞	3	7	-	-
4	∞	∞	∞	∞	∞	6	4	2	4	3	I I/3

В последних столбцах таблицы приведены значения m_i и $\frac{d_i}{m_i}$ для $i \in P$. По формуле (4.20), получаем оценку подмножества

$$\varphi(\{3\}, \Phi) = 7 + 3\tfrac{2}{3} + 2 + 3 + 3 + 1 + 4 + 2\tfrac{2}{3} + 3 = 29\tfrac{1}{3}$$

II. $R = \{3\}$, то есть в пункте 3 база не размещается.

Исключая третью строку из таблицы, получаем

i	Потребители								d_i
	I	2	3	4	5	6	7	8	
I	2	5	∞	4	6	3	I	∞	5
2	3	5	I	∞	∞	2	∞	∞	6
4	∞	8	∞	4	5	6	4	2	4

В столбцах 3 и 8 только S_{23} и S_{48} меньше ∞ Поэтому размещаем базы в пунктах 2 и 4. непосредственной проверкой убеждаемся, что в пункте I базу размещать не следует. Таким образом, в подмножестве $Q = \Phi$, $R = \{3\}$ мы нашли наилучшее решение $Q = \{2,4\}$, $R = \{1,3\}$. Величина затрат в этом решении

$$S = 10+3+5+1+4+5+2+4+2 = 36.$$

Выбираем подмножество $Q = \{3\}$, $R = \Phi$ имеющее меньшую оценку $29\frac{1}{3}$. Для очередного разбиения выбранного подмножества возьмем пункт 4.

I. $Q = \{3,4\}$, $R = \Phi$ то есть в пункте 4 база также размещается. Проверка условий (4.18) показывает, что связей, которые могут быть запрещены, не возникает. Поэтому, переходим к получению оценки. Проверьте самостоятельно, что

$\varphi((3,4),\Phi) = 11+3\frac{2}{3}+2+3+3+1+4+2\frac{2}{3}+2 = 32\frac{1}{3}$.

II. $Q = \{3\}$, $R = \{4\}$ то есть в пункте 4 база не раз-

мещается. Исключая строку 4 из таблицы, замечаем, что в столбце 7 только $S_{11} < \infty$. Поэтому в пункте I база должна быть размещена. Непосредственной проверкой убеждаемся, что в пункте 2 базу размещать не следует. Получаем допустимое решение $Q = \{1,3\}$, $R = \{2,4\}$ с величиной затрат

$$S = 12+2+2+4+3+1+3+1+3 = 31 .$$

Полученное решение является оптимальным, так как оценки остальных подмножеств больше 31 (см.рис.4.10).

Глава У

ЗАДАЧИ РАСПРЕДЕЛЕНИЯ РЕСУРСОВ НА СЕТЯХ

§ I. Сетевая модель комплекса операций

I.I. Основные понятия

В этой главе рассматривается широкий класс задач построения календарных планов выполнения заданного множества работ (комплекса операций) ограниченным количеством ресурсов (людей, оборудования и т.д.). Общим для задач этого класса является наличие определенных ограничений на очередность выполнения работ. Наиболее типичным ограничением является так называемая зависимость. Две работы зависимы, если одну из них нельзя начинать, пока не закончена другая. Будем изображать работы вершинами графа и проводить дугу (i, j), если работу j нельзя начинать, пока не закончена работа i. Полученный граф называется сетевым графиком или просто сетью (не путайте с понятием транспортной сети). Сетевой график не может иметь контуров. Действительно, в противном случае работы, образующие контур, никогда не могли бы начаться. (Слова "работа" и "операция" означают одно и то же - некоторый процесс, требующий времени и ресурсов. Мы будем пользоваться обоими словами, не делая различий).

Для каждой работы задаются количественные оценки продолжительности и необходимых ресурсов. В простейшем случае задается только продолжительность τ_i каждой операции. Сетевой график с продолжительностями τ_i каждой операции образует простейшую сетевую модель комплекса операций. С помощью такой модели можно оценить продолжительность выполнения всех работ (при условии, что имеется достаточное

количество ресурсов). Продолжительность всего комплекса операций определяется путем, имеющим максимальную длину. Под длиной пути в данном случае понимается сумма продолжительностей операций пути. Алгоритмы определения путей максимальной длины, описанные в гл.2, нетрудно видоизменить для определения пути максимальной длины в данном случае. В дальнейшем примем, что выполнение комплекса начинается в момент $t = 0$.

Обозначим Q_i - множество операций, непосредственно предшествующих i -ой (если $j \in Q_i$, то $(j,i) \in U$). Положим $t_i' = \tau_i$ для любой вершины, являющейся входом сети, то есть не имеющей предшествующих. Вычислим последовательно

$$t_i^p = \tau_i + \max_{j \in Q_i} t_i^p \qquad (5.1)$$

Величина t_i^p называется ранним окончанием i -ой операции (докажите, что i -ая операция не может окончиться раньше чем через t_i^p с момента начала выполнения комплекса). Длина критического пути

$$T_{kp} = \max_i t_i^p$$

Важной характеристикой операции комплекса является длина максимального пути, начинающегося в соответствующей вершине. Обозначим эту длину l_i. Очевидно, для вершин, являющихся выходами сети, $l_i = \tau_i$. Обозначим R_i множество операций, непосредственно следующих за i -ой (если $j \in R_i$, то $(i,j) \in U$). Величина l_i последовательно определяется по формуле

$$l_i = \tau_i + \max_{j \in R_i} l_j. \qquad (5.2)$$

Величина $t_i^п = T_{кр} - l_i$ называется поздним началом i -ой операции.

Упражнение 5.I. Докажите, что для завершения комплекса операций за время $T_{кр}$ необходимо и достаточно, чтобы все операции начались не позже соответствующих моментов $t_i^п$. Разность $\Delta_i = t_i^п - t_i^р - \tau$ называется полным резервом i -ой операции.

Пример 5.I. Определим $t_i^р$, $t^п$ и Δ_i всех операций сети, показанной на рис.5.I.(продолжительности операций указаны в нижних секторах).

а) Определение $t_i^р$: $t_1^р = \tau_1 = 4$, $t_2^р = \tau_2 = 3$, $t_3^р = \tau_3 + t_2^р = 5$,

$t_4^р = \tau_4 + \max(t_1^р, t_2^р) = 10$, $t_5^р = \tau_5 + t_2^р = 6$, $t_6^р = \tau_6 + t_4^р = 11$,

$t_7^р = \tau_7 + \max(t_3^р, t_5^р) = 9$,

$T_{кр} = \max(t_6^р, t_7^р) = 11$.

б) Определение l_i:

$l_7 = 3$, $l_6 = 1$, $l_5 = \tau_5 + l_7 = 6$,

$l_4 = \tau_4 + l_6 = 6$, $l_3 = \tau_3 + \max(l_4, l_7) = 8$

; $l_2 = \tau_2 + \max(l_3, l_5) = 11$, $l_1 = \tau_1 + l_6 = 10$

Заметим, что $\max_i l_i = T_{кр} = 11$

в) Определение $t_i^п = T_{кр} - l_i$

i	I	2	3	4	5	6	7
$t_i^п$	I	0	3	5	5	I0	8

г) Определение Δ_i:

i	I	2	3	4	5	6	7
Δ_i	I	0	0	0	2	0	2

Операции, не имеющие резерва, называются критическими. Все критические операции лежат на критических путях (критический путь показан на рис.5.I. двойными дугами).

Замечание. Часто используется изображение операций в виде дуг сети. Вершины соответствуют окончанию одной или нескольких операций и называются событиями. При этом для отражения всех зависимостей между операциями иногда приходится вводить фиктивные операции (зависимости), изображаемые в виде пунктирных дуг. Так, на рис.5.2 показано изображение сети рис.5.I в виде "операции-дуги". Римские цифры у дуг указывают номера операций, числа в скобках равны продолжительностям операций. Критический путь также выделен двойными дугами.

I.2. Задача распределения ресурсов

Если для выполнения комплекса операций выделено ограниченное количество ресурсов, то возникает задача их наилучшего использования. Будем предполагать, если не оговорено особо, что каждая операция выполняется ресурсами только одного вида. Обозначим w_i - объем i-ой операции, $f_i(v_i)$ скорость i-ой операции при количестве ресурсов v_i. В дальнейшем предполагается, что $f_i(v_i)$ неубывающая непрерывная справа функция v_i, причем $f_i(0) = 0$. Если $v_i(t)$ - количество ресурсов на операции i в момент t, то момент t_i окончания операции определяется как минимальное t_i, удовлетворяющее уравнению

$$\int_0^{t_i} f_i[v_i(t)]\,dt = w_i \qquad (5.2)$$

Часто предполагается, что количество ресурсов v_i, назна-

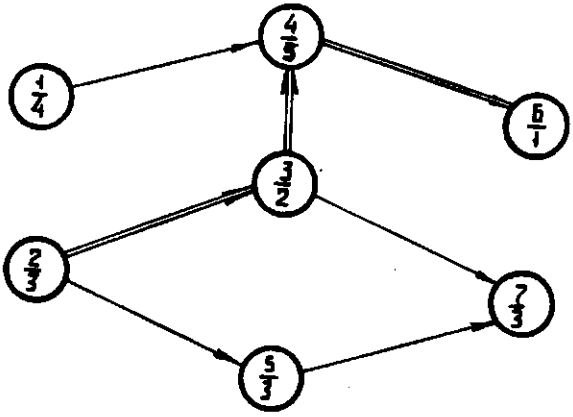


Рис. 5.1

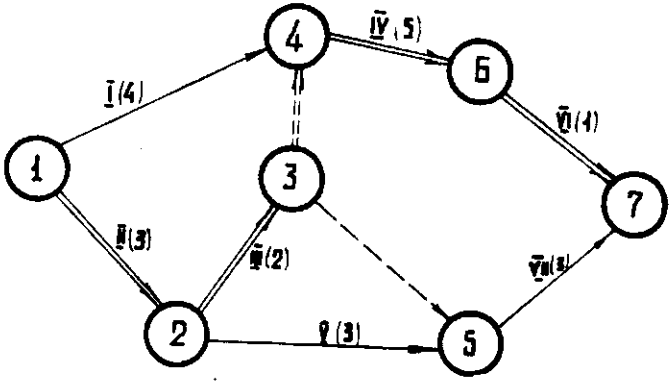


Рис. 5.2

ченное в начале операции, в дальнейшем не меняется (хотя операция может прерываться). В этом случае говорят, что операция выполняется с постоянной интенсивностью. Продолжительность операции определяется выражением

$$\tau_i(V_i) = \frac{w_i}{f_i(V_i)} \qquad (5.3)$$

Если количество ресурсов на операции может принимать только одно значение β_i, то говорят, что операция выполняется с фиксированной интенсивностью. В этом случае операция вполне определена заданием пары чисел: продолжительности τ_i и количества ресурсов β_i. Пусть имеются ресурсы m различных видов в количестве $N_1, N_2, \ldots, N_m$. Обозначим P_j - множество операций, выполняемых ресурсами j-го вида (согласно сделанному предположению каждая операция выполняется ресурсами только одного вида). Пусть, наконец, наша цель - закончить все операции комплекса за минимальное время. Сформулируем задачу оптимального распределения ресурсов.

Задача 5.I. Найти распределение ресурсов $V_i(t)$, $i = 1,2,\ldots,n$ удовлетворяющее ограничениям

$$\sum_{i \in P_j} V_i(t) \le N_j, \quad j = 1,2,\ldots,m \qquad (5.4)$$

так, чтобы время T выполнения комплекса операций было минимальным.

В настоящее время не существует алгоритмов, позволяющих получить оптимальное решение поставленной задачи в общем случае (хотя предложено много эвристических алгоритмов, позволяющих получить достаточно хорошие для практики решения

вразумные сроки). Однако, для ряда частных случаев, представляющих в то же время практический интерес, получены эффективные методы решения. Некоторые из них и рассматриваются в этой главе.

§ 2. Независимые операции

2.1. Выпуклые зависимости скоростей операций от количества ресурсов

Пусть все операции выполняются ресурсами одного вида, количество которых равно N, и $f_i(U_i)$ - выпуклые функции U_i, то есть

$$f_i[\alpha U_i' + (1-\alpha)U_i^2] \geq \alpha f_i(U_i') + (1-\alpha) f_i(U_i^2) \qquad (5.5)$$

для любых $U_i', U_i^2 \geq 0$ и любого $0 \leq \alpha \leq 1$

В этом случае имеет место следующий факт:

Теорема 5.1. Существует оптимальное решение, в котором каждая операция выполняется с постоянной интенсивностью.

Отметим основную идею доказательства. Пусть в некотором интервале длительности Δ, количество ресурсов на i-ой операции равно U_i', а в интервале длительности Δ_2 - U_i^2. Если положить в обоих интервалах количество ресурсов равным $\alpha U_i' + (1-\alpha)U_i^2$ для всех $i = 1, 2, \ldots, n$, где $\alpha = \frac{\Delta_1}{\Delta_1 + \Delta_2}$, то объем работ, который будет выполнен в обоих интервалах, не уменьшится согласно условию (5.1). Повторяя подобное преобразование, мы получим решение, в котором все операции выполняются с постоянной интенсивностью.

Теорема 5.2. Существует оптимальное решение, в котором все операции заканчиваются одновременно.

Эта теорема почти очевидна. Действительно, если в оптимальном решении i-я работа заканчивается раньше времени

завершения всех работ, то уменьшая количество ресурсов на ней, всегда можно увеличить время ее завершения до T. Будем искать решение, в котором все операции заканчиваются одновременно и выполняются с постоянной интенсивностью. Обозначим $V_i = \varphi(w_i)$ функцию, обратную функции $f_i(v_i) = w_i$. Для того, чтобы i-я операция была завершена за время T, необходимо и достаточно, чтобы

$$v_i \geq \varphi_i\left(\frac{w_i}{T}\right). \tag{5.6}$$

Суммируя по всем операциям, получаем

$$\sum_{i=1}^{n} \varphi_i\left(\frac{w_i}{T}\right) \leq N. \tag{5.7}$$

Минимальное T, удовлетворяющее неравенству (5.7), и определяет минимальный срок завершения комплекса.

Если имеется несколько видов ресурсов, то задача решается отдельно для каждого множества D_j операций, выполняемых ресурсами одного вида.

Замечание. Естественным обобщением рассмотренной задачи является случай, когда каждая операция выполняется ресурсами различных видов, но в заданном соотношении. Обозначим $d_{ij} V_i$ количество ресурсов j-го вида на i-ой операции, $\{d_{ij} V_i\}$, $j = 1, 2, \ldots, m$ образует набор ресурсов, $\{d_{ij}\}$ называются параметрами набора, а V_i - мощность набора.

Теорема 5.1. и 5.2. остаются справедливыми и для этого случая. Минимальное время выполнения комплекса определяется как минимальное T, удовлетворяющее системе неравенств

$$\sum_{i=1}^{n} a_{ij}\, g_i\left(\frac{w_i}{T}\right) \le N_j \quad j = 1, 2, \ldots, m. \tag{5.8}$$

Пример 5.2. Пусть $w_i = f_i(v_i) = v_i^{1/\alpha}$, $\alpha > 1$, $i = 1, 2, \ldots, n$.

Имеем $v_i = g_i(w_i) = w_i^{\alpha}$, $i = 1, 2, \ldots, n$

$$\sum_{i=1}^{n} a_{ij} \left(\frac{w_i}{T}\right)^{\alpha} \le N_j$$

$$T_{min} = \max_j \left[\frac{1}{N_j} \sum_{i=1}^{n} a_{ij}\, w_i^{\alpha}\right]^{1/\alpha}. \tag{5.9}$$

Пример 5.3. Пусть $w_i = f_i(v_i) = v_i$ при $0 \le v_i < \beta_i$

и $f_i(v_i) = \beta_i$ при $v_i \ge \beta_i$, $i = 1, \ldots, n$.

Имеем $T \ge \frac{w_i}{\beta_i}$, $i = 1, 2, \ldots, n$

и $\sum_{i=1}^{n} a_{ij} \frac{w_i}{T} \le N_j$, $j = 1, 2, \ldots, m$.

Следовательно,

$$T_{min} = \max\left[\max_i \frac{w_i}{\beta_i}, \max_j \frac{1}{N_j} \sum_{i=1}^{n} a_{ij}\, w_i\right]. \tag{5.10}$$

2.2. Невыпуклые зависимости

Если $f_i(v_i)$ не являются выпуклыми, то решение задачи существенно усложняется. Можно показать, что всегда существует оптимальное решение, состоящее из n интервалов постоянства (внутри этих интервалов распределение ресурсов не меняется).

Упражнение 5.2. Пусть $f_i(v_i)$ – выпуклые книзу функции, то есть

$$f_i(\alpha v_i' + (1-\alpha) v_i'') \le \alpha f_i(v_i') + (1-\alpha) f_i(v_i'') \tag{5.11}$$

для любых V_i^1, $V_i^2 \geq 0$, $0 < \alpha < 1$.

Докажите, что существует оптимальное решение, при котором операции выполняются поочередно, причем

$$T_{min} = \sum_{i=1}^{n} \frac{w_i}{f_i(N)} \qquad (5.12)$$

Обозначим $T_m(W)$ - минимальное время выполнения всех операций как функцию объемов операций.

Теорема 5.3. $T_m(\bar{W})$ - выпуклая (к низу) функция $\bar{W}$, то есть

$$T_m(\alpha \bar{W}^1 + (1-\alpha)\bar{W}^2) \leq \alpha T_m(\bar{W}^1) + (1-\alpha) T_m(\bar{W}^2), \quad 0 < \alpha < 1$$

Доказательство. Произведем замену переменных $w_i = f_i(V_i)$, $i = 1,2,\ldots,n$ и обозначим H множество допустимых $\bar{w}$. Множество H определяется неравенствами

$$\sum_{i=1}^{n} a_{ij} g_i(w_i) \leq N_j, \quad j = 1,2,\ldots,m.$$

Без ограничения общности можно считать H выпуклым множеством (в противном случае заменяем H его выпуклой оболочкой). В силу теорем 5.1 и 5.2. имеем

$$\bar{W}^1 T_m(\bar{W}^1) = \bar{W}^1, \quad \bar{W}^2 T_m(\bar{W}^2) = \bar{W}^2$$

Определим продолжительность T комплекса при

$$\bar{W} = \beta \bar{W}^1 + (1-\beta)\bar{W}^2 = \frac{\bar{W}}{T} = \frac{\alpha}{T} T_m(\bar{W}^1)\bar{W}^1 + \frac{1-\alpha}{T} T_m(\bar{W}^2)\bar{W}^2$$

Приравнивая коэффициент при $\bar{W}^1$ и $\bar{W}^2$, получаем

$$T = \alpha T_m(W^1) + (1-\alpha) T_m(W^2)$$

Поэтому $T_m(\bar{W}) \leq T = \alpha T_m(W^1) + (1-\alpha) T_m(W^2)$

2.3. Фиксированные интенсивности

Пусть каждая операция выполняется с фиксированной ин-

тенсивностью β_i за время τ_i, $i=1,2,\dots,n$. Обозначим $x_i=1$ если i-я операция выполняется и $x_i=0$, в противном случае. Тогда вектор $x=(x_1,x_2,\dots,x_n)$ определяет множество операций, которые можно выполнять одновременно, если

$$\sum_{i=1}^{n} x_i \beta_i \leq N . \qquad (5.13)$$

Предположим, что нам удалось выписать все различные векторы, удовлетворяющие условию (5.13). Расположим их в некотором порядке $x^1, x^2, \dots, x^k, \dots, x^s$ $(x^k=(x_1^k, x_2^k, \dots, x_n^k))$

Обозначим Δ_k - длительность интервала, в котором множество выполняемых работ описывается вектором x^k. Очевидно, что любому плану выполнения работ соответствует вполне определенный набор чисел $\Delta_k \geq 0$, удовлетворяющих условиям

$$\sum_{k=1}^{s} \Delta_k x_i^k = \tau_i, \quad i=1,2,\dots,n . \qquad (5.14)$$

Наоборот, любой набор $\{\Delta_i \geq 0\}$, удовлетворяющий условиям (5.14) определяет некоторый план выполнения работ. Поскольку время завершения комплекса

$$T=\sum_{k=1}^{s} \Delta_k \qquad (5.15)$$

то задача свелась к минимизации (5.15) при условиях (5.14). Это задача линейного программирования, однако, трудность заключается в том, что различных векторов, удовлетворяющих условиям (5.13) в общем случае достаточно много, и выписать все не представляется практически возможным.

Опишем алгоритм, позволяющий, начиная с некоторого допустимого решения, либо перейти к лучшему решению, либо до-

казать оптимальность данного. Применение алгорима рассмотрим на примере конкретных данных, приведенных ниже.

i	I	2	3	4
τ_i	I2	I5	I0	20
β_i	I7	I2	8	I0

I этап. Получение допустимого решения. Пусть N = 30. Возьмем произвольное множество работ, которые можно выполнять одновременно, например, работы I и 2. Соответствующий вектор $X^1 = (I,I,0,0)$. Максимальная длительность интервала, в котором работы I и 2 могут выполняться одновременно, равна очевидно

$$\Delta_1 = \min(\tau_1, \tau_2) = 12$$

Через Δ_1 =I2 останутся не выполненными работы 2,3 и 4.

i	2	3	4
τ_i'	3	I0	20
β_i	I2	8	I0

Все три работы можно вести одновременно, так как $\beta_2 + \beta_3 + \beta_4 = 30$. Получаем $X^2 = (0,I,I,I)$, $\Delta_2 = 3$.

Далее, аналогично, $X^3 = (0,0,I,I)$, $\Delta_3 = 7$

и $X^4 = (0,0,0,I)$, $\Delta_4 = I0$.

Получили допустимое решение с временем выполнения комплекса T = 32.

Замечание. Для дальнейшего важно, чтобы число полученных векторов было равно числу работ (четырем в нашем примере). Поэтому, если получено допустимое решение с числом векторов меньше 4, следует добавить недостающее число векторов, взяв произвольные допустимые и положив соответствую-

щие Δ равными 0. Целесообразно в этом случае брать простые вектора, у которых всего одна координата ненулевая (например, (0,0,0,I), (0,I,0,0) и т.д.).

II этап. Улучшение полученного решения. Обозначим y_i вектор, у которого только i -я координата равна I:

$y^1 = (I,0,0,0)$, $y^2 = (0,I,0,0)$, $y^3 = (0,0,I,0)$, $y^4 = (0,0,0,I)$.

Наша первая задача выразить каждый вектор y^i через вектора x^1, x^2, x^3, x^4. Это сделать нетрудно. Действительно,

$$y^4 = x^4$$

Так как $x^3 = (0,0,I,I) = y^3 + y^4$, то $y^3 = x^3 - y^4 = x^3 - x^4$

Далее $x^2 = (0,I,I,I) = y^2 + y^3 + y^4$ и значит

$$y^2 = x^2 - y^3 - y^4 = x^2 - (x^3 - x^4) - x^4 = x^2 - x^3$$

Наконец, $x^1 = (I,I,0,0) = y^1 + y^2$ и поэтому

$$y^1 = x^1 - y^2 = x^1 + x^2 + x^3$$

В общем случае мы получим

$$y^i = \sum_{j=1}^{n} c_{ij} x^j \qquad (5.I6)$$

Вычислим теперь $a_i = \sum_{j=1}^{n} c_{ij}$ (a_i равно сумме коэффициентов разложения вектора y^i по векторам x^1, x^2, x^3, x^4). Имеем $a_1 = 1, a_2 = 0, a_3 = 0, a_4 = 1$. Сформулируем следующую задачу: Определить $x_i = 0$ или I, удовлетворяющие условию

$$\sum_{i=1}^{n} x_i \beta_i \leq N \qquad (5.I7)$$

и максимизирующие линейную форму

$$A = \sum_{i=1}^{n} a_i x_i \qquad (5.I8)$$

Теорема 5.4. Если $A \leq 1$, то решение $(x^1, x^2, \ldots, x^n)$ является оптимальным.

Задача (5.I7), (5.I8) является задачей о ранце. Методы ее решения рассмотрены в приложении. Для нашего примера задача о ранце имеет вид:

$$1 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 1 \cdot x_4 \rightarrow max$$

при условии

$$17x_1 + 12x_2 + 8x_3 + 10x_4 \leq 30.$$

Ее решение очевидно $x^5 = (1,0,0,1)$, $A = 2 > 1$. Так как $A > 1$, то начальное решение можно улучшить. Заметим, что

$$x^5 = (1,0,0,1) = y^1 + y^4 = x^1 - x^2 + x^3 + x^4 \qquad (5.I9)$$

Если ввести в решение интервал длительности $\Delta_5 > 0$, в котором выполняются операции 1 и 4, то для сохранения ограничений (5.I4) необходимо (см.формулу 5.I9) уменьшить на Δ_5 длительности Δ_1 , Δ_3 , и Δ_4 и увеличить на Δ_5 длительность Δ_2 . Отсюда получаем, что максимальное значение $\Delta_5 = 7$. При этом, вектор x^3 исключается из решения, так как $\Delta_3^1 = 0$. Получаем новое решение (x^1, x^2, x^4, x^5) с длительностями интервалов

$$\Delta_1^1 = \Delta_1 - 7 = 5, \quad \Delta_2^1 = \Delta_2 + 7 = 10, \quad \Delta_4^1 = \Delta_4 - 7 = 3, \quad \Delta_5^1 = \Delta_5 = 7$$

Проверьте самостоятельно, что все ограничения (5.I4) выполняются в новом решении. Для полученного решения следует повторить описанную процедуру. При этом, для выражения векторов y^1, y^2, y^3, y^4 через вектора x^1, x^2, x^4, x^5 достаточно в полученные ранее разложения векторов y^1, y^2, y^3, y^4 по векторам x^1, x^2, x^3, x^4 подставить вместо x^3 его разложение по векторам x^1, x^2, x^4, x^5 , которое имеет

вид (см.формулу 5.19).

$$x^3 = x^5 - x^1 + x^2 + x^4$$

Подставляя, получаем

$$y^1 = x^5 - x^4 \qquad a_1 = 0,$$
$$y^2 = x^1 + x^4 - x^5 \qquad a_2 = 1,$$
$$y^3 = x^5 - x^1 + x^2 - 2x^4 \qquad a_3 = 0,$$
$$y^4 = x^4 \qquad a_4 = 1.$$

Новая задача о ранце имеет вид:

$$A = 0 \cdot x_1 + 1 \cdot x_2 - 1 \cdot x_3 + 1 \cdot x_4 \longrightarrow max$$
$$17x_1 + 12x_2 + 8x_3 + 10x_4 \leqslant 30 .$$

Оптимальное решение $x^6 = (0,1,0,1)$, $A = 2 > 1$. Поскольку

$$x^6 = y^2 + y^4 = x^1 + 2x^4 - x^5,$$

то при увеличении Δ_6 уменьшается Δ_1 на величину Δ_6, Δ_4 на величину $2\Delta_6$ и увеличивается Δ_5 на Δ_6 величину Δ_6. Максимальное значение Δ_6 равно $min(\Delta_1, \frac{\Delta_4}{2}) = 1\frac{1}{2}$, если взять $\Delta_6 > 1\frac{1}{2}$, то $\Delta_4 - 2\Delta_6 < 0$).

Новое решение x^1, x^2, x^5, x^6,

$\Delta_1^2 = \Delta_1^1 - 1\frac{1}{2} = 3\frac{1}{2}$, $\Delta_2^2 = \Delta_2^1 = 10$, $\Delta_5^2 = \Delta_5^1 + 1\frac{1}{2} = 8\frac{1}{2}$, $\Delta_6^2 = \Delta_6 = 1\frac{1}{2}$.

Повторяя процедуру улучшения еще раз, придем к следующей задаче о ранце:

$$A = \frac{1}{2}x_1 + \frac{1}{2}x_2 + 0 \cdot x_3 + \frac{1}{2}x_4 \longrightarrow max$$

при условии

$$17x_1 + 12x_2 + 8x_3 + 10x_4 \leqslant 30 .$$

Ее оптимальное решение имеет $A = 1$. Следовательно, согласно теореме 5.4, решение x^1, x^2, x^5, x^6 является оптимальным.

Замечание I. Изложенный алгоритм является ничем иным,

как процедурой симплекс-метода решения задач линейного программирования, учитывающей возможность представления любого допустимого вектора X^K в виде $(x_1, \ldots, x_n)$ где $x_i = 0$ или I и выполняется условие (5.I3). Это позволяет проверить оптимальность исходного базиса не перебирая все остальные вектора, а решая задачу о ранце.

Замечание 2. Если вектор, подлежащий исключению из базиса имеет $\Lambda = 0$, то, очевидно, и вновь вводимый вектор будет иметь $\Delta = 0$, и уменьшения времени выполнения операций не произойдет. При этом возможно зацикливание, то есть возврат через несколько шагов к исходному базису. Для того, чтобы избежать зацикливания необходимы специальные меры. Простейший способ - запоминать вектора, исключаемые из базиса с тем, чтобы не вводить их снова. При этом, как только очередной вводимый вектор приведет к уменьшению времени выполнения операций, все ранее запомненные вектора можно забыть.

Замечание 3. Если в каждой операции участвуют ресурсы нескольких видов, то задача решается аналогично. Однако, вместо задачи о ранце на каждом шаге улучшения решения придется решать задачу целочисленного линейного программирования:

$$\sum_{i=1}^{n} a_i x_i \to max$$

при условиях

$$\sum_{i=1}^{n} \beta_{ij} x_i \le N_j, \quad j = 1, 2, \ldots, m,$$

где b_{ij} - количество ресурсов j -го вида на i -ой операции.

2.4. Минимизация числа перерывов операций

Оптимальное решение задачи, рассмотренной в предыдущем пункте, определено с точностью до перестановки интервалов Δ_k. Действительно, при любой очередности интервалов продолжительность комплекса операций остается постоянной. Обозначим $\{x^i\}$ оптимальное решение задачи, где $i = 1,2,\ldots,n$ номера соответствующих интервалов.

Задача 5.2. Определить перестановку интервалов, при которой суммарное число перерывов операций минимально.

Для решения задачи определим полный граф с $(n+1)$ вершинами $0,1,2,\ldots, n$, где вершина i соответствует вектору x^i, вершины 0 соответствует нулевому вектору $x^0 = (0,\ldots,0)$. Определим длину l_{ij} дуги (i,j) по формуле

$$l_{ij} \sum_{k=1}^{n} |x_k^i - x_k^j| \qquad (5.20)$$

Пусть $(0, i_1, i_2, \ldots, i_n, 0)$ некоторый гамильтонов контур. Его длина $\mathcal{L} = 2(n+z)$, где z - число перерывов операций при очередности интервалов $i_1, i_2, \ldots, i_n$. Следовательно, минимальное число перерывов будет соответствовать гамильтонову контуру минимальной длины. Для его определения можно применить алгоритм, описанный в приложении.

Пример 5.4. В предыдущем пункте было найдено оптимальное решение

$x^1 = (1,1,0,0)$, $x^2 = (0,1,1,1)$, $x^3 = (1,0,0,1)$,
$x^4 = (0,1,0,1)$

(вектора x^3 и x^4 обозначены в предыдущем пункте как x^5 и x^6). Полный граф с расстояниями, вычисляемыми по формуле (5.20) приведен на рис.5.3. Один из гамильтоновых контуров минимальной длины выделен двойными дугами. Его длина равна $\mathcal{L}$ = I0. Следовательно, минимальное число перерывов

$$r = \frac{\mathcal{L}}{2} - n = 1 .$$

Соответствующее упорядочение интервалов 4,2,3,I или I,3,2,4.

Задачу 5.2. можно поставить в более общем виде.

Задача 5.3. Дано конечное число векторов $\{x^j\}$, $j = 1,2,\dots,s$ и определено расстояние l_{ij} между любой парой векторов x^i и x^j. Найти перестановку $(i_1, i_2, \dots, i_n)$, на которой

$$\mathcal{L} = \sum_{k=1}^{n} l_{i_k i_{k+1}} \qquad (i_{n+1} = i_1)$$

достигает минимальной величины.

Упражнение 5.2. Пусть x^j одномерные вектора и пусть

$$l_{ij} = \max (x^i, x^j)$$

Докажите, что любая перестановка $i_1, i_2, \dots, i_n$, такая что

$$x^{i_1} < x^{i_2} < \dots \leqslant x^{i_s} \geqslant x^{i_{s+1}} \geqslant \dots \geqslant x^{i_n}$$

определяет оптимальное решение задачи 5.3. Докажите, что эта же перестановка определяют оптимальное решение задачи 5.3.

Упражнение 5.3. Решите задачу 5.3 при

$$l_{ij} = \max_k |x_k^i - x_k^j|$$

§ 3. Сети с упорядоченными событиями

3.1. Минимизация продолжительности комплекса операций

В этой главе нам удобно пользоваться изображением сетей в виде "операции-дуги". Как мы уже отмечали в § I этой главы, вершины сети при таком изображении соответствуют событиям (событием называется факт окончания одной или нескольких операций). Говорят, что события сети, упорядочены, если задана очередность их свершения. Будем предполагать, что события пронумерованы в порядке их свершения. Заметим, что если в сети существует гамильтонов путь, то существует единственная очередность свершения событий, определяемая этим путем.

При изображении операций дугами сети обычно каждую операцию обозначают также как и соответствующую ей дугу (например, операция (2.4) имеет начальное событие 2 и конечное 4). Однако, мы сохраним прежний способ обозначения каждой операции одним числом. Пусть сеть имеет $(+I)$ событий $0,I,2,\ldots,$ z. Обозначим R_S - множество операций, которые могут выполняться в интервале между $(S-I)$ и S -ым событиями $(S = 1,2,\ldots, z)$, Q_i - множество интервалов, в которых может выполняться операция i. Наконец, обозначим x_{iS} - объем операции i, выполняемый в S -ом интервале. Заметим, что если x_{iS} заданы, то для каждого интервала получаем задачу с независимыми операциями, поскольку части операций, выполняемые внутри одного интервала, независимы. Поэтому, применяя методы, рассмотренные в § 2, можно определить минимальную длительность $\Delta_S(z_S)$, где $z_S = \{x_{iS} \mid i \in R_S\}$. Продолжительность комплекса

$$T(z) = T(z_1, z_2, \ldots, z_k) = \sum_{s=1}^{k} \Delta_s(z_s) \qquad (5.21)$$

В силу теоремы 5.4 $T(z)$ – выпуклая (к низу) функция z даже при невыпуклых $f_i(v_i)$. Таким образом, задача минимизации $T(z_1, z_2, \ldots, z_k)$ при ограничениях

$$\sum_{s \in Q_i} x_{is} = w_i, \quad i = 1, 2, \ldots, n \qquad (5.22)$$

является задачей выпуклого программирования. Этот факт имеет фундаментальное значение, поскольку методы решения задач выпуклого программирования достаточно хорошо разработаны.

Пример 5.5. Пусть все операции выполняются ресурсами одного вида и $f_i(v_i) = v_i^{1/\alpha}$, $(\alpha > 1)$, $i = 1, 2, \ldots, n$. В примере 5.2 была получена формула для минимальной длительности комплекса в случае независимых операций. В нашем случае эта формула дает

$$\Delta_s(z_s) = \frac{1}{N^{1/\alpha}} \left(\sum_{i \in R_s} (x_i^{\alpha}) \right)^{1/\alpha} \qquad (5.23)$$

Время завершения комплекса

$$T(z_1, z_2, \ldots, z_k) = \frac{1}{N^{1/\alpha}} \sum_{s=1}^{k_1} \left(\sum_{i \in R_s} (x_{is})^{\alpha} \right)^{1/\alpha} \qquad (5.24)$$

Применяя метод множителей Лагранжа для задачи минимизации (5.24) при условиях 5.22, получаем необходимые и достаточные условия оптимальности

$$w_{is} = \frac{x_{is}}{\Delta_s(z_s)} = \lambda_i, \quad i = 1, 2, \ldots, n \qquad (5.25)$$

Из (5.25) следует, что скорость операции i не зависит от $S \in Q_i$, то есть одна и та же в любом интервале множества Q_i. А это значит, что в оптимальном решении каждая операция выполняется с постоянной интенсивностью U_i. Опираясь на этот факт, можно доказать еще одно свойство оптимального решения в случае степенных зависимостей $f_i(U_i)$: сумма ресурсов U_i на операциях, имеющих событие S конечным, равна сумме ресурсов на операциях, имеющих событие S начальным для любого $S = 1,2,\ldots, z-1$. Таким образом, ресурсы образуют поток величины N по сети.

Упражнение 5.3. Получите необходимые и достаточные условия оптимальности при линейной зависимости скоростей операций от количества ресурсов, то есть

$$f_i(U_i) = \begin{cases} U_i, & \text{если } 0 \le U_i \le \beta_i \\ \beta_i, & \text{если } U_i \ge \beta_i \end{cases}$$

для всех $i = 1,2,\ldots, n$. Предполагается, что все операции выполняются ресурсами одного вида, количество которых N. Приведем еще одно важное свойство, которым обладают сети с упорядоченными событиями.

Теорема 5.5. Минимальная продолжительность $T_m(\bar{W})$ комплекса является выпуклой (к низу) функцией $\bar{W}$.

Доказательство опирается на аналогичную теорему 5.4. для независимых операций. Пусть $\bar{x}^1 = \{x^1_{is}\}$ оптимальное решение задачи при объемах $\bar{W}^1$, $\bar{x}^2 = \{x^2_{is}\}$ оптимальное решение задачи при объемах $\bar{W}^2$. В силу теоремы 5.4

$$\Delta_s(\alpha \bar{x}^1_s + (1-\alpha)\bar{x}^2_s) \le \alpha \Delta_s(\bar{x}^1_s) + (1-\alpha)\Delta_s(\bar{x}^2_s), \quad 0 \le \alpha \le 1$$

Так как решение $\bar{x} = \alpha \bar{x}^1 + (1-\alpha)\bar{x}^2$ является допустимым решением задачи при объемах $\bar{W} = \alpha \bar{W}^1 + (1-\alpha)\bar{W}^2$,

то $$T_m(\alpha\bar{W}^1+(1-\alpha)\bar{W}^2) \le T(Z) = \sum_{s=1}^{2}\Delta_s(\alpha Z_s^1+(1-\alpha)Z_s^2) \le$$
$$\le \alpha T_m(\bar{W}^1)+(1-\alpha)T_m(\bar{W}^2).$$

3.2. Фиксированные интенсивности

Алгоритм минимизации продолжительности комплекса при фиксированных интенсивностях операций, рассмотренный в п.2.3, можно с небольшими изменениями применить для решения соответствующей задачи на сети с упорядоченными событиями. Изменения касаются решения задачи о ранце. Поскольку любой допустимый вектор x должен определять набор независимых операций, то эти операции должны все принадлежать одному из множеств R_s. Если обозначить R_{sj} - множество операций $i \in R_s$, выполняемых ресурсами j-го вида, то для проверки оптимальности решения необходимо решить задачу о ранце вида

$$C_{js} = \sum_{i\in R_{sj}} a_i x_i \to max$$

$$\sum_{i\in R_{sj}} \beta_i x_i \le N_j$$

для каждого множества $R_{sj} \neq \Phi$. При этом, если $C = \max_s \sum_{j=1}^{m} C_{js} \le 1$, то решение оптимально.

Пример 5.4. Решим задачу для сети, приведенной на рис.5.4 (числа у дуг соответствуют номерам операций). Данные об операциях приведены ниже. Примем, что все операции выполняются ресурсами одного вида в количестве N = II.

i	I	2	3	4	5
τ_i	2	4	6	5	2
β_i	6	5	4	4	7

I этап. Получение начального решения. Для получения начального решения применим эвристический алгоритм, согласно которому в первую очередь выполняются операции с минимальным значением момента позднего начала t_i^n (в случае одинаковых t_i^n начинаем любую операцию). Величины t_i^n приведены ниже

i	I	2	3	4	5
t_i^n	0	I	I	2	5

Применяя описанное выше правило, получаем следующее решение

$$x^1 = (I,I,0,0,0), \quad \Delta_1 = 2$$
$$x^2 = (0,I,I,0,0), \quad \Delta_2 = 2$$
$$x^3 = (0,0,I,I,0), \quad \Delta_3 = 4$$
$$x^4 = (0,0,0,I,I), \quad \Delta_4 = I$$
$$x^5 = (0,0,0,0,I), \quad \Delta_5 = I.$$

Продолжительность комплекса $T = \sum_{s=1}^{5} \Delta_s = 10$.

Выражаем вектора y^1, y^2, y^3, y^4, y^5 через базисные вектора

$$y^5 = x^5, \qquad a_5 = I,$$
$$y^4 = x^4 - x^5, \qquad a_4 = 0,$$
$$y^3 = x^3 - x^4 + x^5, \qquad a_3 = I,$$
$$y^2 = x^2 - x^3 + x^4 + x^5, \qquad a_2 = 0,$$
$$y^1 = x^1 - x^2 + x^3 - x^4 + x^5, \qquad a_1 = I.$$

Решаем следующие три задачи о ранце для множеств R_1, R_2, R_3

а) $C_1 = x_1 + x_3 \longrightarrow max$,

$6x_1 + 4x_3 \leq 11$.

б) $x_3 \longrightarrow max$,

$4x_3 \leq 11$

в) $x_3 + x_5 \longrightarrow max$,

$4x_3 + 7x_5 \leq 11$.

Легко видеть, что оптимальные значения $C_1=2$, $C_2=1$, $C_3=2$. Вводим в базис, например, вектор $x^6=(0,0,I,0,I)= y^3+y^5= x^3-x^4+2x^5$ являющийся оптимальным решением задачи (в). Имеем $\Delta_6 = min\left(\frac{\Delta_3}{1}, \frac{\Delta_5}{2}\right) = \frac{1}{2}$. Исключая вектор x^5, получаем новое решение $\Delta_1' = \Delta_1 = 2$, $\Delta_2' = \Delta_2 = 2$, $\Delta_3' = \Delta_3 - \frac{1}{2} = 3\frac{1}{2}$,

$$\Delta_4' = \Delta_4 + \frac{1}{2} = 1\frac{1}{2}, \quad \Delta_6' = \frac{1}{2}$$

Подставляем $x^5 = \frac{1}{2}(x^6 - x^3 + x^4)$ в формулу для y^i

$$y^1 = x^1 - x^2 + \frac{1}{2}x^3 - \frac{1}{2}x^4 + \frac{1}{2}x^6, \quad a_1 = I/2,$$

$$y^2 = x^2 - \frac{1}{2}x^3 + \frac{1}{2}x^4 - \frac{1}{2}x^6, \quad a_2 = I/2,$$

$$y^3 = -\frac{1}{2}x^3 - \frac{1}{2}x^4 + \frac{1}{2}x^6, \quad a_3 = I/2,$$

$$y^4 = \frac{1}{2}x^1 + \frac{1}{2}x^3 - \frac{1}{2}x^6, \quad a_4 = I/2,$$

$$y^5 = -\frac{1}{2}x^3 + \frac{1}{2}x^4 + \frac{1}{2}x^6, \quad a_5 = I/2.$$

Новые задачи о ранце

а) $C_1 = \frac{1}{2}(x_1 + x_2 + x_3) \longrightarrow max$,

$6x_1 + 5x_2 + 4x_3 \leq 11$.

б) $C_2 = \frac{1}{2}(x_2 + x_3 + x_4) \longrightarrow max$,

$5x_2 + 4x_3 + 4x_4 \leq 11$.

в) $C_3 = \frac{1}{2}(x_3 + x_4 + x_5) \to max$,

$4x_3 + 4x_4 + 7x_5 \leq 11$.

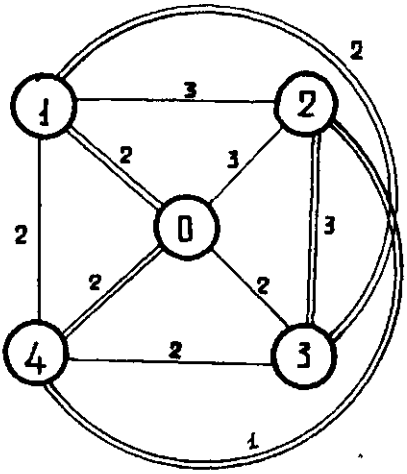


Рис. 5.3

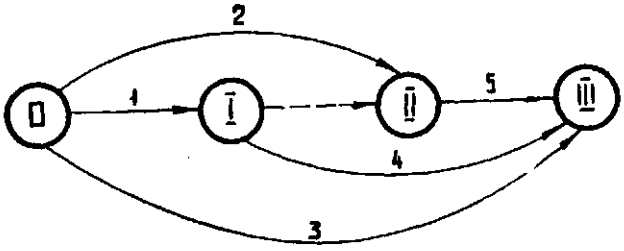


Рис. 5.4

Все три задачи имеют оптимальное значение $C = 1$.

Поэтому полученное ранее решение $(x^1, x^2, x^3, x^4, x^6)$.

T = 9,5, является оптимальным.

Упражнение 5.3. Поставьте задачу о минимизации числа перерывов операций в полученном решении (аналогично случаю независимых операций п.2.4). Какому дополнительному условию должен удовлетворять гамильтонов контур, определяющий порядок векторов?

Упражнение 5.4. Определите минимальное увеличение уровня ресурсов N, необходимое для сокращения продолжительности комплекса и решите задачу для нового значения уровня. При каком уровне ресурсов величина простоев, равная $NT-W$ меньше?

Если все $\beta_i = 1 \quad i = 1,2,\ldots, n$, то каждая задача о ранце решается элементарно. Максимальное значение C_{js} равно сумме N_j положительных наибольших значений $a_i (i \in R_{sj})$ или просто сумме положительных величин a_i, если их число меньше N_j. Если к тому же все $N_j = 1$, $j = 1,2,\ldots, m$, существует эффективный алгоритм решения задачи (предложен В.Деменчуком). Оптимальное решение получается по следующему правилу: в любой момент времени ресурсы в первую очередь назначаются на операции с меньшим номером конечного события.

Докажем оптимальность решения, получаемого по этому правилу. Пусть дано оптимальное решение, в некотором интервале (t_1, t_2) которого выполняется операция i, а в интервале (t_3, t_4) - операция j, причем $t_3 \geq t_2$ и номер конечного события операции i больше чем номер конечного события операции j (операции i и j требуют од-

ного и того же вида ресурсов). Переставляя части операций длительности $\Delta = min(t_2 - t_1, t_4 - t_3)$ получаем решение, в котором операция i выполняется раньше операции j. При этом, очевидно, продолжительность комплекса не меняется. Повторяя такое преобразование всякий раз, когда нарушается описанное выше правило, получим решение, совпадающее с решением, полученным при помощи правила Деменчука с той же продолжительностью комплекса, которое также является оптимальным.

Пример 5.6. Применим правило Деменчука к решению задачи для сети, приведенной на рис.5.5 (операции первого класса показаны одинарными дугами, операции второго класса - двойными, числа в скобках равны продолжительностям τ_i)

$t = 0$ Начинаем операции I и 2.

$t = 3$ Операции I и 2 закончены.

Поэтому начинаем операцию 5 и операцию 3 (поскольку она имеет меньший номер конечного события).

$t = 5$ Операция 3 закончена. Начинаем операцию 4 и операцию 6 (операцию 5 прерываем, поскольку она имеет больший номер конечного события, чем операция 6).

$t = 9$. Операция 4 закончена. Начинаем операцию 7.

$t = 10$. Операция 7 закончена.

$t = 11$. Операция 6 Закончена. Возобновляем операцию 5 и начинаем операцию 8.

$t = 14$. Операция 8 закончена.

$T = 16$. Все операции закончены.

Упражнение 5.5. Докажите, что если $f_i(U_i)$ выпуклые

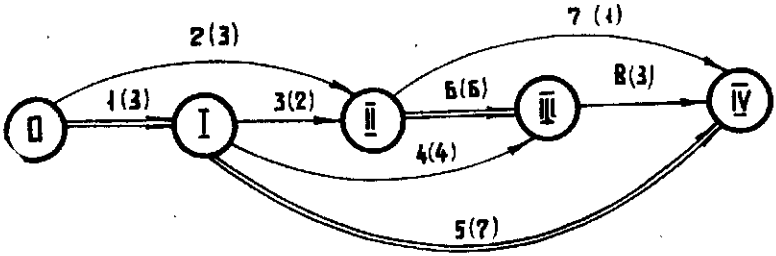

2(3)
7 (1)
1(3)
3(2)
6(6)
8(3)
4(4)
5(7)
0
I
II
III
IV

Рис. 5.5

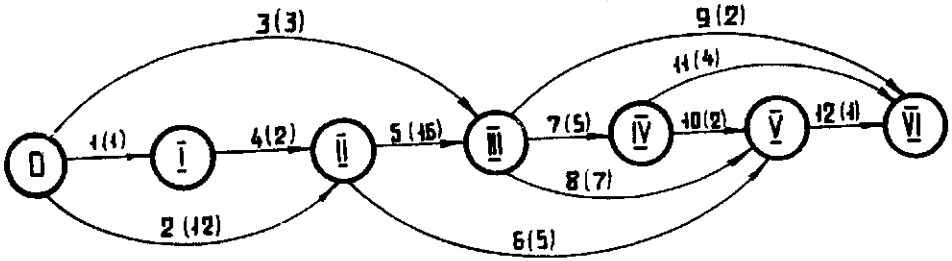

3(3)
9(2)
11(4)
1(1)
4(2)
5(15)
7(5)
10(2)
12(1)
8(7)
2(12)
6(5)
0
I
II
III
IV
V
VI

Рис. 5.6

(книзу) функции $(i = 1, 2, \ldots, n)$, то существует оптимальное решение, в котором каждая операция выполняется максимальным количеством ресурсов своего вида.

Правило Демунчука дает оптимальное решение задачи и для случая, когда все $f_i(U_i)$ – выпуклые (книзу) функции (см. упражнение 5.5).

§ 4. Оптимизация при заданных продолжительностях интервалов

4.1. Постановки задач

В предыдущих параграфах мы в основном решали задачу минимизации продолжительности комплекса при заданных уровнях ресурсов. Рассмотрим несколько простых обратных задач, связанных с минимизацией уровня ресурсов или в некотором смысле с равномерным использованием ресурсов при заданной продолжительности комплекса. Более того, будем предполагать заданными и длительности Δ_s интервалов между событиями комплекса. Без ограничения общности можно считать, что все работы выполняются ресурсами одного вида, так как в случае m видов ресурсов задача распадается на m независимых задач.

Пусть $f_i(U_i) = U_i, \quad 0 \le U_i \le \beta_i, \quad i = 1, 2, \ldots, n$

Задача 5.4. (минимизация уровня ресурсов). Определить минимальное количество N ресурсов, необходимых для выполнения комплекса за время $T = \sum_{s=1}^{s} \Delta_s$

Задача 5.5. (равномерное использование ресурсов). Пусть $f_i(U_i) = U_i, \quad i = 1, 2, \ldots, n$. Требуется определить $\{U_i(t)\}$ так, чтобы комплекс был выполнен за время $T = \sum_{s=1}^{s} \Delta_s$ и величина

$$\Phi = \int_0^T \left(\sum_{i=1}^{n} v_i(t) \right)^2 dt \qquad (5.26)$$

была минимальной.

Замечая, что в оптимальном решении количество ресурсов v_{is} на i -ой операции в S -ом интервале равно

$$v_{is} = \frac{x_{is}}{\Delta_s}, \quad i \in R_s, \quad S = 1, 2, \ldots, z,$$

получаем задачу минимизации

$$\Phi = \sum_{S=1}^{z} \frac{1}{\Delta_s} \left(\sum_{i \in R_s} x_{is} \right)^2 \qquad (5.27)$$

при ограничениях

$$\sum_{S \in Q_i} x_{is} = W_i, \quad i = 1, 2, \ldots, n \qquad (5.28)$$

Рассмотримрешение каждой из задач.

4.2. Минимизация уровня ресурсов

Определим транспортную сеть со входом x_0 , выходом z и вершинами i, S, $i = 1, 2, \ldots, n$, $S = 1, 2, \ldots, z$. Вход x_0 соединим с каждой вершиной i дугой (x_0, i) пропускной способности W_i , каждую вершину i соединим с каждой вершиной S дугой (i, S) пропускной способности $\beta < \Delta_S$. наконец, каждую вершину S соединим с выходом z дугой (S, z) пропускной способности N_{Δ_S} .

Полагаем $N_0 = \frac{1}{T} \sum_{i=1}^{n} W_i = \frac{W}{T}$

Определяем поток максимальной величины в сети 2,5 . Если величина потока $\mathcal{Y}_{max} = W$, то потоки y_{is} по дугам (i, S) определяют оптимальное решение задачи.Пусть $\mathcal{Y}_{max} < W$.

Обозначим D множество вершин S, помеченных на последнем шаге алгоритма Форда-Фалкерсона [5] (дуги $(S,\bar{z})$ $S \in D$ входят в разрез минимальной пропускной способности). Вычисляем

$$N_1 = \frac{W - \mathcal{Y}_{max}}{\sum_{S \in D} \Delta_S} + N_0$$

С новым значением N_1 повторяем описанную процедуру. За конечное число шагов будет получен поток величины $\mathcal{Y}_{max} = W$, который и определит оптимальное решение задачи. Обоснование алгоритма предоставляем читателям.

Упражнение 5.6. Учитывая специфику транспортной сети, предложите более эффективный алгоритм определения потока максимальной величины, чем описанный в [2,5].

4.3. Равномерное использование ресурсов

Обозначим A_K суммарный объем операций, которые должны выполняться в интервалах от I до K, B_K - суммарный объем операций, которые могут выполняться в интервалах от I до K. Далее, обозначим $T_K = \sum_{S=1}^{K} \Delta_S$ сумму продолжительностей первых K интервалов.

Теорема 5.5. Если выполняются условия

$$A_K \leqslant \frac{W}{T} \cdot T_K \leqslant B_K, \quad K = 1,2,\ldots,\tau \qquad (5.29)$$

то минимальная величина (5.27)

$$\Phi_{min} = \frac{W^2}{T}$$

Доказательство. Для случая $\tau = 1$ теорема очевидна, так как в этом случае $A_1 = B_1 = W$. Допустим, что теорема

справедлива для сети с (z - I) событиями. Рассмотрим сеть с z событиями, для которой условия (5.29) выполняются. Удалим из сети все операции, которые должны выполняться в первом интервале, затем будем удалять операции, которые могут выполняться в первом интервале и должны выполняться в первом и во втором, затем те, которые могут выполняться в первом и должны выполняться в первых трех и т.д. до тех пор, пока суммарный объем удаленных операций (или частей операций) не составит $\frac{W}{T} \cdot \Delta_1$. После этого исключим I-ний интервал. Нетрудно видеть, что условия (5.29) продолжают выполняться для оставшейся сети с (z -I) интервалами. Это доказывает теорему.

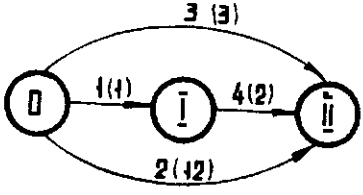
Пусть K_0 ближайший интервал, для которого условие (5.29) нарушается. Если $A_{K_0} > \frac{W}{T} T_{K_0}$, то задачу равномерного использования ресурсов решаем отдельно для сети, которая включает все операции, составляющие A_{K_0} , и отдельно для оставшейся сети.

Если $\frac{W}{T} T_{K_0} > B_{K_0}$, то задачу решаем отдельно для сети, которая включает все операции, составляющие B_{K_0} и отдельно для оставшейся сети. Продолжая таким образом, разбиваем сеть на подсети, для каждой из которых выполняются условия (5.29).

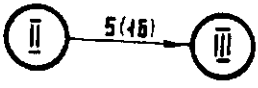
Пример 5.6. Решим задачу для сети, изображенной на рис.5.6 (объемы операций указаны в скобках у соответствующих дуг, числа без скобок соответствуют номеру операции). Пусть все Δ_S = I, S = I,2,3,4,5,6. Имеем W = 60, T = 6, $T_K = K \frac{W}{T}$ = I0. Вычисляем

A_1 = I, B_1 = I6, I < I0 < I6

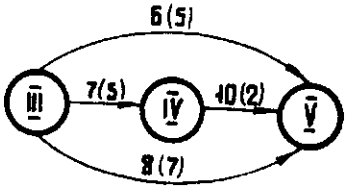
A_2 = I5, B_2 = I8, I5 < 20 < I8



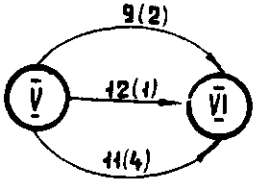
G_1



G_2



G_3



G_4

Рис. 5.7

Условие (5.29) нарушилось для второго интервала. Так как $B_2 < \frac{W}{T} T_2$, то разбиваем сеть на две. В первую подсеть входят операции I, 2, 3 и 4, а во вторую - все остальные. Продолжая процедуру проверки условий (5.29) и разбиения сетей, получаем четыре подсети, показанные на рис.5.7. Проверьте самостоятельно, что для каждой из подсетей G_1, G_2, G_3 и G_4 условия (5.29) выполняются (для подсетей G_2 и G_4 это очевидно, так как они имеют всего один интервал. Обозначим $W(G_i)$ и $T(G_i)$, соответственно, суммарный объем операций и суммарную длительность интервалов сети G_i , i = I,2,3,4.

Имеем

i	I	2	3	4
$W(G_i)$	I8	I6	I9	7
$T(G_i)$	2	I	2	I

Согласно теореме 5.5.

$$\Phi_{min} = \sum_{i=1}^{4} \frac{W^2(G_i)}{T(G_i)} = \frac{18^2}{2} + \frac{16^2}{1} + \frac{19^2}{2} + \frac{7^2}{1} = 647,5.$$

Глава УI.

ЗАДАЧИ КАЛЕНДАРНОГО ПЛАНИРОВАНИЯ

Название этой главы несколько условно, поскольку и задачи, рассмотренные в предыдущей главе, можно отнести к задачам календарного планирования. По сути дела, в этой главе мы рассмотрим задачи распределения ресурсов, в которых каждая операция выполняется с единичной интенсивностью, причем прерывать выполнение начатой операции запрещается. Задачи этого класса носят типично комбинаторный характер, и эффективные точные методы получены только для относительно небольшого числа частных постановок.

§ I. Обработка деталей на станках

I.I. Постановка задачи

Требуется обработать n деталей. Каждая деталь должна пройти обработку в определенной последовательности на m станках. Предполагается, что одновременно станок может обрабатывать только одну деталь, и каждая деталь может одновременно обрабатываться только на одном станке. Перерывы в выполнении операций запрещены. Заданы продолжительности t_{ij} обработки i -ой детали на j -ом станке. Количество станков j -го типа равно N_j . Требуется организовать обработку деталей таким образом, чтобы минимизировать время обработки всех деталей. Для двух частных случаев задачи известны эффективные точные алгоритмы. Эти случаи получили специальные названия "задача Джонсона" и "задача редактора".

Для их решения рассмотрим предварительно понятие псевдопотенциального графа.

I.2. Псевдопотенциальный граф

Полный $(n+1)$ вершинный симметрический граф G с длинами дуг l_{ij}, $i,j = 0,1,2,\dots,n$ называется псевдопотенциальным, если длина его любого гамильтонова контура равна одному и тому же числу.

Теорема 6.I. Любой подграф псевдопотенциального графа является псевдопотенциальным (докажите самостоятельно).

Теорема 6.2. Для того, чтобы граф был псевдопотенциальным, необходимо и достаточно существование чисел α_i, β_i, $i = 0,1,2,\dots,n$ таких, что $l_{ij} = \beta_j - \alpha_i$ для всех $i,j = 0,1,2,\dots,n$. Достаточность очевидна, поскольку длина любого гамильтонова контура $(0, i_1, i_2, \dots, i_n, 0)$ равна

$$\mathcal{L} = \sum_{i=0}^{n} (\beta_i - \alpha_i) \qquad (6.I)$$

Докажем необходимость. Теорема тривиальна при $n = I$. Допустим теорема справедлива при любом числе вершин не превосходящим (n) и докажем для $(n + I)$ - вершинного графа. Удалим вершину n. Получим n - вершинный граф, который является псевдопотенциальным в силу теоремы 6.1. Следовательно, существуют числа α_i, β_i, $i = 0,1,2,\dots,n-1$ такие, что $l_{ij} = \beta_j - \alpha_i$ для всех $i,j = 1,2,\dots,n-1$. Добавим вершину n. Из условия псевдопотенциальности графа нетрудно получить условия $l_{in} - l_{kn} = \alpha_k - \alpha_i$ для любых $k \neq i$. Следовательно, $l_{in} = l_{kn} + \alpha_k - \alpha_i$ и значит $l_{kn} + \alpha_k = \beta_n$ не зависит от k. Окончательно получаем $l_{kn} = \beta_n - \alpha_k$

Аналогично доказывается существование α_n, такого, что $c_{nk} = \beta_k - \alpha_n$ для всех k. Теорема доказана.

Для дальнейшего примем $\alpha_0 = 0$. Пусть $\mu = (0, i_1, i_2, \ldots, i_n, 0)$ произвольный гамильтонов контур. Обозначим $M_j(\mu) = \sum_{k=1}^{j} (\beta_{i_k} - \alpha_{i_{k-1}})$ сумму длин первых j дуг контура μ.

Задача 6.1. Определить гамильтонов контур, имеющий минимальное значение

$$M(\mu) = \max_{1 \le j \le n} M_j(\mu) \qquad (6.2)$$

Обозначим $\gamma_i^e = \alpha_i - \beta_i$, $i = 1, 2, \ldots, n$.

Обозначим далее M_{min} – минимальную величину $M(\mu)$ и пусть $(0, i_1, i_2, \ldots, i_n, 0)$ оптимальный гамильтонов контур. Тогда имеет место следующая система неравенств

$$\begin{aligned} M_{min} &\ge \beta_{i_1}, \\ M_{min} + \gamma_{i_1}^e &\ge \beta_{i_2}, \\ M_{min} + \gamma_{i_1}^e + \gamma_{i_2}^e &\ge \beta_{i_3}, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & \\ M_{min} + \gamma_{i_1}^e + \ldots + \gamma_{i_{n-1}}^e &\ge \beta_{i_n}. \end{aligned} \qquad (6.3)$$

Если $\gamma_{i_{s-1}}^e < 0$, $\gamma_{i_s}^e \ge 0$, то из (6.3) следует справедливость соответствующей системы неравенств для гамильтонова контура $(0, i_1, \ldots, i_{s-2}, i_s, i_{s-1}, i_{s+1}, \ldots, i_n, 0)$. Поэтому всегда существует оптимальное решение, в котором сначала обходятся вершины с положительными γ^e, а затем – с отрицательными (вершины с $\gamma_i^e = 0$ можно отнести либо в группу положительных, либо в группу отрицательных). Далее, если $\gamma_{i_{s-1}}^e > 0$, $\gamma_{i_s}^e \ge 0$ и $\beta_{i_{s-1}} > \beta_{i_s}$ то из (6.3) следует справедливость соответствующей системы неравенств для гамильтонова контура $(0, i_1, \ldots, i_{s-1}, i_s, i_{s+1}, \ldots, i_n, 0)$

Наконец, если $\gamma^e_{i_{s-1}} \le 0$, $\gamma^e_{i_s} \le 0$ и $\beta_{i_{s-1}} > \beta_{i_s}$,
то из (6.3) также следует справедливость соответствующей системы неравенств для гамильтонова контура $(0, i_1, \dots, i_{s-2}, i_{s-1}, i_s, i_{s+1}, \dots, i_n, 0)$.
Докажем, например, последнее утверждение. Из (6.3) имеем

$$M_{min} + \sum_{j=1}^{s-2} \gamma^e_{ij} \ge \beta_{i_{s-1}} - \alpha_{i_{s-1}} - \gamma^e_{i_{s-1}}$$

$$M_{min} + \sum_{j=1}^{s-2} \gamma^e_{ij} + \gamma^e_{i_{s-1}} \ge \beta_{i_s} - \alpha_{i_s} - \gamma^e_{i_s}.$$

Но тогда тем более

$$M_{min} + \sum_{i=1}^{s-2} \gamma^e_{ij} + \gamma^e_{i_s} \ge \alpha_{i_s}, \quad \text{так как } \gamma^e_{i_{s-1}} \le 0,$$

$$M_{min} + \sum_{j=1}^{s-2} \gamma^e_{ij} + \gamma^e_{i_s} + \gamma^e_{i_{s-1}} \ge \alpha_{i_s} \ge \alpha_{i_{s-1}}.$$

Таким образом, мы доказали следующую теорему.

Теорема 6.3. Существует оптимальное решение, в котором сначала идут вершины с $\gamma^e_i \ge 0$ в порядке возрастания величин β_i, а затем вершины с $\gamma^e_i \le 0$ в порядке убывания величин α_i. Пусть $(0, i_1, i_2, \dots, i_s, i_{s+1}, \dots, i_n, 0)$ соответствующий гамильтонов контур, то есть

$\gamma^e_{ij} \ge 0$, $j = 1, 2, \dots, s$ причем $\beta_{i_1} \le \beta_{i_2} \le \dots \le \beta_{i_s}$

$\gamma^e_{ij} \le 0$, $j = s+1, \dots, n$ причем $\alpha_{i_{s+1}} \ge \alpha_{i_{s+2}} \ge \dots \ge \alpha_{i_n}$

Тогда

$$M_{min} = \max\left[\beta_{i_1}; \max_{1 \le k \le n} \left(\beta_{i_{k+1}} - \sum_{j=1}^{k} \gamma^e_{ij}\right)\right] \qquad (6.4)$$

Пример 6.I. Пусть n = 5. Значения α_i , β_i и γ_i приведены ниже.

i	I	2	3	4	5
α_i	I6	I9	I6	I3	20
β_i	I6	20	I8	II	I9
γ_i	0	- I	-2	-2	I

Оптимальное решение (0,4,I,5,2,3,0),

$M_{min} = max\,[11;\ 16-2;\ 19-(2+0);\ 20-(2+0+1);\ 18-(2+0+1-1)] = 17$.

I.3. Задача Джонсона

В этой известной задаче каждая деталь проходит обработку на двух станках, причем порядок обработки одинаковый для всех деталей (сначала на первом станке, а затем на втором). Имеется по одному станку каждого типа, то есть $N_1 = N_2 = 1$. Обозначим для упрощения записи a_i - продолжительность обработки i -ой детали на I-ом станке, а b_i - на втором. Докажем сначала следующее простое утверждение:

Теорема 6.4. Существует оптимальное решение, в котором порядок обработки деталей одинаковый на обоих станках.

Доказательство. Пусть $(i_1, i_2, \ldots, i_n)$ некоторая очередность обработки деталей на первом станке. Обозначим

$$T_{i_s} = \sum_{j=1}^{s} a_{i_j}$$

момент окончания обработки детали i_s на I-ом станке (T_{i_s} определяет ранний возможный момент начала обработки детали на втором станке). Поставим задачу определить оптимальный порядок обработки деталей на втором станке при условии начала обработки детали i_s не раньше T_{i_s} . Предостав-

ляем самостоятельно доказать, что одно из оптимальных решений $(i_1, i_2, \dots, i_n)$. Это доказывает теорему.

Заметим теперь, что существует оптимальное решение, в котором первый станок работает непрерывно в течение времени $\sum_1^n a_i$, а второй станок также работает непрерывно в течение времени $\sum_1^n b_i$. Для получения такого решения достаточно при оптимальной последовательности обработки деталей, начинать обработку каждой детали на первом станке в наиболее ранний возможный срок, а обработку на втором станке в наиболее поздний допустимый срок. Пусть T_{min} - минимальная продолжительность обработки всех деталей, $(i_1, i_2, \dots, i_n)$ - соответствующая очередность обработки. Имеем

$$T_{min} - \sum_{j=s}^{n} b_{ij} \geq \sum_{j=1}^{s} a_{ij}, \quad s = 1, 2, \dots, n \qquad (6.5)$$

Обозначим $T_{min} - \sum_1^n b_i = M_{min}$, $b_i = \alpha_i$, $a_i = \beta_i$, $i = 1, 2, \dots, n$

Тогда (6.5) принимает вид $M_{min} \geq \beta_i$,

$$M_{min} + \sum_{j=1}^{s-1} \alpha_{ij} \geq \beta_{i_s}, \quad s = 2, \dots, n, \qquad (6.6)$$

что полностью совпадает с условиями (6.3) задачи 6.I. Отсюда сразу получаем, что существует оптимальное решение, в котором сначала обрабатываются детали с $b_i \geq a_i$ в порядке возрастания a_i, а затем обрабатываются детали с $b_i < a_i$ в порядке убывания b_i

I.4. Обработка партии деталей

Результаты предыдущего пункта можно применить для решения более сложной задачи. Пусть детали обрабатываются партиями. По-прежнему считаем, что каждая деталь должна пройти

обработку сначала на первом станке, а затем на втором, и что имеется по одному станку каждого типа. Задача заключается в определении очередности обработки партий и очередности обработки деталей каждой партии, минимизирующих продолжительность обработки всех партий. Сначала решаем задачу Джонсона для деталей каждой партии. Пусть T_i - минимальная продолжительность обработки i-ой партии

$$A_i = \sum_{j=1}^{n_i} a_{ji}, \quad B_i = \sum_{j=1}^{n_i} b_{ji}, \text{ где } a_{ji}\ (b_{ji})$$

- продолжительность обработки j детали i-й партии на первом (втором) станке, n_i - число деталей i-ой партии.

Упражнение 6.I. Докажите, что теорема 6.4 остается справедливой для партий деталей, то есть существует оптимальное решение, в котором порядок обработки партий на I-ом станке совпадает с порядком обработки на 2-м станке. Кроме того, существует оптимальное решение, в котором оба станка работают непрерывно, первый - в течение времени $\sum_{i=1}^{\rho} A_i$, а второй $\sum_{i=1}^{\rho} B_i$, где ρ - число партий. Поэтому, если T_{min} - минимальное время обработки партии, а $(i_1, i_2, \ldots, i_\rho)$ - оптимальная очередность обработки партий, то выполняются неравенства

$$T_{i_1} \leq T_{min} - \sum_{i=1}^{\rho} B_i,$$

$$\sum_{j=1}^{s-1} A_{ij} + (T_{i_s} - B_{i_s}) \leq T_{min} - \sum_{s}^{\rho} B_{ij}, \quad s = 2, \ldots, \rho. \tag{6.7}$$

Если положить $M_{min} = T_{min} - \sum_{i=1}^{\rho} B_i$, $\gamma_i = B_i - A_i$,

$$\beta_i = T_i - B_i, \quad \alpha_i = T_i - A_i, \quad i = 1, 2, \ldots, n,$$

то условия (6.7) примут вид

$$M_{min} \geq \beta_{i_1},$$

$$M_{min} + \sum_{j=1}^{s-1} \alpha_{i_j} \geq \beta_{i_s}, \quad s = 2, \ldots, p,$$

что полностью совпадает с условиями (6.3) для задачи 6.I. Таким образом, существует оптимальное решение задачи обработки партий, в котором в первую очередь обрабатываются партии с $B_i \geq A_i$ в порядке возрастания $T_i - B_i$, а затем партии с $B_i \leq A_i$ в порядке убывания $T_i - A_i$.

Пример 6.2. Решим задачу обработки трех партий деталей, данные о которых приведены ниже.

№	I партия			II партия			III партия		
a_i	2	6	4	I	5	4	8	6	2
b_i	5	3	7	6	4	2	3	4	5

Решите самостоятельно задачу Джонсона для каждой партии и определите T_i, A_i и B_i, $i = 1,2,3$. Проверьте правильность решения по данным, приведенным ниже

i	T_i	B_i	A_i	T_i-B_i	T_i-A_i	B_i-A_i
I	I7	I5	I2	2	5	3
2	I3	I2	I0	1	3	2
3	I9	I2	I6	7	3	-4

Оптимальный порядок обработки партий (2,1,3). Вычисляем по формуле (6.4)

$$M_{min} = T_{min} - \sum_{i}^{3} B_i = \max(1; 2-2; 7-5) = 2.$$

Окончательно получаем

$$T_{min} = M_{min} + \sum_{i}^{3} B_i = 41.$$

В заключение пункта рассмотрим простой случай, когда каждая партия состоит из одинаковых деталей. Пусть число деталей настолько велико, что можно пренебречь временем обработки одной детали по сравнению с временем обработки партии. Имеем продолжительность обработки i -ой партии

$$T_i = max\,(n_i a_i + b_i,\ n_i b_i + a_i),$$

$$A_i = a_i n_i,\quad B_i = n_i b_i,\quad i = 1,2,\ldots,p.$$

Так как $A_i \gg b_i$ и $B_i \gg a_i$, то

$$T_i = max\,(A_i, B_i).$$

Заметим, что если $B_i \gg A_i$, то $T_i = B_i$, $T_i - B_i = 0$,
а если $B_i \leqslant A_i$, то $T_i = A_i$, $T_i - A_i = 0$.
Получаем простое правило формирования оптимального решения: сначала обрабатываются партии с $B_i \geqslant A_i$ (в любом порядке), а затем партии с $B_i \leqslant A_i$ (в любом порядке).

I.5. Задача редактора

В задаче редактора каждая деталь сначала обрабатывается на первом станке, затем на втором, а затем снова на первом. При этом $N_1 = 1$, $N_2 = n$, то есть на втором станке все детали могут обрабатываться одновременно. Обозначим t_i, ℓ_i и τ_i соответственно продолжительность обработки i -ой детали на первом, втором и снова на первом станках. Свое название задача получила благодаря следующей интерпретации: требуется выпустить n рукописей. Каждая рукопись проходит три стадии обработки: первичное редактирование, набор в типографии и вторичное редактирование. И первичное, и вторичное редактирование производится одним редактором. Возможности типографии достаточны для одновременного набора

всех рукописей. Требуется определить порядок обработки рукописей редактором, минимизирующий продолжительность выпуска всех рукописей. Другая интересная интерпретация связана с ремонтными работами; требуется отремонтировать n агрегатов. Ремонт каждого агрегата состоит из трех стадий: разработка, ремонт узлов и сборка. Разработка и сборка всех агрегатов производится одной бригадой. Ремонт производится в ремонтном цехе, возможности которого достаточны для одновременного ремонта узлов всех агрегатов. Требуется определить очередность разборки и сборки всех агрегатов, при которой продолжительность ремонта всех агрегатов минимальна.

Решим задачу редактора при следующем предположении: очередность первичной обработки (или разборки) совпадает с очередностью вторичной обработки (сборки). Пусть T_{min} минимальная продолжительность обработки, а $(i_1, i_2, \dots, i_n)$ – оптимальная очередность. Тогда имеет место следующее неравенство

$$\sum_{j=1}^{s} t_{ij} + l_{is} + \sum_{j=s}^{n} \tau_{ij} \le T_{min}, \quad s = 1,2,\dots,n \qquad (6.8)$$

Обозначим $M_{min} = T_{min} - \sum_{i=1}^{n} \tau_i$, $\gamma_i = \tau_i - t_i$

$\alpha_i = \tau_i + l_i$, $\beta_i = t_i + l_i$, $i = 1,2,\dots,n$

Тогда (6.8) принимает вид

$$M_{min} + \sum_{j=1}^{s-1} \gamma_{ij} \ge \beta_{is}, \quad s = 2,\dots,n$$

$$M_{min} \ge \beta_{i_1}$$

что полностью совпадает с (6.3).

Таким образом, существует оптимальное решение, в котором, в первую очередь обрабатываются рукописи с $\tau_i \geq t_i$ в порядке возрастания $t_i + \ell_i$, а затем - рукописи с $\tau_i < t_i$ в порядке убывания $\tau_i + \ell_i$

Упражнение 6.2. Поставьте и решите задачу обработки партий рукописей.

Упражнение 6.3. Решите задачу редактора, допуская разные очередности первичной и вторичной обработки рукописей.

I.6. Метод ветвей и границ

Возможны различные варианты применения метода ветвей и границ к задаче обработки деталей, отличающиеся способом получения оценки подмножеств, процедурой разбиения на подмножества, стратегией движения по дереву ветвлений и т.д. Рассмотрим один из возможных вариантов. Пусть имеется по одному станку каждого типа и последовательность прохождения станков одинакова для всех деталей (примем, что эта последовательность $1,2,\ldots,m$). Кроме того, будем считать, что очередность обработки деталей одинакова для всех станков (для случая двух станков выше было показано, что существует оптимальное решение, обладающее таким свойством; то же самое справедливо и для трех станков). Поэтому решением задачи является перестановка $(i_1, i_2, \ldots, i_n)$ номеров деталей, где i_j - номер детали, обрабатываемой в j-ю очередь. Обозначим $Q(i_1, i_2, \ldots, i_s)$ - подмножество решений, в котором первыми обрабатываются детали $i_1, i_2, \ldots, i_s$ в очередности их следования. Обозначим, далее, $T(i_1, i_2, \ldots, i_s)$ оценку снизу продолжительности обработки деталей для реше-

ний множества $Q(i_1, i_2, \ldots, i_s)$. Для получения этой оценки будем считать операцией A_{ij} обработку i -ой детали на j -ом станке. Построим сеть, вершины которой соответствуют операциям A_{ij} . Такая сеть для подмножества Q (I,2,3) (число деталей равно 5, число станков 4) изображена на рис.6.I. Горизонтальные дуги отражают порядок обработки каждой детали на станках.

Пунктирные дуги (A_{1j} , A_{2j}) и (A_{2j} , A_{3j}), j = = I,2,3,4, отражают очередность обработки деталей I,2 и 3 в подмножестве Q (I,2,3). Наконец, дуги (A_{3j} , A_{4j}) и (A_{3j} , A_{5j}), j = I,2,3,4.указывают, что детали 4 и 5 обрабатываются после детали 3. Аналогичным образом можно построить сеть для любого подмножества $Q(i_1, i_2, \ldots, i_s)$. Для получения оценки $T(i_1, i_2, \ldots, i_s)$ определим следующие величины:

а) ранние моменты $T^{o}_{i_s j}(i_1, i_2, \ldots, i_s)$ окончания операций

$$A_{i_s j}, \quad j = 1, 2, \ldots, m;$$

б) ранние моменты $T^{o}_{i_\kappa j}(i_1, i_2, \ldots i_s)$

окончания операций $A_{i_\kappa j}$, $\kappa = s+1, \ldots, n$, $j = 1,$

Вычисляем $T_{\kappa p}(i_1, i_2, \ldots, i_s) = \max\limits_{s<\kappa\leq n} T_{i_\kappa m}$

$$T^{o}_{i_\kappa, 0} = T^{o}_{i_s, 1}$$

$$\theta_j(i_1, i_2, \ldots, i_s) = \max\left[T^{o}_{i_s j}(i_1, i_2, \ldots, i_s); \min_{s<\kappa\leq n} T^{o}_{i_\kappa, j-1}\right], \quad j = 1, 2, \ldots, n$$

$$\sigma_j(i_1, i_2, \ldots, i_s) = \min_{s<\kappa\leq n} \sum_{p=j+1}^{m} t_{i_\kappa p}$$

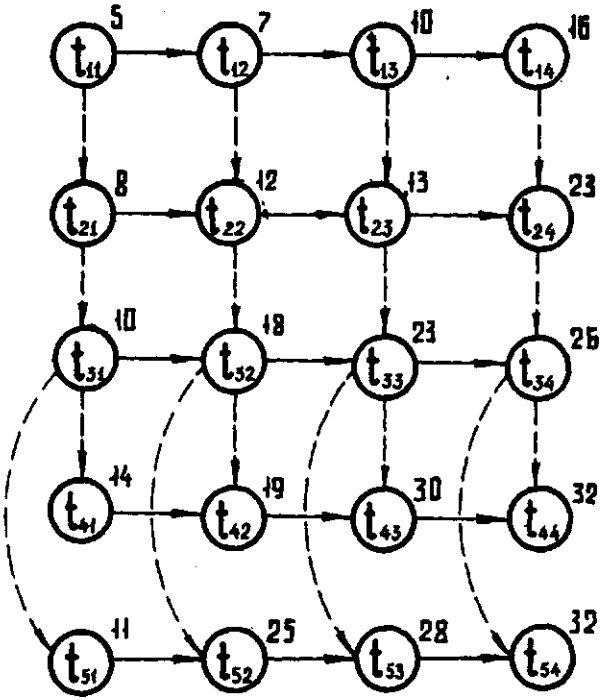


Рис. 6.1

Смысл величин θ_j и σ_j в том, что операции $M_{i_\kappa j}$, $\kappa = s+1, \dots, n$ не могут начаться раньше момента $\theta_j(i_1, i_2, \dots, i_s)$ а после выполнения этих операций должно пройти не менее $\sigma_j(i_1, i_2, \dots, i_s)$ до завершения всех операций. Поэтому

$$T_{min} \geqslant \theta_j(i_1, \dots, i_s) + \sum_{\kappa=s+1}^{n} t_{i_\kappa j} + \sigma_j(i_1, \dots, i_s)$$

Окончательно получаем оценку

$$T(i_1, i_2, \dots, i_s) = \max \Big[T_{кр}(i_1, i_2, \dots, i_s); \max_{1 \leqslant j \leqslant m} \Big(\theta_j(i_1, \dots, i_s) + \sum_{\kappa=s+1}^{n} t_{i_\kappa j} + \sigma_j(i_1, \dots, i_s) \Big) \Big]. \tag{6.9}$$

Пример 6.3. Получим оценку T (I,2,3) для сети, изображенной на рис.6.I. с данными (t_{ij}), приведенными ниже

$$(t_{ij}) = \begin{pmatrix} 5 & 2 & 3 & 6 \\ 3 & 4 & 1 & 7 \\ 2 & 6 & 5 & 3 \\ 4 & 1 & 7 & 2 \\ 1 & 7 & 3 & 4 \end{pmatrix}$$

Ранние моменты окончания всех операций, полученные в результате расчета сети приведены у соответствующих вершин на рис.6.I. Имеем

$$T_{кр}(1,2,3) = \max \left[T^{p}_{44}(1,2,3);\ T^{p}_{54}(1,2,3) \right] = 32;$$

$$\theta_1(1,2,3) = T^{p}_{31}(1,2,3) = 10;$$

$$\theta_2(1,2,3) = \max \left[T^{p}_{32}(1,2,3);\ \min \left(T^{p}_{41}(1,2,3);\ T^{p}_{51}(1,2,3) \right) \right] = 18;$$

$$\theta_3(1,2,3) = \max \left[T^{p}_{33}(1,2,3);\ \min \left(T^{p}_{42}(1,2,3);\ T^{p}_{52}(1,2,3) \right) \right] = 23;$$

$$\theta_4(1,2,3) = \max \left[T^{p}_{34}(1,2,3);\ \min \left(T^{p}_{43}(1,2,3);\ T^{p}_{53}(1,2,3) \right) \right] = 28.$$

Далее

$$\theta_1(1,2,3) = \min(t_{42} + t_{43} + t_{44};\ t_{52} + t_{53} + t_{54}) = 10,$$

$$\theta_2(1,2,3) = \min(t_{43} + t_{44};\ t_{53} + t_{54}) = 7,$$

$$\theta_3(1,2,3) = \min(t_{44},\ t_{54}) = 2,$$

$$\theta_4(1,2,3) = 0.$$

Окончательно получаем

$T(1,2,3) = \max[32;\ 10+5+10;\ 18+8+7;\ 23+10+2;\ 28+6+0] = 35.$

Пример 6.4. Решим методом ветвей и границ задачу с матрицей (t_{ij}), приведенной ниже

$$(t_{ij}) = \begin{pmatrix} 6 & 3 & 5 \\ 2 & 5 & 4 \\ 4 & 7 & 2 \\ 5 & 1 & 6 \end{pmatrix}$$

I шаг. оценки

$T(1) = 26,\quad T(2) = 24,\quad T(3) = 28,\quad T(4) = 25.$

Выбираем подмножество $Q(2)$, то есть в первую очередь обрабатываем вторую деталь.

II шаг. Определим оценку $T(2,1)$. Соответствующая сеть приведена на рис.6.2. Имеем

$$T_{кр}(2,1) = 22$$

j	1	2	3
$\theta_j(2,1)$	8	12	16
$\sigma_j(2,1)$	7	2	0

Окончательно получаем

$T(2,1) = \max(22,\ 8+9+7;\ 12+9+2;\ 16+8+0) = 24.$

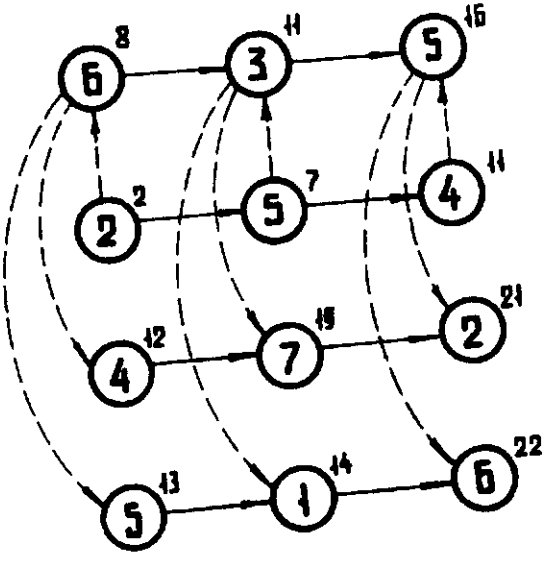


Рис. 6.2

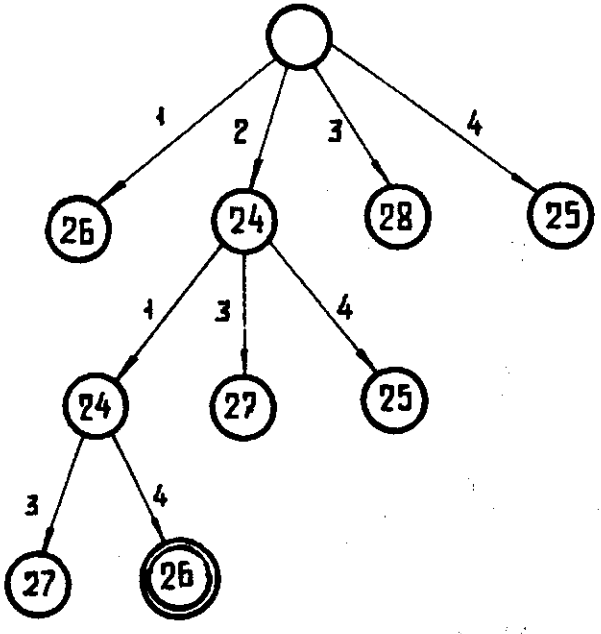


Рис. 6.3

Аналогично получаем

$T(2,3)=27, \quad T(2,4)=25$

Выбираем подмножество $Q(2,1)$ с минимальной оценкой.

III шаг. Подмножества $Q(1,2,3)$ и $Q(2,1,4)$ состоят из одного решения каждое. Имеем

$$T(2,1,3,4) = 27, \quad T(2,1,4,3)=26.$$

Дерево ветвлений изображено на рис.6.3. Полученное решение (2.1,4,3) с временем обработки 26 может не быть оптимальным, поскольку имеются подмножества $Q(4)$ и $Q(2,4)$, оценки которых равны 25. Поэтому следует провести дальнейшее разбиение этих подмножеств. Однако, мы не будем этого делать, а просто покажем, что оценки 25 не достижимы. Действительно, оценка $T(4) = 25$ получена сложением времен

$$t_{41} + t_{31} + t_{21} + t_{11} + t_{12} + t_{13}$$

Для того, чтобы эта оценка была достижима, необходимо последней обрабатывать деталь I (в противном случае, оценка будет 26). Однако, в этом случае

$\theta_2(4) = 7, \quad \sigma_2(4) = 5$ и

$$\theta_2(4) + t_{12} + t_{22} + t_{32} + \sigma_2(4) = 27$$

то есть оценка будет равна 27.

Оценка $T(2,4) = 25$ также недостижима. Таким образом, решение (2,1,4,3) является оптимальным.

I.7. Некоторые обобщения

Цеху требуется выполнить ряд операций, необходимая очередность выполнения которых определяется сетью (вершины сети соответствуют операциям, а дуги определяют зависимости

между операциями). Каждая операция выполняется на станке определенного типа. Таким образом, множество операций разбито на m классов (по числу различных типов станков). Обозначим P_j - множество операций, выполняемых на станках j -го типа, τ_i - продолжительность i -ой операции, N_j как и ранее, количество станков j -го типа. В отличие от задачи обработки деталей на станках, в которой сеть состоит из независимых путей, отражающих технологию обработки каждой детали, в данном случае сеть может иметь произвольный вид. Обозначим t_i'' - момент начала i -ой операции. Совокупность $\{t_i''\}$, $i = 1, 2, \ldots, n$ удовлетворяющая условиям

$$t_i'' \geq 0, \quad i = 1, 2, \ldots, n$$
$$t_j'' \geq t_i'' + \tau_i, \text{ если } (ij) \in U, \qquad (6.10)$$

называется календарным планом (расписанием). Обозначим, далее $Q_j(t)$ - множество операций, выполняемых на станках типа в момент t , то есть таких, что

$$t_i'' \leq t < t_i'' + \tau_i \text{ для } i \in Q_j(t)$$

(в данном случае предполагается, что прерывать выполнение начатой операции запрещается). План называется допустимым, если

$$|Q_j(t)| \leq N_j \quad j = 1, 2, \ldots, m \qquad (6.11)$$

то есть число выполняемых операций j -го класса в любой момент времени не превышает числа станков соответствующего типа.

Задача 6.2. Определить допустимый план минимальной продолжительности

$$T = \max_i (t_i'' + \tau_i) \qquad (6.12)$$

Задача 6.2 имеет много интересных интерпретаций (параллельное выполнение нескольких программ на вычислительных машинах, планирование строительных, проектных и других работ). К сожалению, в общем случае она не имеет эффективных методов решения. В предыдущей главе были рассмотрены алгоритмы решения задачи для случая упорядоченных событий (при изображении операций в виде дуг сети). На практике при решении подобных задач широко применяются эвристические алгоритмы, позволяющие получать достаточно хорошие решения в разумные сроки. Рассмотрим один такой алгоритм. Предварительно для каждой операции определим максимальную длину l_i пути, соединяющего данную операцию с одной из конечных операций сети, просчитывая сеть с конца (см.главу У). Затем рассматриваем операции поочередно в порядке уменьшения l_i и устанавливаем момент начала рассматриваемой операции на наиболее ранний возможный срок, допускаемый сетью и ограничениями на число станков, при фиксированных моментах начала ранее рассмотренных операций.

Пример 6.5. Применим описанный выше эвристический алгоритм для комплекса операций, изображенного на рис.6.4. Примем, что имеется по одному станку каждого типа (операции I-го класса изображены одинарными кружками, операции 2-го класса - двойными). Продолжительности операций приведены ниже

i	I	2	3	4	5	6	7
τ_i	2	I	4	3	4	3	4

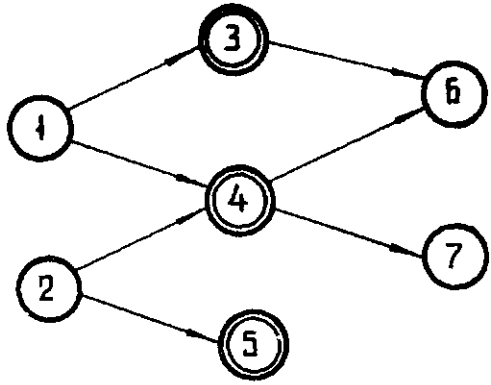


Рис. 6.4

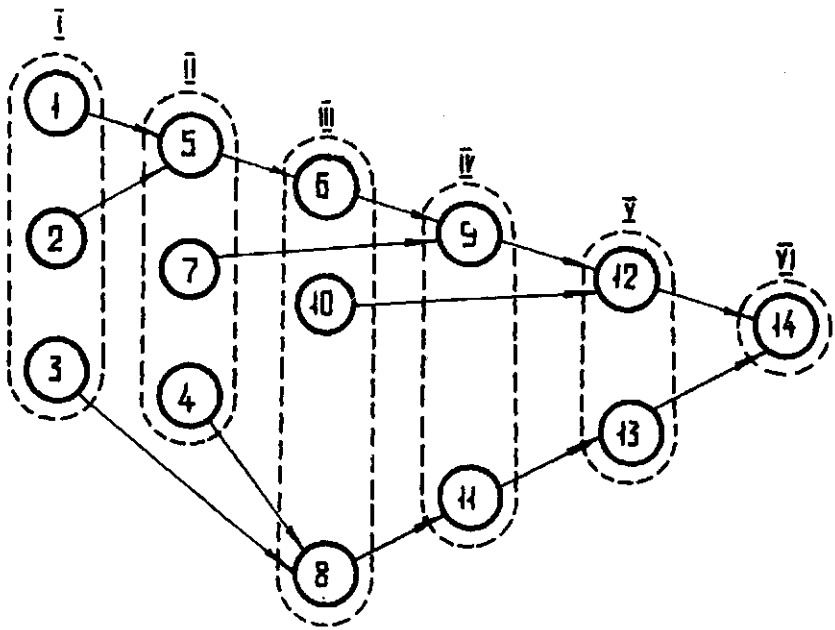


Рис. 6.5

Проверьте самостоятельно значения l_i, просчитав сеть с конца

i	I	2	3	4	5	6	7
l_i	I0	9	7	7	4	3	4

I. Рассматриваем операцию I так, как l_1 = IC. Полагаем $t_1^{н} = 0$.

П. Рассматриваем операцию 2, так как $l_2 = 9$. Полагаем $t_2^{н} = t_1^{н} + \tau_1 = 2$

Ш. Рассматриваем операцию 4, так как $l_4 = 7$. Полагаем $t_4^{н} = max\,(t_1^{н} + \tau_1;\ t_2^{н} + \tau_2) = 3$

IУ. Рассматриваем операцию 3, так как $l_3 = 7$. Полагаем $t_3^{н} = max\,(t_1^{н} + \tau_1;\ t_4^{н} + \tau_4) = 6$

У. Рассматриваем операцию 5, так как $l_5 = 4$ (можно операцию 7, так как l_4 также равно 4). Полагаем

$$t_5^{н} = max\,(t_2^{н} + \tau_2;\ t_3^{н} + \tau_3) = 10$$

УI. Рассматриваем операцию 7. Полагаем $t_7^{н} = t_4^{н} + \tau_4 = 6$

УП. Рассматриваем операцию 6 ($l_6 = 3$). Полагаем

$$t_6^{н} = max\,(t_3^{н} + \tau_3;\ t_7^{н} + \tau_7) = 10$$

Продолжительность комплекса

$$T = max\,(t_5^{н} + \tau_5;\ t_6^{н} + \tau_6) = 14$$

Часто применяется другой алгоритм, согласно которому в первую очередь из числа возможных начинаются операции с максимальным l_i. Чтобы понять отличие этого алгоритма от первого, применим его для планирования комплекса из примера 6.4.

Пример 6.6.

I. В момент t = 0 можно начинать операции I и 2. Начинаем операцию I, так как $\ell_1 > \ell_2$. Имеем $t_1'' = 0$.

II. В момент t = 2 операция I закончена. Можно начинать операции 2 и 3. Поскольку они выполняются на разных станках, начинаем обе. Имеем $t_2'' = 2$, $t_3'' = 2$ (заметим, что в предыдущем алгоритме $t_3'' = 6$).

III. В момент t = 3 операция 2 закончена.

IУ. В момент t = 6 операция 3 закончена. Можно начинать операции 4 и 5. Начинаем операцию 4, так как $\ell_4 > \ell_5$. Имеем $t_4'' = 6$.

У. В момент t = 9 операция 4 закончена. Можно начинать операции 5,6 и 7. Начинаем операции 5 и 7, так как $\ell_4 > \ell_5$. Имеем $t_5'' = 9$, $t_7'' = 9$.

УI. В момент t = I3. Операции 5 и 7 закончены. Имеем $t_6'' = 13$. Продолжительность комплекса

$$T = t_4'' + \tau_6 = 16$$

Упражнение 6.4. Примените описанные алгоритмы для сети, изображенной на рис.6.4 с данными, приведенными ниже

i	I	2	3	4	5	6	7
τ_i	I	3	4	3	4	3	4

Является ли лучшее из полученных решений оптимальным? Почему?

Замечание. Описанные алгоритмы без существенных изменений применимы в случае, когда каждая операция выполняется не с единичной, а с некоторой фиксированной интенсивностью β_i , а также при допущении перерывов операций.

В заключение рассмотрим частный случай, когда удается получить точное решение задачи. Пусть длительности всех операций одинаковы (примем их равными I), все операции выполняются на N одинаковых станках и сеть имеет вид обращенного прадерева (прадерево, у которого направления всех дуг изменены на обратные). Заметим, что в этом случае оба, рассмотренные выше, эвристических алгоритма эквивалентны (докажите самостоятельно).

Докажем, что они дают оптимальное решение. Пусть имеется оптимальное решение, в котором для δ пар вершин i,j имеет место $\ell_i > \ell_j$ и в то же время $t_i'' > t_j''$ (можно сказать, что имеется δ нарушений правила, согласно которому операции с большим значением ℓ_i начинаются не позже операций с меньшими значениями ℓ_i . Пусть i,j пара вершин, имеющая минимальное ℓ_j среди всех таких пар и такая, что $t_i'' = t_j'' + 1$ (покажите, что такая пара всегда существует). Тогда для вершины k , непосредственно следующей за вершиной j имеет место $t_k'' > t_j'' + 1$. Действительно, в противном случае для вершин S , непосредственно следующей за вершиной i имеет место $t_S'' > t_i'' = t_k''$ и $\ell_S = t_i - 1 > t_j - 1 = t_k$, то-есть для пары k, S имеет место нарушение правила, и в то же время $\ell_k < \ell_j$, что противоречит условию выбора пары (i,j) . Поэтому, если положить $\tilde{t}_i'' = t_j''$, $\tilde{t}_j'' = t_i''$, то изменения начал других операций не произойдет, а число нарушений δ уменьшится на I. Продолжая таким образом, получим оптимальное решение без нарушений правила. Аналогичные рассуждения позволяют убедиться, что все решения,

удовлетворяющие условию

$$t_i'' \le t_j'' \qquad \text{если} \qquad l_i > l_j$$

имеют одинаковую продолжительность, и следовательно, являются оптимальными (естественно при условии начала операций в наиболее ранние сроки, допускаемые сетью и ограничениями на число станков).

Пример 6.7. Определим минимальную продолжительность комплекса, изображенного на рис.6.5 при наличии трех станков. Значения l_i приведены ниже

i	I	2	3	4	5	6	7	8	9	I0	II	I2	I3	I4
l_i	6	6	5	5	5	4	4	4	3	3	3	2	2	I

Пунктиром на рис.6.5. выделены множества операций, выполняемые в каждую единицу времени. Минимальная продолжительность комплекса равна 6.

§ 2. Динамическая задача назначения

2.I. Постановка задачи

В предыдущем параграфе рассматривались задачи календарного планирования, в которых каждая машина (рабочий, станок и т.п.) могла выполнять только определенный класс операций. В более общем случае каждая машина может выполнять различные операции, но с разной производительностью. Обозначим τ_{ij} - продолжительность i-ой операции, если она выполняется на j-ой машине. Пусть m машин назначены для выполнения комплекса из n операций.

Задача 6.3. Определить множества операций, выполняемых на каждой машине и порядок их выполнения на соответствующей машине, так чтобы продолжительность комплекса была минималь-

ной (прерывать начатые операции запрещено).

Если множества операций, выполняемых на каждой машине, определены, то задача 6.3 переходит в задачу 6.2.

Задача 6.3 как и задача 6.2 является типичной экстремальной комбинаторной задачей, и эффективные точные методы ее решения, в общем случае, отсутствуют.

2.2. Назначение по одной операции

Пусть операции комплекса независимы, и число операций равно числу машин. Примем далее, что каждая машина выполняет одну и только одну операцию. Обозначим $x_{ij} = 1$, если i-ая операция выполняется на j-ой машине и $x_{ij} = 0$, в противном случае. Получаем следующую задачу.

Задача 6.4. Определить $\{x_{ij}\}$ удовлетворяющие условиям

$$\sum_{i=1}^{n} x_{ij} = 1$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad (6.13)$$

и минимизирующие

$$\max_{i} \sum_{j=1}^{n} x_{ij} \tau_{ij} \qquad (6.14)$$

Описание алгоритма. Вычислим

$$T_0 = \max \left[\max_i \min_j \tau_{ij} ; \max_j \min_i \tau_{ij} \right]$$

Определим сеть с вершинами $x_0, \bar{z}, i, j, \ (i,j = 1,2,\dots,n)$

Вершину x_0 соединим с каждой вершиной i дугой (x_0, i)

единичной пропускной способности. Вершину i соединяем с вершиной j дугой (i , j)единичной пропускной способности, если $\tau_{ij} \le T_0$ каждую вершину j соединяем с вершиной z дугой (j, z) единичной пропускной способности.

Определим поток максимальной величины в полученной сети [2,5] Если величина потока $\gamma_{max} = n$, то потоки по дугам (i , j)определят оптимальные назначения операций на машины. Пусть $\gamma_{max} < n$. Обозначим Q - множество помеченных вершин i , R - множество непомеченных вершин j . Вычисляем

$$T_1 = \min_{\substack{i \in Q \\ j \in R}} \tau_{ij}$$

Добавляем в сеть дуги (i , j), такие что $\tau_{ij} < T_1$ и увеличиваем поток. Продолжая таким образом, за конечное число шагов получаем поток величины n , который и определит оптимальное решение задачи. Обоснование алгоритма элементарно. Предлагаем сделать это самостоятельно.

Пример 6.8. Данные о продолжительностях операций приведены ниже

$$(\tau_{ij}) = \begin{pmatrix} 3 & 5 & 4 & 7 \\ 2 & 1 & 2 & 1 \\ 1 & 6 & 9 & 8 \\ 2 & 7 & 8 & 9 \end{pmatrix}$$

Выполняем $T_0 = max$ (3; 1; 1; 2; 1; 1; 2; 1) = 3.
Сеть с дугами (i, j) , для которых $\tau_{ij} \le 3$ изображена на рис.6.6 (поток максимальной величины γ_{max} = 2). Множество Q состоит из вершин 1, 3, 4, множество R - из

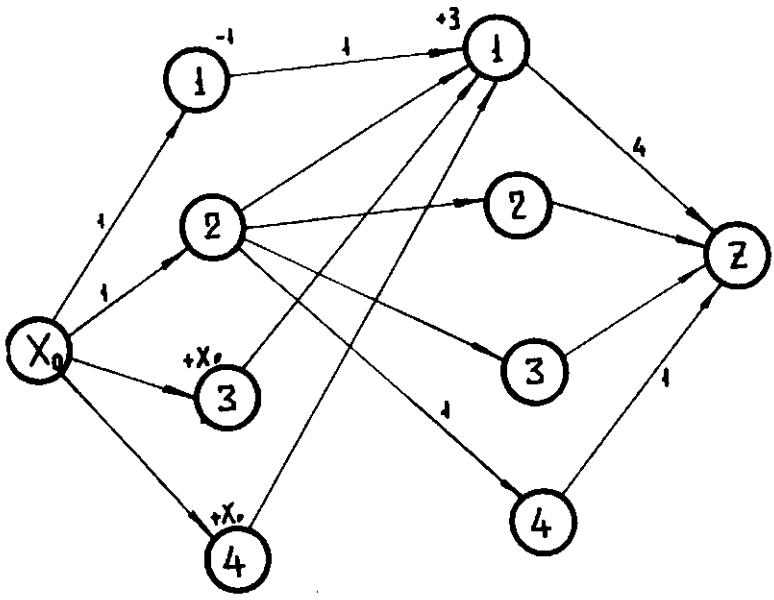


Рис. 6.6

вершин 2,3,4. Определяем

$$T_1 = min(5,4,7,6,9,8,7,8,9) = 6.$$

Включаем в сеть дугу (I.3). Новый поток максимальной величины показан на рис.6.7. Имеем $Q = \{3,4\}$, $R = \{2,3,4\}$,

$$T_2 = min(6,9,8,7,8,9) = 6.$$

Включаем в сеть дугу (3,2). Проверьте самостоятельно, что поток максимальной величины в полученной сети равен 4. Оптимальное решение задачи $X_{13} = X_{24} = X_{32} = X_{41} = 1$, остальные $X_{ij} = 0$.

Описанный алгоритм можно применить и для решения задачи минимизации отклонения сроков окончания работ от заданных сроков. Пусть T_i - заданный срок окончания i-ой работы. Задача заключается в минимизации

$$\max_i \left(\sum_j X_{ij} \tau_{ij} - T_i \right) \qquad (6.15)$$

при ограничениях (6.I3). Эта задача эквивалентна задаче 6.4 с матрицей

$$\tau'_{ij} = \tau_{ij} - T_i, \quad i,j = 1,2,\dots,n$$

Действительно

$$\sum_{j=1}^{n} \tau'_{ij} X_{ij} = \sum_{j=1}^{n} \tau_{ij} \cdot X_{ij} - \sum_{j=1}^{n} X_{ij} T_i = \sum_{j=1}^{n} \tau_{ij} X_{ij} - T_i$$

2.3 Эвристический алгоритм

Опишем простой алгоритм решения задачи 6.3, который является,по существу,развитием эвристических алгоритмов решения задачи 6.2. Предварительно определяем для каждой операции минимальную продолжительность

$$\tau_i = \min_j \tau_{ij}$$

Положив продолжительность каждой операции равной τ_i, вычисляем длину ℓ_i максимального пути,соединяющего i-ю операцию с одной из конечных операций ($i = 1,2, \dots, n$). Операции рассматриваются в очередности убывания ℓ_i. При рассмотрении i-ой операции моменты начала ранее рассмот-

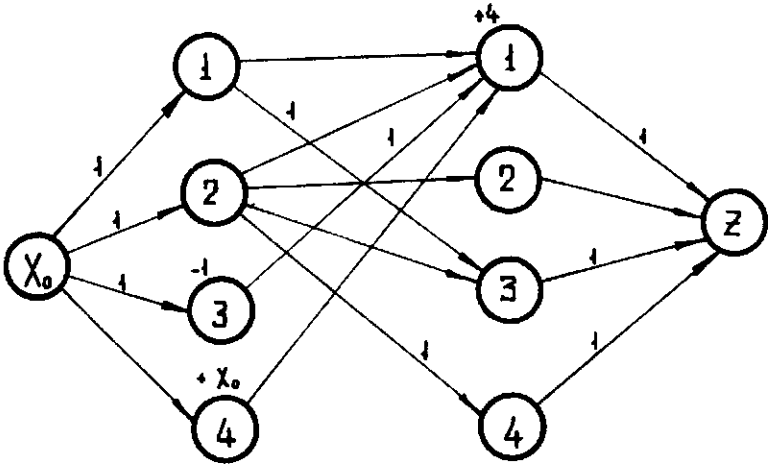


Рис. 6.7

ренных операций, фиксируются. Поэтому можно определить ранний момент θ_{ij} возможного начала i -ой операции на j - ом станке.

Вычисляем $min\,(\theta_{ij} + \tau_{ij}) = \theta_{ij_1} + \tau_{ij_1}$,

Полагаем $t_i'' = \theta_{ij_1}$,

и назначаем i -ю операцию на соответствующий станок j_1.

Пример 6.9. Решим задачу 6.I3 для сети, изображенной на рис.6.8 при наличии двух станков. Матрица (τ_{ij}) приведена ниже

$j \backslash i$	I	2	3	4	5	6	7
I	5	7	2	3	9	I	6
2	3	5	4	8	4	3	2

Имеем $\tau_1 = 3$, $\tau_2 = 5$, $\tau_3 = 2$, $\tau_4 = 3$, $\tau_5 = 4$, $\tau_6 = I$, $\tau_7 = 2$.

Просчитывая сеть с конца, определяем l_i :

i	I	2	3	4	5	6	7
l_i	8	II	3	5	6	I	2

I. Рассматриваем операцию 2 с максимальным l_i

Полагаем $t_2'' = 0$, операция выполняется на втором станке.

II. Рассматриваем операцию I.

Имеем $\theta_{11} = 0$, $\theta_{12} = 5$, $min(5;\ 5+3) = 5$.

Полагаем $t_1'' = 0$ операция выполняется на первом станке.

III. Рассматриваем операцию 5.

Имеем $\theta_{51} = 5$, $\theta_{52} = 5$, $min\,(5+9,\ 5+4) = 9$.

Полагаем $t_5'' = 5$;

операция выполняется на втором станке.

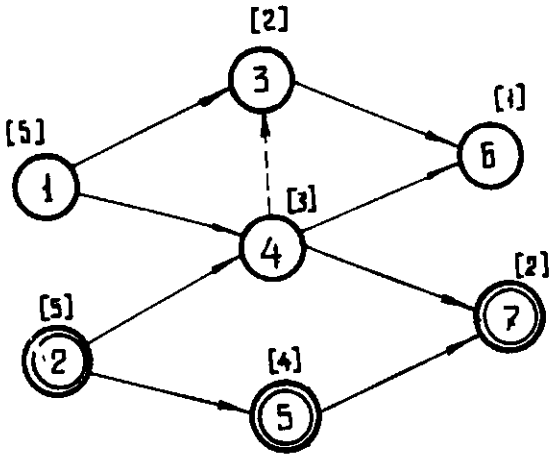


Рис. 6.8

IУ. Рассматриваем операцию 4.

Имеем $\theta_{41} = 5$, $\theta_{42} = 9$, $min\ (5+3,\ 9+8) = 8$.

Полагаем $t_4'' = 5$, операция выполняется на первом станке.

У. Рассматриваем операцию 3.

Имеем $\theta_{31} = 8$, $\theta_{32} = 9$, $min\ (8+2;\ 9+4) = 10$.

Полагаем $t_3'' = 8$, операция выполняется на первом станке.

УI. Рассматриваем операцию 7.

Имеем $\theta_{71} = 10$, $\theta_{72} = 9$, $min\ (10+6;\ 9+2) = 11$

Полагаем $t_7'' = 9$, операция выполняется на втором станке.

УП. Рассматриваем операцию 6.

Имеем $\theta_{61} = 10$, $\theta_{62} = 11$, $min\ (10+1;\ 11+3) = 11$.

Полагаем $t_6'' = 10$, операция выполняется на первом станке.

Полученное решение с временем выполнения комплекса $T = 11$.является в данном случае оптимальным, так как $l_2 = 11$.

На рис.6.8 двойными кружками отмечены операции, выполняемые на втором станке, продолжительности выполнения операций на соответствующих станках указаны в квадратных скобках (пунктирная дуга (4,3) указывает, что операция 4 выполняется на первом станке раньше, чем операция 3.

Упражнение 6.5. Разработайте схему метода ветвей и границ для задачи 6.3.

§ 3. Задачи переналадки станков

3.1. Постановка задачи

Когда на одном и том же станке выполняются операции различного вида, то при переходе от операций одного вида к операциям другого вида требуется определенное время на

переналадку станка. Аналогичная ситуация возникает при выполнении разнотипных операций одним специалистом, при съемках различных фильмов на одной площадке (требуется время на подготовку декораций), при перевозке грузов одной машиной (судном и т.д.) на разных маршрутах (требуется время на переброску машины с одного маршрута на другой). Время переналадки можно интерпретировать и как затраты, связанные с выполнением на одном оборудовании различных работ. Отметим, например, задачу определения последовательности перекачки нефтепродуктов по трубопроводу. Если продукты i,j перекачиваются один за другим, то возникают потери S_{ij} из-за смешивания нефтепродуктов (эта задача была рассмотрена Ф.Нуриевым для случая $S_{ij} = |C_i - C_j|$, где C_i, C_j - стоимости нефтепродуктов i,j).

Задача 6.4. Требуется обработать n деталей. Продолжительность обработки i -ой детали - T_i $(i = 1,2, \ldots, n)$. Имеется m одинаковых станков. Время переналадки станка (любого) после выполнения i -ой операции на выполнение j -ой операции равно t_{ij} . Распределить детали по станкам и упорядочить обработку деталей на каждом станке, таким образом, чтобы минимизировать время обработки всех деталей (при этом, перерыв в обработке деталей может быть запрещен для всех деталей, либо для некоторых; либо разрешен для всех).

Обозначим i_j - начальное состояние j -го станка (то есть перед началом работ станок j налажен на обработку деталей i_j). Определим $(n+m)$ - вершинный граф, множество P вершин которого соответствует деталям, а

множество Q станкам. Две вершины $i, k \in P$ соединим дугой (i,k) длины $l_{ik} = \tau_k + t_{ik}$ $(i,k = 1,2,\dots,n)$. Две вершины $j \in P$, $i \in Q$ соединим дугой (j,i) длины

$$l_{ji} = \begin{cases} 0 & \text{, если } i_j = i, \\ t_{ji} & \text{, если } i_j \neq i \end{cases}$$

Задача 6.4 заключается в определении в полученном графе множества из m путей, покрывающих граф (множество путей покрывает граф, если каждая вершина графа принадлежит одному и только одному пути), таких что длиннейший из них имеет минимальную длину. В частности, при $m = 1$, получаем задачу определения кратчайшего гамильтонова пути.

3.2. Обработка двух деталей на каждом станке

Пусть на каждом станке допускается обработка не более двух деталей. Примем, что $n = 2m$ (если n - нечетное, то добавляем фиктивную деталь с нулевой продолжительностью обработки и нулевыми временами переналадки). В этом случае на каждом станке будут обрабатываться ровно две детали. Далее, для упрощения задачи будем считать, что все станки находятся в одном и том же начальном состоянии (обозначим это состояние 0, а время переналадки станка из этого состояния для обработки детали i обозначим t_{0i}, $i = 1,2,\dots,n$. Определим полный n - вершинный граф G с длинами дуг.

$$l_{ij} = \tau_i + \tau_j + \min(t_{0i} + t_{ij}, t_{0j} + t_{ji}), \quad i,j = 1,2,\dots,n.$$

Заметим, что если пара деталей i,j обрабатывается на одном станке, то l_{ij} равно продолжительности обработки обеих деталей с учетом времени переналадок. Любому

решению задачи будет соответствовать некоторое паросочетание X графа G

При этом, продолжительность обработки всех деталей

$$T = \max_{(i,j) \in X} t_{ij}$$

Опишем алгоритм решения задачи, использующий метод чередующихся цепей. Определяем

$$T_0 = \max_i \min_j t_{ij}$$

и удаляем из графа G ребра с длинами $t_{ij} > T_0$. В оставшемся графе определяем максимальное паросочетание (паросочетание с максимальным числом ребер). Если число дуг паросочетания равно m, то оно определяет оптимальное решение задачи 5.4.I. Если число дуг максимального паросочетания меньше m, то определяем ребра с минимальной длиной T_1 из числа удаленных ребер, и добавляем в граф все такие ребра. Продолжая подобным образом за конечное число шагов, находим минимальное T, при котором число дуг максимального паросочетания будет равно m (такое паросочетание называется совершенным). Это паросочетание определяет оптимальное решение задачи.

Упражнение 6.6. На основе теоремы 4.2 главы 4 предложите алгоритм определения максимального паросочетания.

3.3. Задача развозки грузов

Приведем постановку задачи развозки грузов, которая близка к задаче 6.4, отличаясь от нее рядом дополнительных условий. Имеется n пунктов потребления и m

пунктов снабжения. Заданы расстояния между всеми пунктами, количество продукции в каждом пункте снабжения и потребность в каждом пункте потребления (предполагается, что суммарное количество продукции в пунктах снабжения не меньше суммарной потребности). Развозка продукции производится k машинами грузоподъемностью C. В начальный момент $t = 0$ известно расположение машин по пунктам (если машина находится в дороге, то известно, в какой пункт она направляется). Кроме того, заданы нормы погрузки и нормы разгрузки грузов. Требуется определить маршрут каждой машины так, чтобы все грузы были доставлены потребителям за минимальный срок. Если задан срок доставки грузов, можно поставить задачу минимизации числа машин, либо выбора наиболее оптимальной грузоподъемности машин и т.д. Ограничиваясь описательной постановкой, отметим, что задачи такого типа очень сложны и не имеют эффективных алгоритмов решения, хотя применение общих методов решения комбинаторных задач с учетом специфики конкретных постановок позволяет получать решения, удовлетворительные, с точки зрения практического применения.

Упражнение 6.7. Решите задачу развозки грузов, предполагая, что все пункты снабжения и потребления расположены по одной дороге (то есть расстояние между любыми двумя пунктами i, j равно $|z_i - z_j|$, где z_i, z_j – координаты пунктов вдоль дороги). Предполагается, что величина доставляемой партии может быть любым положительным числом (не большим, конечно, грузоподъемности машин). Все числа считайте целыми.

П Р И Л О Ж Е Н И Е

Поскольку большинство излагаемых в предыдущих главах экстремальных задач носит комбинаторный характер, а понимание алгоритмов решения некоторых из них требует знания соотношения двойственности в линейном программировании, в настоящее приложение введены главы, знакомящие читателя с методами решения комбинаторных и основами теории двойственности в линейном программировании.

Глава УП.

ГРАФЫ И ЭКСТРЕМАЛЬНЫЕ КОМБИНАТОРНЫЕ ЗАДАЧИ

§ I. Экстремальные задачи комбинаторного типа

Многие прикладные задачи теории графов носят комбинаторный характер. В данной главе мы кратко рассмотрим основные общие методы решения экстремальных комбинаторных задач с тем,чтобы использовать эти методы при рассмотрении конкретных постановок.

Рассмотрим некоторое множество P комбинаций. Такими комбинациями могут быть перестановки из n элементов (число возможных комбинаций $N = n$), сочетания из n элементов (число возможных сочетаний $N = 2^n$, включая сочетание, не содержащее ни одного элемента), последовательности длины n , каждый член которых принимает одно из m возможных значений (число таких последовательностей $N = m^n$) и другие. Пусть на множестве P задана функция $f(\pi)$, то есть задан способ вычисления значения

$f(T)$ для любой комбинации T из множества P .

Задача 7.I. Определить комбинацию $T_0 \in P$, такую, что $f(T_0)$ принимает максимальное (или минимальное) значение.

Комбинация T_0 называется оптимальной. Самый простой способ решения задачи 7. это перебор всех возможных комбинаций и определение таким путем оптимальной. Однако, характерным для комбинаторных задач является быстрый рост числа возможных комбинаций с ростом числа элементов. Пусть, например, вы можете вычислять I00 значений $f(T)$ в секунду (конечно, с помощью вычислительной машины) и машина работает непрерывно 24 часа в день 365 дней в году. Тогда для перебора всех возможных $n!$ перестановок вам потребуется около 7 сек при $n = 6$, I0 час при $n = I0$, уже 2 года при $n = I5$ и более 400 лет при $n = I5$.

Таким образом, уже при небольшом числе элементов, мы получаем число возможных комбинаций, практически не доступное полному перебору даже на современных вычислительных машинах. Поэтому возникла необходимость разработки специальных методов, позволяющих получить если не оптимальное, то вполне приемлемое для практики, решение задачи в разумные сроки. Мы рассмотрим два таких достаточно общих методов: метод локальной оптимизации и метод ветвлений. Методы будем иллюстрировать на примере типичной экстремальной комбинаторной задачи.

Задача 7.2. Задан полный симметрический граф с длинами дуг l_{ij} , $i,j = 0, 1, 2, \ldots, n$

Определить гамильтонов контур, имеющий минимальную длину. Примем вершину 0 за начальную. Тогда любая перестановка n чисел $\pi = (i_1, i_2, \ldots, i_n)$ определит гамильтонов контур $(0, i_1, i_2, \ldots, i_n, 0)$ длины

$$f(\pi) = \sum_{k=0}^{n} l_{i_k i_{k+1}}, \qquad (7.1)$$

где $i_0 = i_{n+1} = 0$.

Таким образом, задача заключается в определении перестановки n чисел, минимизирующей функцию (7.1). Эта задача имеет много интересных интерпретаций. Приведем некоторые из них.

Задача коммивояжера. Коммивояжер должен объехать $(n+1)$ городов, начиная с города 0 и заканчивая маршрут в том же городе. Стоимость проезда между городами i и j равна l_{ij}. Определить самый дешевый маршрут.

Задача переналадки оборудования. Требуется обработать n деталей на одном станке. Одновременно на станке можно обрабатывать только одну деталь. Пусть станок находится в некотором начальном состоянии 0. Обозначим l_{ij} – время переналадки станка на обработку j-ой детали, если перед этим на нем обрабатывалась i-я деталь (l_{0j} – время наладки станка из начального состояния для обработки j-ой детали, l_{i0} – время приведения станка в первоначальное состояние после обработки i-ой детали). Определить очередность обработки деталей, при которой общее время переналадок минимально (после обработки всех деталей станок должен быть возвращен в начальное состояние).

§ 2. Метод локальной оптимизации

Определим для каждого решения (комбинации) T_i множество Q_i соседних с ним: множество Q_i называется окрестностью решения T_i. Процедура локальной оптимизации начинается с произвольной начальной комбинации T_0. Перебираем все решения, соседние с T_0, то есть все решения из множества Q_0. Вычисляем $f(T_1) = \min_{T \in Q_0} f(T)$. Если $f(T_1) < f(T_0)$, то принимаем T_1 за новое начальное решение и повторяем процедуру. Если $f(T_1) \geq f(T_0)$, то получено локально-оптимальное решение. Поскольку нет уверенности, что полученное решение дает абсолютный минимум, то процедуру можно повторить, либо взяв другое начальное решение, либо по другому определяя соседние решения. По существу метод локальной оптимизации является методом упорядоченного перебора, если множества соседних решений определяются произвольным образом. Однако часто анализ задачи подсказывает целесообразный выбор понятия соседства решений, позволяющий существенно сократить перебор.

Определим граф с N вершинами, где N - число возможных комбинаций. Вершина i графа соответствует комбинации T_i. Вершины i, j графа соединим дугой (i, j), если T_j является соседним с T_i, то есть $T_j \in Q_i$. Полученный граф называется <u>графом соседства комбинаций</u>.

Пример 7.I. Применим метод локальной оптимизации для решения задачи при $n = 4$. Матрица длин l_{ij} приведена ниже

i \ j	0	I	2	3	4
0	X	2	I	3	2
I	5	X	3	5	3
2	2	4	X	2	I
3	I	2	3	X	6
4	5	I	2	3	X

Будем считать соседними две перестановки, получаемые одна из другой путем транспозиции (транспозиций называется изменение порядка двух соседних элементов перестановки). В этом случае граф соседства будет симметрическим. Так перестановка $\mathcal{T}_0 = (I,2,3,4)$ имеет три соседних $\mathcal{T}_1 = (2,I,3,4)$, $\mathcal{T}_2 = (I,3,2,4)$ и $\mathcal{T}_3 = (I,2,4,3)$, то есть $Q_0 = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$
Вычисляем:

$$f(\mathcal{T}_0) = l_{01} + l_{12} + l_{23} + l_{34} + l_{40} = 18,$$
$$f(\mathcal{T}_1) = l_{02} + l_{21} + l_{13} + l_{34} + l_{40} = 21,$$
$$f(\mathcal{T}_2) = l_{01} + l_{13} + l_{32} + l_{24} + l_{40} = 16,$$
$$f(\mathcal{T}_3) = l_{01} + l_{12} + l_{24} + l_{43} + l_{30} = 10.$$

Так как $f(\mathcal{T}_3) = \min[f(\mathcal{T}_1), f(\mathcal{T}_2), f(\mathcal{T}_3)] = 10 < 18 = f(\mathcal{T}_0)$ то переходим к перестановке $\mathcal{T}_3$.
У перестановки $\mathcal{T}_3$ только две новых соседних $\mathcal{T}_4 = (2,I,4,3)$ и $\mathcal{T}_5 = (I,4,2,3)$. Имеем $f(\mathcal{T}_4) = I2$, $f(\mathcal{T}_5) = I0$.
Так как $f(\mathcal{T}_4) = 12 > f(\mathcal{T}_5) = f(\mathcal{T}_3) = 10$, то $\mathcal{T}_3$ является локально-оптимальным решением.

Упражнение 7.I. Повторите процедуру локальной оптимизации, взяв начальную перестановку (4,3,2,I).

Часть графа соседства решений показана на рис.7.I.

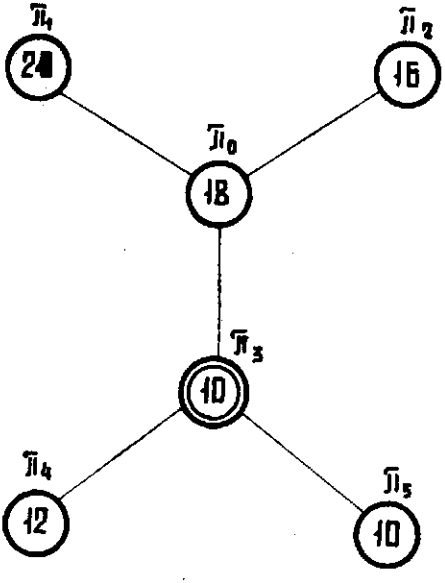


Рис. 7.1

Числа в вершинах равны $f(\mathcal{P})$ соответствующей комбинации. Изменим определение соседних решений. Будем считать окрестностью перестановки $\mathcal{P}$ множество перестановок, получаемых из $\mathcal{P}$ помещением первого элемента на второе, третье и четвертое места. В этом случае перестановка $\mathcal{P}_1 = (1,2,4,3)$ будет иметь соседними $\mathcal{P}_2 = (2,1,4,3)$, $\mathcal{P}_4 = (2,4,1,3)$ и $\mathcal{P}_7 = (2,4,3,1)$. Заметим, что при таком определении окрестностей граф соседства решений уже не будет симметрическим (например, $\mathcal{P}_3$ не является соседней с $\mathcal{P}_4$, хотя $\mathcal{P}_4$ соседняя с $\mathcal{P}_3$).

Вычисляем $f(\mathcal{P}_6) = 9$, $f(\mathcal{P}_1) = 12$. Так как $f(\mathcal{P}_6) = 9 < f(\mathcal{P}_1)$, то переходим к $\mathcal{P}_6$.

$\mathcal{P}_6$ – локально-оптимальное решение. Ниже мы убедимся, что $\mathcal{P}_6$ является абсолютным минимум.

§ 3. Метод ветвлений

3.1. Описание метода

Разобьем множество P всех решений на подмножества, каждое подмножество снова разобьем на более мелкие подмножества и т.д. пока не получим отдельные решения. Такое разбиение удобно изображать в виде дерева ветвлений. Вершины дерева соответствуют подмножествам, висячие вершины – отдельным решениям. От каждой не висячей вершины проводятся дуги к вершинам, соответствующим подмножествам, на которые разбивается данное.

Пример 7.2. Множество всех перестановок из четырех элементов можно разбивать на подмножества следующим образом. Первое подмножество содержит все перестановки, у кото-

Рис. 7.2

рых на первом месте стоит I, второе подмножество содержит все перестановки, в которых на первом месте стоит 2, и т.д. Каждое из этих подмножеств в свою очередь можно разбить на три подмножества, отличающихся элементом, стоящим на втором месте перестановки и т.д. Дерево ветвлений, при таком разбиении показано на рис.7.2. Будем обозначать $(i_1, i_2, \ldots, i_s, ?)$ подмножество перестановок, имеющих на первых s местах соответственно элементы $i_1, i_2, \ldots, i_s$. Заметим, что при $n = 4$ подмножества $(i_1, i_2, i_3, ?)$ состоят из одного элемента. Действительно, подмножество (I,2,3,?) например, содержит всего одну перестановку (I,2,3,4).

Если теперь определить правила, позволяющие в каждой вершине дерева ветвлений, выбирать направление движения, то применяя последовательно эти правила, мы придем к висячей вершине, то есть получим некоторое решение. Чаще всего определяется некоторая количественная оценка каждого подмножества и на очередном шаге выбирается подмножество с минимальной оценкой (или максимальной оценкой).Эффективность применения метода ветвлений во многом зависит от способа оценки подмножеств. Определение хорошего способа оценки подмножеств требует тщательного анализа задачи, учета ее специфики и в настоящее время во многом зависит от искусства исследователя. Итак, для применения метода ветвлений необходимо, во-первых, определить способ разбиения на подмножества, во-вторых, способ оценки подмножеств, и наконец, в-третьих, стратегию движений по дереву ветвлений. Под стратегией движения по дереву ветвлений будем понимать правила,

позволяющие выбрать следующее подмножество для разбиения. Наиболее часто применяются следующие две стратегии. В первой стратегии для очередного разбиения выбирается подмножество с минимальной оценкой, полученное в результате разбиения на предыдущем шаге. Во второй стратегии для очередного разбиения выбирается подмножество, имеющее минимальную оценку среди всех подмножеств, оцененных на предыдущих шагах. Достоинством первой стратегии является быстрое получение допустимого решения. Возможны и более сложные стратегии (например, чередование первой и второй).

Пример 7.3. Применим метод ветвлений для решений задачи, рассмотренной в примере 7.I. Возьмем способ разбиения на подмножества, рассмотренный в примере 7.2. Оценкой подмножества $(i_1, i_2, \ldots, i_s, ?)$ будем считать $\sum_{j=0}^{s-1} l_{i_j i_{j+1}}$, $i_0 = 0$. Воспользуемся первой стратегией движения по дереву ветвлений.

I. Подмножества $\{1,?\}$, $\{2,?\}$, $\{3,?\}$, $\{4,?\}$ имеют оценки соответственно $\gamma(1) = l_{01} = 2$, $\gamma(2) = l_{02} = 1$, $\gamma(3) = l_{03} = 3$, $\gamma(4) = l_{04} = 2$. Выбираем подмножество $\{2,?\}$ с минимальной оценкой.

2. Разбиваем подмножество $\{2,?\}$ на три подмножества $\{2,1,?\}$, $\{2,3,?\}$, $\{2,4,?\}$ с оценками $\gamma(2,1) = 5$, $\gamma(2,3) = 3$, $\gamma(2,4) = 2$. Выбираем подмножество $\{2,4,?\}$.

3. Разбиваем подмножество $\{2,4,?\}$ на два (2,4,1,3) и (2,4,3,1), которые уже являются решениями. Имеем $f(2,4,1,3) = 9$, $f(2,4,3,1) = 12$. Полученное решение (2,4,1,3) является в данном случае оптимальным.

3.2. Метод ветвей и границ

Иногда удается достаточно просто получать оценку снизу (нижнюю границу) функции $f(\mathcal{I})$ для любого подмножества. В этом случае целесообразно взять нижнюю границу для оценки подмножеств в методе ветвлений. Метод ветвлений с нижней границей в качестве функции оценки получил название метода ветвей и границ.

Обозначим Q семейство подмножеств, уже оцененных, но еще не разбитых на более мелкие подмножества. Если среди этих подмножеств имеется подмножество с минимальной оценкой, состоящее из одной комбинации, то эта комбинация является оптимальным решением задачи. Это очевидное свойство метода ветвей и границ является его важным преимуществом. Пусть $\mathcal{I}_1$ лучшее из уже полученных решений, а $\mathcal{E}$ - минимальная оценка подмножеств семейства Q. Тогда, если $\mathcal{I}$ - оптимальное решение, то

$$f(\mathcal{I}_1) - f(\mathcal{I}) \leq f(\mathcal{I}_1) - \mathcal{E}$$

В частности, если $f(\mathcal{I}_1) = \mathcal{E}$, то $\mathcal{I}_1$ - оптимальное решение. Таким образом, в методе ветвей и границ можно оценить насколько полученное решение близко оптимальному. Именно это свойство является причиной широкого использования метода ветвей и границ для решения экстремальных комбинаторных задач.

Пример 7.4. Применим метод ветвей и границ для решения задачи из примера 7.1. Способ разбиения подмножеств и стратегию движения по дереву ветвлений возьмем те же самые, что и в примере 7.3. Изменим только способ оценки

подмножеств. Для этого опишем сначала процедуру приведения матрицы расстояний. Докажем сначала простое утверждение: добавление любого числа ко всем элементам строки (или столбца) матрицы расстояний приводит к изменению длин всех гамильтоновых контуров на величину этого числа. Это утверждение следует из очевидного факта, что в любой гамильтонов контур обязательно входит один и только один элемент каждой строки (столбца). Следовательно, при таком преобразовании матрицы расстояний оптимальный гамильтонов контур остается оптимальным. Будем добавлять числа к строкам и столбцам матрицы таким образом, чтобы получить так называемую приведенную матрицу. В приведенной матрице все элементы неотрицательны, причем в каждой строке и столбце имеется по крайней мере один нулевой элемент. Для приведенной матрицы любой гамильтонов контур имеет неотрицательную длину и поэтому 0 является нижней границей для множества всех решений. Чтобы получить приведенную матрицу, вычтем из каждой строки минимальный элемент. Затем подобную операцию произведем со столбцами. Сумма вычтенных чисел определит нижнюю границу, для длин гамильтоновых контуров. Приведенная матрица для задачи, рассмотренной в примере 7.I. имеет вид

i \ j	0	I	2	3	4	u_i
0	X	I	0	I	I	I
I	2	X	I	I	0	3
2	I	3	X	0	0	I
3	0	2	4	X	6	I
4	5	0	I	I	X	I
v_j	0	0	0	I	0	

В последнем столбце записаны числа u_i, которые вычитались из строк, а в последней строке v_j, которые вычитались из столбцов. Их сумма $W = \sum_{i=0}^{4} u_i + \sum_{j=0}^{4} v_j = 8$ и является оценкой снизу множества всех решений.

Получим нижнюю границу для подмножества $\{1,?\}$. Для этого объединим вершины 0 и 1 в одну вершину x_0 и примем $l_{x_0 i} = l_{1i}$, $l_{i x_0} = l_{i0}$, $i = 2,3,4$ где (l_{ij}) – приведенная матрица расстояний. Получим следующую матрицу:

i \ j	x_0	2	3	4
x_0	X	1	1	0
2	1	X	0	0
3	0	4	X	6
4	5	1	1	X

Для того, чтобы получить приведенную матрицу, достаточно вычесть 1 из четвертой строки. Учитывая также, что $l_{01} = 1$, получаем оценку подмножества $\{1, ?\}$, равную 10. Аналогично вычисляем $\varphi(2) = 8$, $\varphi(3) = 9$, $\varphi(4) = 10$. Выбираем подмножество $\{2,?\}$, имеющее минимальную оценку. Далее получаем оценки подмножеств $\{2,1,?\}$, $\{2,3,?\}$ и $\{2,4,?\}$ $\varphi(2,1) = 12$, $\varphi(2,3) = 12$, $\varphi(2,4) = 9$. Выбираем подмножество $\{2,4,?\}$. Наконец, оценивая подмножества (2,4,1,) и (2,4,3,), получаем $f(2,4,1,) = 9$, $f(2,4,3,) = 13$. Соответствующая часть дерева решений показана на рис.7.3 (оценки проставлены у вершин). Полученное решение (2,4,1,) является оптимальным, так как оценки остальных подмножеств не меньше 9 (рис.7.3).

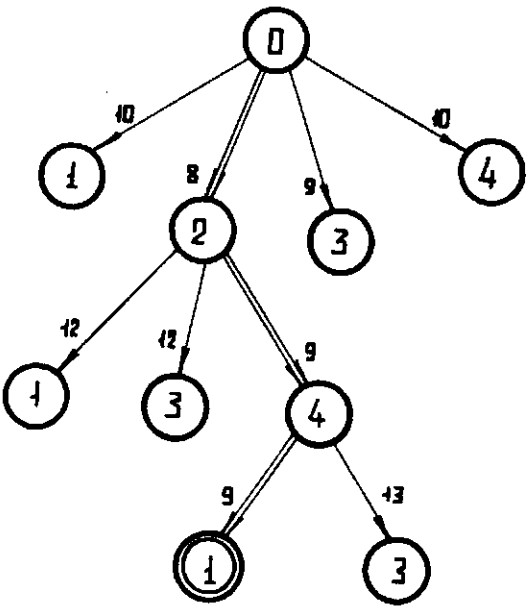


Рис. 7.3

§ 4. Метод динамического программирования

Поставим в соответствие множеству P решений задачи некоторую сеть G, обладающую следующими свойствами:

I. Каждому решению T соответствует единственный путь, соединяющий вход x_0 с выходом z (каждому такому пути могут соответствовать несколько решений).

2. Длина любого пути в сети, соединяющего x_0 с z, равна (минимальному) максимальному из значений $f(T)$ соответствующих решений.

Очевидно, что оптимальному решению задачи соответствует путь в сети, соединяющий вход x_0 с выходом z и имеющий минимальную длину (если решается задача на минимум). Метод динамического программирования применительно к комбинаторным задачам представляет собой метод поиска экстремального пути на сети, обладающей отмеченными выше свойствами. Заметим, что тривиальную сеть можно определить как сеть, вершины которой соответствуют отдельным комбинациям. Однако, число вершин такой сети равно N (не считая x_0 и z), и определение кратчайшего пути эквивалентно полному перебору. С другой стороны, можно определить сеть G с малым числом вершин, однако, определение длин дуг этой сети может потребовать большого объема вычислений.

Пример 7.5. Рассмотрим применение метода динамического программирования к решению задачи коммивояжера с матрицей длин l_{ij} из примера 7.I. Определим подмножества решений $[(i_1, i_2, \dots, i_k), i_{k+1}]$ в каждом решении которого первыми посещаются k городов $(i_1, i_2, \dots, i_k)$ (в любом порядке), а затем город i_{k+1} $(k = 0, 1, 2, \dots, n-1)$.

Поставим в соответствие каждому такому подмножеству вершину сети G. Вершины, соответствующие подмножествам $[(i_1, i_2, \ldots, i_k), i_{k+1}]$ и $[(i_1, i_2, \ldots, i_{k+1}), i_{k+2}]$ соединим дугой длины $l_{i_{k+1} i_{k+2}}$. Получим сеть G, показанную на рис. 7.4. Каждому пути в сети, соединяющему X_0 с Z, соответствует одна и только одна перестановка и что длина пути равна длине соответствующего маршрута коммивояжера. Определим кратчайший путь, применяя алгоритм Форда. Потенциал λ вершины, соответствующий подмножеству $[(i_1, i_2, \ldots, i_k), i_{k+1}]$ определяется по формуле

$$\lambda[(i_1, i_2, \ldots, i_k), i_{k+1}] = \min_{1 \le s \le k} \{\lambda[(i_1, \ldots, i_{s-1}, i_{s+1}, \ldots, i_k), i_s] + l_{i_s i_{k+1}}\} \qquad (7.2)$$

Формула (7.2) выражает принцип оптимальности Беллмана для задачи коммивояжера. Кратчайший путь выделен на рис.7.4, двойными дугами он определяет перестановку (2,4,1,3). Нетрудно определить число вершин графа G для задачи коммивояжере с n городами. Будем называть вершинами k-го уровня вершины, соответствующие подмножеством $[(i_1, i_2, \ldots, i_{k-1}), i_k]$. Имеется n вершин 1-го уровня, $2 \cdot C_n^2$ вершин 2-го уровня, $3 \cdot C_n^3$ – вершин 3-го уровня и т.д. Всего вершин

$$M = \sum_{k=1}^{n} k\, C_n^k + 2$$

Число M быстро растет с ростом n, что накладывает ограничения на размер решаемой задачи.

Можно предложить другой способ, при котором число вершин графа G уменьшается, но для получения длин дуг требуются дополнительные вычисления. Определим два типа

подмножеств. Подмножества первого типа P_i характеризуются номером i города, который посещается первым (таких подмножеств 4), подмножества второго типа Q_j характеризуются номером j города, который посещается последним (таких подмножеств также 4). Длина дуги (P_i, Q_j) равна кратчайшему маршруту из города i в город j проходящему через все остальные города. Например,

$$l_{12} = \min (l_{13} + l_{34} + l_{42} ; \; l_{14} + l_{43} + l_{32}) = 10$$

$$l_{13} = \min (l_{12} + l_{24} + l_{43} ; \; l_{14} + l_{42} + l_{23}) = 7 .$$

Соответствующая сеть показана на рис.7.5. Длины указаны у соответствующих дуг. Кратчайший путь выделен двойными дугами. Экономия памяти машины при втором способе очевидна. Сравним суммарные количества операций сравнения и сложения. При вычислении по схеме рис.7.4 количество сложений равно 56, а количество сравнений 23. Суммарное количество операций равно 79. При вычислении по схеме рис.7.5 количество сложений равно 20, а количество сравнений - II. Кроме того, получение всех длин l_{ij} требует 24 сложения и I2 сравнений. Суммарное число операций 67. Таким образом, второй способ экономнее и по числу операций.

Упражнение 7.2. Определите суммарное число операций сложения и сравнения для задачи коммивояжера с n городами, решаемой по схеме, приведенной на рис.7.4.

Упражнение 7.3. Определите суммарное число операций сложения и сравнения для задачи коммивояжера с n городами, решаемой по схеме рис.7.5, где каждая длина l_{ij} определяется в результате решения задачи коммивояжера с $(n-2)$ городами.

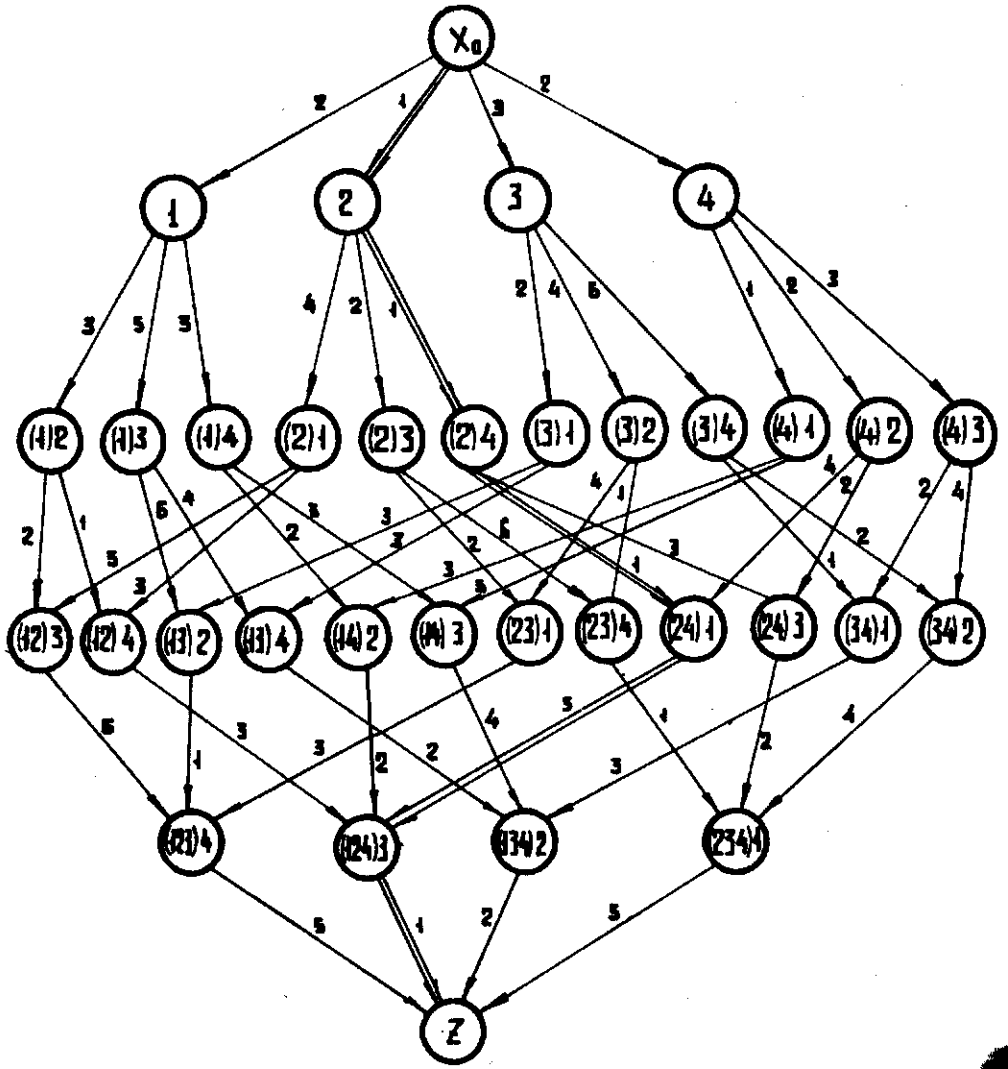

X0
1
2
3
4
(4)2
(1)3
(1)4
(2)1
(2)3
(2)4
(3)1
(3)2
(3)4
(4)1
(4)2
(4)3
(12)3
(12)4
(13)2
(13)4
(14)2
(14)3
(23)1
(23)4
(24)1
(24)3
(34)1
(34)2
(123)4
(124)3
(134)2
(234)1
Z

Рис. 7.4

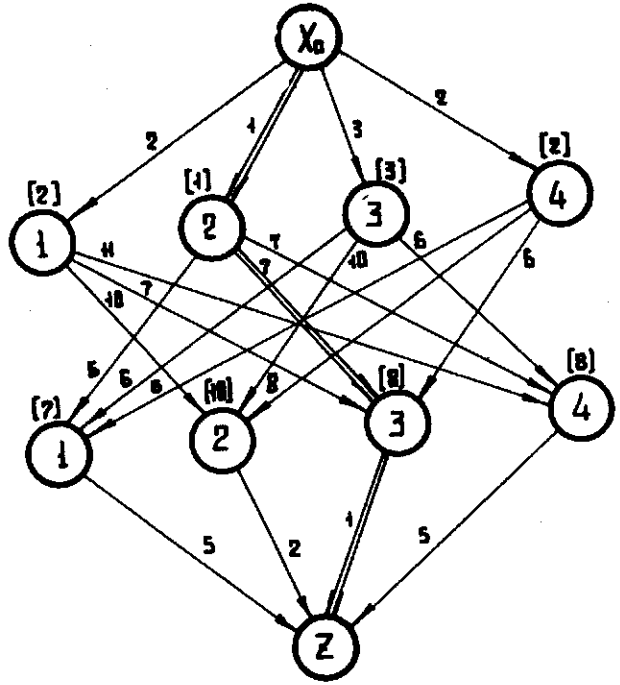

X0
[2]
[1]
[3]
[2]
[7]
[10]
[8]
[8]
1
2
3
4
Z

Рис. 7.5

Предполагается, что каждая задача с $(n-2)$ городами решается по схеме рис.7.4.

§ 5. ВЫВОДЫ

Метод ветвлений (ветвей и границ) и метод динамического программирования являются частными случаями более общей схемы последовательного конструирования, анализа и отбора вариантов . Часто применяются комбинированные методы. Например, метод ветвлений может использоваться для получения начального решения с последующим его улучшением путем локальной оптимизации. Многие методы могут применяться в схеме случайного поиска. Так, в методе локальной оптимизации можно случайным образом выбирать начальное решение. В методе ветвей и границ можно организовать случайный выбор подмножеств таким образом, чтобы вероятность выбора данного подмножества была тем большей, чем меньше его нижняя граница (при решении задачи на минимум). Большой интерес вызывают самонастраивающиеся или адаптивные схемы решения, в которых алгоритм меняется в зависимости от исходных данных (однако, в этом направлении имеется мало результатов и пока трудно судить о перспективности адаптивных схем). Более подробные сведения о методах решения экстремальных комбинаторных задач можно найти в монографии , в книгах, в обзорах.

Глава УШ.

ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ И СООТНОШЕНИЕ ДВОЙСТВЕННОСТИ

В предыдущих главах и особенно в главе IУ мы использовали понятие двойственности в линейном программировании. Для удобства пользования книгой, коротко рассмотрим соотношение двойственности в линейном программировании в настоящей главе.

В данной главе также рассмотрена задача о ранце, являющаяся частным случаем общей задачи линейного программирования, которая используется при описании алгоритмов в гл. V.

§ I. Задача линейного программирования

Задачу линейного программирования можно сформулировать в следующем виде: определить $x_i \geq 0,\ i=1,2,\dots,n$, удовлетворяющие ограничениям

$$\sum_{i=1}^{n} a_{ij} x_i \leq b_j, \quad j=1,2,\dots,m \tag{8.I}$$

и максимизирующие линейную форму

$$C = \sum_{i=1}^{n} c_i x_i \tag{8.2}$$

Поставим в соответствие j -му неравенству (8.I) переменную $y_j\ (j=1,2,\dots,m)$ и сформулируем следующую задачу: определить $y_j \geq 0,\ j=1,2,\dots,m$, удовлетворяющие условиям

$$\sum_{j=1}^{m} a_{ij} y_j \geq c_i, \quad i=1,2,\dots,n \tag{8.3}$$

и минимизирующие линейную форму

$$B = \sum_{j=1}^{m} b_j y_j \tag{8.4}$$

215

Задача (8.3), (8.4) называется двойственной к задаче (8.I), (8.2), а переменные y_j называются двойственными переменными.

Заметим, что для любых $\bar{x}=(x_1, x_2, \dots, x_n)$ и $\bar{y}=(y_1, y_2, \dots, y_m)$ удовлетворяющих соответственно (8.I) и (8.3), имеет место

$$C=\sum_{i=1}^{n} c_i x_i \le \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij} y_j x_i \le \sum_{j=1}^{m} b_j y_j = B. \qquad (8.5)$$

то есть значение целевой функции прямой задачи всегда меньше или равно значению целевой функции двойственной задачи. Поэтому, если нам удалось найти такие $\bar{x}^\circ$ и $\bar{y}^\circ$, что

$$\bar{C}\bar{x}^\circ = \bar{b}\bar{y}^\circ, \qquad (8.6)$$

то $\bar{x}^\circ$ является оптимальным решением прямой задачи, а $\bar{y}^\circ$ - двойственной. Приведем без доказательства основную теорему теории линейного программирования: если одна из задач имеет решение (то есть система 8.I совместна и значение целевой функции в оптимальном решении конечно),то двойственная ей также имеет решение, причем для оптимальных решений прямой и двойственной задач имеет место соотношение (8.6). Эта теорема позволяет получить важные условия,которы называются соотношениями двойственности. Если $\bar{x}^\circ$ и $\bar{y}^\circ$ опти мальные решения,соответственно,прямой и двойственной задач, то из (8.5) и (8.6) следует:

$$\text{если } \sum_{j=1}^{m} a_{ij} y_j > c_i \text{ , то } x_i = 0, \qquad (8.7)$$

$$\text{Если } \sum_{i=1}^{n} a_{ij} x_i < b_j \text{ , то } y_j = 0. \qquad (8.6)$$

Действительно, в противном случае знак равенства в (8.6)

был бы невозможен. Условия (8.7) и называются соотношениями двойственности.

§ 2. Задача о ранце

Много прикладных задач связано с размещением грузов определенного объема, веса или формы в помещении (трюм судна, самолет, поезд, склад) ограниченного объема, грузоподъемности и т.д. Если задана форма размещаемых предметов, а также форма помещения, то задача иногда называется задачей размещения (например, разместить некоторое число объектов с заданными размерами в помещении минимального объема). Если размещаемые предметы являются плоскими фигурами, то задача интерпретируется как задача раскроя (например, из заданного куска материала получить максимальное число выкроек определенной формы). Большинство задач этого типа очень сложны, носят типично комбинаторный характер и не имеют эффективных методов решения. Мы ограничимся рассмотрением одной задачи упаковки (раскроя, размещения в зависимости от интерпретации), которую часто называют задачей о ранце.

2.I. Постановка задачи

Имеется n грузов определенного веса и ценности. Обозначим $C_i > 0$ ценность i-го груза, $P_i > 0$ его вес. Требуется упаковать в "ранец" ограниченной грузоподъемности P такие грузы, чтобы суммарная ценность упакованных грузов была максимальной.

Введем переменные x_i, принимающие два значения 0 или I. Примем $x_i = I$, если i-ый груз упакован и $x_i = 0$, в противном случае, $i = I,2,\ldots,n$. Теперь задачу можно сформулировать как задачу целочисленного линейного

программирования:

максимизировать $$C = \sum_{i=1}^{n} c_i x_i \qquad (8.8)$$

при ограничении

$$\sum_{i=1}^{n} p_i x_i \leq N \qquad (8.9)$$

Заметим, что если допустить дробление груза (то есть x_i может принимать любое значение на отрезке $[0,1]$), то задача решается сразу. Предположим, что грузы пронумерованы таким образом, что

$$\frac{c_1}{p_1} \geq \frac{c_2}{p_2} \geq \cdots \geq \frac{c_n}{p_n}$$

Пусть S такое, что

$$\sum_{i=1}^{S-1} p_i \leq P < \sum_{i=1}^{S} p_i$$

Оптимальное решение задачи

$$x_i = \begin{cases} 1 \text{ если} & i < S \\ \frac{1}{p_S}\left(P - \sum_{i=1}^{S-1} p_i\right), \text{ если} & i = S, \\ 0 \text{ если} & i > S, \end{cases} \qquad (8.10)$$

При этом

$$C' = \sum_{i=1}^{S-1} c_i + c_S \frac{P - \sum_{i=1}^{S-1} p_i}{p_S} \qquad (8.11)$$

является оценкой сверху всего множества решений.

Требование неделимости грузов существенно затрудняет решение задачи.

2.2. Метод динамического программирования

Определим подмножество решений $Q(K, N)$, содержащее все решения, в которых общий вес упакованных грузов

с номерами не большими K равен N_1 $(0 \le N_1 \le N_2,\ 1 \le K \le n-1)$. Каждому $Q(K, N_1) \neq \Phi$ поставим в соответствие вершину сети G. Две вершины, соответствующие подмножествам $Q(K, N_1)$ и $Q(K+1, N_2)$ $(N_1 \le N_2)$ соединим дугой длины

$$l = \begin{cases} 0 & \text{если } N_2 = N_1 \\ C_{K+1} & \text{если } N_2 - N_1 \ge P_{K+1} \end{cases}$$

Вершину x_0 соединим дугой длины C_1 с вершиной, соответствующей множеству $Q(1, P_1)$ и дугой нулевой длины с вершиной, соответствующей множеству $Q(1,0)$. Наконец, вершины, соответствующие множеству $Q(n-1, N_1) \neq \Phi$, соединим с вершиной $\mathcal{Z}$ дугой нулевой длины, если $N - N_1 < P_n$ и дугой длиной C_n, если $N - N_1 \ge P_n$. Путь, соединяющий x_0 с $\mathcal{Z}$ и имеющий максимальную длину, определит оптимальное решение задачи.

Пример 8.1.

$$C = 5x_1 + 7x_2 + 6x_3 + 3x_4 \to \max$$
$$2x_1 + 3x_2 + 5x_3 + 7x_4 \le 9$$

Сеть G приведена на рис.8.1. В вершинах проставлены значения N_1 соответствующего подмножества $Q(K, N_1)$ числа у дуг равны их длинам. Путь максимальной длины выделен двойными дугами. Этот путь определяет оптимальное решение задачи $X_1 = 0$, $X_2 = 1$, $X_3 = 1$, $X_4 = 0$, $C = 13$ (числа в квадратных скобках равны потенциалам вершин). Если n велико, то соответствующая сеть G может содержать большое число вершин. В этом случае целесообразно определять подмножества $Q(K, N_1)$ как подмножества решений, в которых общий вес упакованных грузов с номерами, не большими n_K, равен N_1. Если при этом $n_K > K$, то число вершин

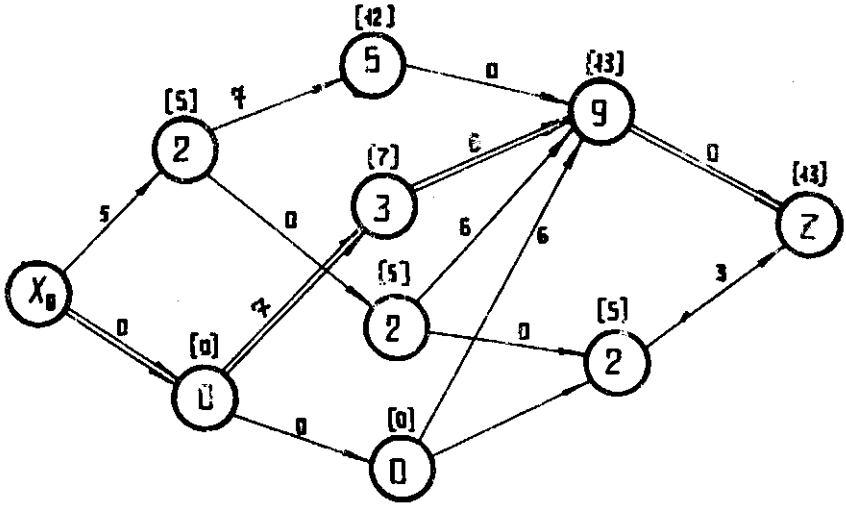


Рис. 8.1

сети G уменьшается. Однако, потребуются дополнительные вычисления для определения длин дуг.

Пример 3.2.

$$C = 17x_1 + 9x_2 + 11x_3 + 13x_4 + 15x_5 + 7x_6 + 2x_7 \to max$$
$$7x_1 + 3x_2 + 4x_3 + 5x_4 + 6x_5 + 2x_6 + x_7 \le 15$$

Будем рассматривать подмножества двух типов $Q(1,N_1)$, $Q(2,N_2)$ причем примем $n_1 = 2$, $n_2 = 5$; для определения длин дуг между вершиной X_0 и вершинами, соответствующими подмножествам $Q(1,N_1)$ необходимо решить задачу о ранце

$$17x_1 + 9x_2 \to max$$
$$7x_1 + 3x_2 \le N_2 \qquad (N_1 = 0,3,7 \text{ и } 10)$$

Для определения длин дуг между вершинами, соответствующими подмножествам $Q(1,N_1)$ и $Q(2,N_2)$, необходимо решить задачу о ранце

$$11x_3 + 13x_4 + 15x_5 \to max$$
$$4x_3 + 5x_4 + 6x_5 \le N_2 - N_1 \quad (N_2 > N_1)$$

Наконец, для определения длин дуг между вершинами, соответствующими подмножествам $Q(2,N_2)$, и вершиной Z необходимо решить задачу о ранце

$$7x_6 + 2x_7 \to max$$
$$2x_6 + x_7 \le 15 - N_2$$

где N_2 может принимать значения 15, 14, 13 и 12.

Определим, например, длину дуги между вершинами, соответствующими подмножествам Q (1,3) и Q (2,14). Для этого решаем задачу

$$11x_3 + 13x_4 + 15x_5 \to max$$
$$4x_3 + 5x_4 + 6x_5 \le 11 .$$

Ее оптимальное решение очевидно $X_3 = 0$, $X_4 = 1$, $X_5 = 1$ и длина дуги равна 28. Поступая аналогично, вычисляем все длины дуг. Полученная сеть показана на рис.8.2. Пути максимальной длины выделены двойными дугами. Они определяют два оптимальных решения:

1) $X_1 = 0$, $X_2 = 1$, $X_3 = 1$, $X_4 = 0$, $X_5 = 1$, $X_6 = 1$, $X_7 = 0$.

2) $X_1 = 0$, $X_2 = 1$, $X_3 = 1$, $X_4 = 1$, $X_5 = 0$, $X_6 = 1$, $X_7 = 1$.

со значением $C = 42$.

В различных приложениях часто встречается задача определения максимальной $\sum_{i=1}^{n} x_i p_i$, не превышающей, однако, P. Такую задачу можно интерпретировать как задачу максимальной загрузки судна заданной грузоподъемности P. По сути дела это частный случай задачи о ранце, когда $p_i = c_i$, $i = 1, 2, \ldots, n$.

Пример 8.3. Требуется выполнить n работ на двух одинаковых машинах. Обозначим p_i - продолжительность i-ой работы. Каждая работа должна выполняться полностью на одной машине, и каждая машина одновременно может выполнять только одну из работ. Требуется распределить работы по машинам, так, чтобы выполнить их за минимальное время. Обозначим $x_i = 1$, если i-я работа выполняется на 1-ой машине и $x_i = 0$, если - на 2-ой. Получаем задачу

$$\sum_{i=1}^{n} x_i p_i \longrightarrow max$$

при условии

$$\sum_{i=1}^{n} x_i p_i \leq P = \left[\frac{\sum_{i=1}^{n} p_i}{2} \right] \qquad (X \text{ - целая часть } X$$

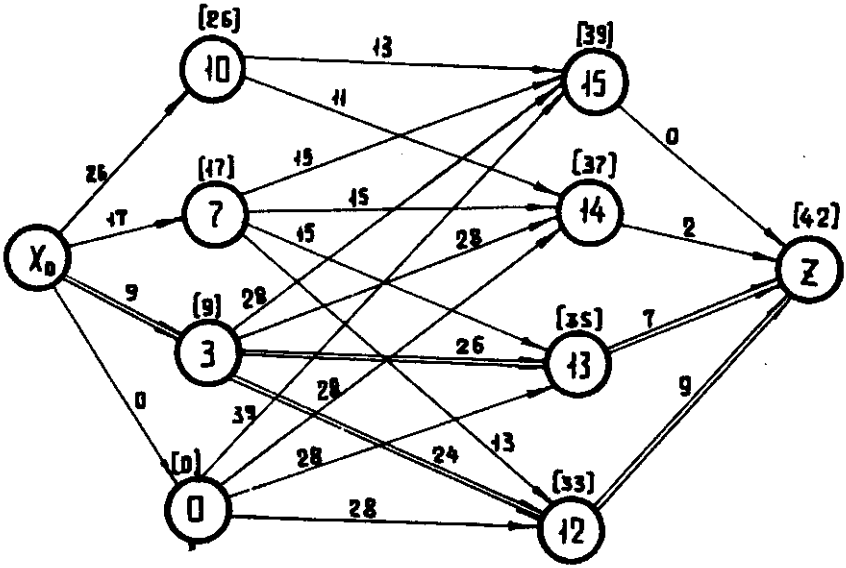

[26]
10
13
[39]
15
11
0
26
[17]
15
[37]
17
7
15
14
2
[42]
X_0
15
28
Z
9
28
[9]
[35]
7
3
26
13
0
28
9
39
13
[0]
28
24
[33]
0
28
12

Рис. 8.2

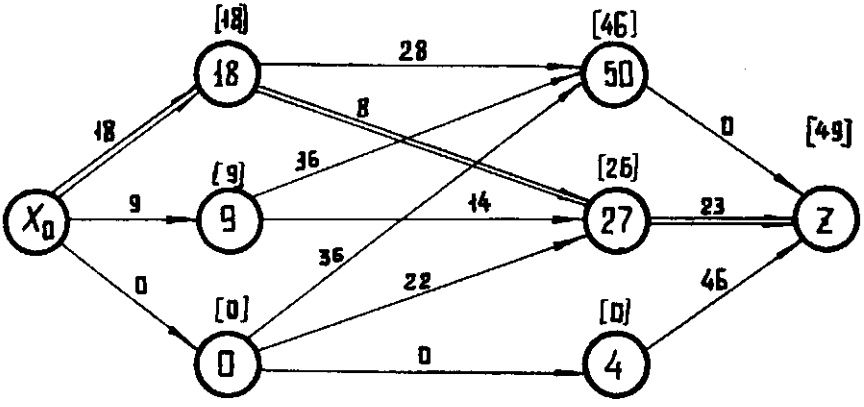


Рис. 8.3

Решим эту задачу для следующих данных

i	I	2	3	4	5	6	7
p_i	9	9	I4	I4	8	23	23

Имеем

$$P = \left[\frac{100}{2}\right] = 50$$

Как и в предыдущем примере примем, что в подмножествах первого типа $Q(1, N_1)N_1$ ограничивает суммарное время работ с номерами I и 2 на первой машине, а в подмножествах второго типа $Q(2, N_2)N_2$ ограничивает суммарное время работ с номерами не большими 5, выполняемых на I-ой машине. Сеть G приведена на рис.8.3. Путь максимальной длины $= 49$ определяет оптимальное решение

$x_1 = I$, $x_2 = I$, $x_3 = 0$, $x_4 = 0$, $x_5 = I$, $x_6 = 0$, $x_7 = I$.

При этом минимальное время выполнения всех работ $P_{min} = 5I$.

2.3. Метод ветвей и границ

Для применения метода ветвей и границ необходимо определить процедуру разбиения множества всех решений на подмножества, а также способ оценки подмножеств. Пусть грузы пронумерованы в порядке убывания их относительной ценности $\frac{C_i}{p_i}$, то есть $\frac{C_1}{p_1} \geqslant \frac{C_2}{p_2} \geqslant \cdots \geqslant \frac{C_n}{p_n}$. Разобьем множество всех решений на два. В любом решении первого подмножества $x_1 = I$, в любом решении второго - $x_1 = 0$. Каждое из подмножеств в свою очередь разбиваем на два, в зависимости от значения $x_2 = I$ или 0, и т.д. Для оценки любого подмножества Q_k, в котором выбраны значения первых k переменных, решаем для оставшихся переменных непрерывную задачу о ранце. Полученное решение вместе со значениями

первых k переменных и определит оценку сверху для решений подмножества Q_k .

Пример 3.4. Решим описанным способом задачу из примера 3.1. Предварительно получим оценку сверху всего множества решений, решая задачу

$$C = 5x_1 + 7x_2 + 6x_3 + 3x_4 \longrightarrow max$$
$$2x_1 + 3x_2 + 5x_3 + 7x_4 \leq 9$$
$$0 \leq x_i \leq 1, \quad i = 1,2,3,4$$

Ее оптимальное решение

$x_1 = x_2 = 1, \quad x_3 = \frac{4}{5}, \quad x_4 = 0, \quad C = 16\frac{4}{5}$

Будем обозначать $\mathcal{Y}(x_1, x_2, \ldots, x_k)$ оценку подмножества, определяемого значениями $x_1, x_2, \ldots, x_k$. Заметим, что $\mathcal{Y}(1)$ также равна $16\frac{4}{5}$. Однако $\mathcal{Y}(1,1) = 12$, так как неравенство $5x_3 + 7x_4 \leq 4$ имеет тривиальное решение $x_3 = x_4 = 0$ (в числах 0 и 1). Найдем оценку $\mathcal{Y}(0)$. Для этого решаем задачу

$$7x_2 + 6x_3 + 3x_4 \longrightarrow max,$$
$$3x_2 + 5x_3 + 7x_4 \leq 9$$

Оптимальное решение $x_2 = 1, \quad x_3 = 1, \quad x_4 = 1/7$ определяет оценку $\mathcal{Y}(0) = 13\,3/7$. Найдем оценку $\mathcal{Y}(1,0)$, решая задачу

$$6x_3 + 3x_4 \longrightarrow max$$
$$5x_3 + 7x_4 \leq 7$$

Оптимальное решение $x_3 = 1, \quad x_4 = 2/7$, определяет оценку $\mathcal{Y}(1,0) = 11\,6/7$. Соответствующая часть дерева ветвлений показана на рис.3.4 (в вершинах указаны оценки). Выбираем подмножество $Q(0)$ и разбиваем его на два $Q(0,1)$ и $Q(0,2)$. Оценка $\mathcal{Y}(0,1)$ по-прежнему равна $15\frac{5}{7}$. Найдем

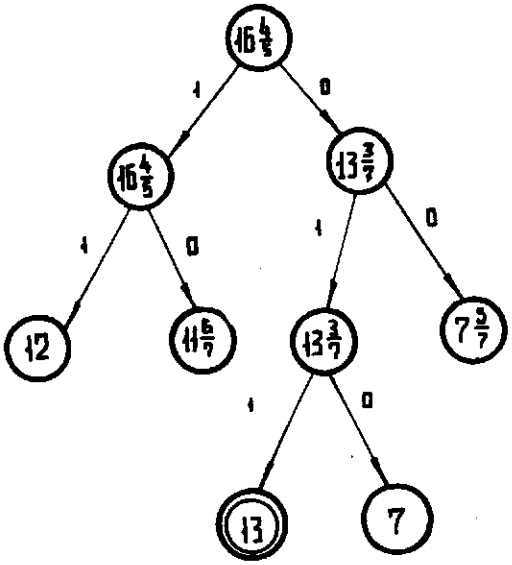


Рис. 8.4

оценку $\varphi(0,0)$, решая задачу

$$6x_3 + 3x_4 \to \max,$$
$$5x_3 + 7x_4 \leq 9$$

Оптимальное решение $x_3 = 1$, $x_4 = 4/7$ определяет оценку $\varphi(0,0) = 7\frac{4}{7}$. Выбираем подмножество $Q(0,1)$ и разбиваем его на два $Q(0,1,1)$ и $Q(0,1,0)$. Оценка подмножества $Q(0,1,1)$ равна 13 и определяет допустимое решение $x_1 = 0$, $x_2 = 1$, $x_3 = 1$, $x_4 = 0$, которое является оптимальным, так как остальные подмножества имеют оценки, меньше 13. Заметим, что метод ветвей границ нецелесообразно применять, если относительные ценности грузов мало отличаются (например, в задаче максимальной загрузки).

2.4. Ограничения несовместимости

Ограничения несовместимости запрещают совместную упаковку некоторых грузов. Эти условия удобно задавать графом H, вершины которого соответствуют грузам и две вершины соединяются ребром, если соответствующие грузы несовместимы. При наличии ограничений типа несовместимостей метод ветвей и границ по-прежнему применим. При получении оценки подмножества $Q(x_1, x_2, \ldots, x_k)$ следует исключать грузы, несовместимые хотя бы с одним из уже упакованных грузов. Пусть в примере 8.4 грузы 2 и 3 несовместимы. В этом случае при получении оценки подмножества $Q(0,1)$ следует исключить груз 3. Оценка $\varphi(0,1)$ при этом будет равна 7, и оптимальное решение задачи

$$x_1 = x_2 = 1, \quad x_3 = x_4 = 0, \quad C = 12$$

Упражнение 8.1. Предложите схему динамического программирования для решения задачи о ранце при ограничениях

несовместимости.

Пусть все грузы имеют одинаковый вес, который примем за I. Если ограничения несовместимости отсутствуют, то задача становится элементарной (следует упаковать p наиболее ценных грузов). При наличии ограничений несовместимости задача становится более сложной. Заметим, что при отсутствии ограничения на общий вес любому допустимому набору грузов соответствует максимальное независимое множество вершин графа H.

Задача 8.I. Определить независимое множество A, имеющее максимальную величину $\sum_{i \in A} c_i$.

Если ценности всех грузов одинаковы, то задача 8.I заключается в определении числа вершинной независимости.

З А К Л Ю Ч Е Н И Е

Читателям, заинтересованным в более глубоком изучении теории графов, рекомендуем монографии [I-3] . Вопросы, связанные с определением экстремальных путей в графах, особенно, теория потоков в сетях, а также транспортные задачи на сетях подробно рассмотрены в книгах [4,5] . С задачами дискретной оптимизации и методами их решения, коротко изложенными в гл. УШ более подробно можно ознакомиться в работах [6-9] . Изучение методов решения задач распределения ресурсов на конечных сетях и связанных с ними задач календарного планирования можно продолжить по книгам [I0-I2] .

В настоящей книге не затронуты вопросы применения теории графов при расчетах электрических цепей, которые подробно рассмотрены в [I4] , а также опущен раздел теория графов и конечные автоматы, достаточно полно освещенный в монографии [I3] .

Л И Т Е Р А Т У Р А

I. Зыков А.А. Теория конечных графов, I том, "Наука", "Сибирское отделение", Новосибирск, I969 г.

2. К.Берж. Теория графов и ее применения. "Иностранная литература", Москва, I962 г.

3. О. Оре. Теория графов. "Наука", Москва, I968.

4. Ю.М.Ермольев, И.М.Мельник, Экстремальные задачи на графах, "Наукова думка", Киев, I968 г.

5. Л.Г.Форд, Д.Р.Фалкерсон, Потоки в сетях. "Мир",Москва, I966г.

6. А.А.Корбут, Ю.Ю.Финкельштейн, Дискретное программирование "Наука", Москва, I969 г.

7. В.Н.Бурков, С.Е.Ловецкий. Методы решения экстремальных комбинаторных задач (обзор), Изв. АН СССР, "Техническая кибернетика", № 4, I968 г.

8. В.Н.Бурков, С.Е.Ловецкий. Методы решения экстремальных задач комбинаторного типа (обзор). "Автоматика и телемеханика", № I, I968 г.

9. В.Н.Бурков, С.Е.Ловецкий. Комбинаторика и развитие техники, "Знание", Москва, I968г.

I0. С.И.Зуховицкий, И.А.Радчик, математические методы сетевого планирования, "Наука", Москва, I965 г.

II. В.В.Шкурба, Т.П.Подгасова, А.Н.Питчук, Л.П.Тур, Задачи календарного планирования и методы их решения, "Наукова думка", Киев, I966 г.

I2. В.Н.Бурков, Б.Д.Ланда, С.Е.Ловецкий, А.И.Тейман, В.Н.Чернышев, Сетевые модели и задачи управления "Советское радио", Москва, I967 г.

I3. А.Н. Мелихов, Ориентированные графы и конечные автоматы, " Наука ", Москва, I97I .

I4. С. Сешу, М.Б. Рид, Линейные графы и электрические цепи, " Высшая школа ", Москва, I97I.

С О Д Е Р Ж А Н И Е

Напечатано по заказу Вычислительного центра

Редактор издательства К.И.Имнадзе

Сдано в набор 24.VI.1974 подписано к печати I2.XII.I973;
формат бумаги 60x90 I/I6; печатаных л. I4,75; уч.-изд.л.8,44.
Цена 72 коп.

Заказ УЭ 0I209 Тираж 500

Издательство "Мецниереба",Тбилиси,380060,ул.Кутузова, I9.
Типография АН ГССР, Тбилиси,380060, ул.Кутузова,I9.

ВЛАДИМИР НИКОЛАЕВИЧ БУРКОВ, ИВАН АЛЕКСЕЕВИЧ ГОРГИДЗЕ,
СЕРГЕЙ ЕВГЕНЬЕВИЧ ЛОВЕЦКИЙ

ПРИКЛАДНЫЕ ЗАДАЧИ ТЕОРИИ ГРАФОВ

под редакцией А.Я.Горгидзе