Лабораторная работа № 1

СТРУКТУРНЫЙ АНАЛИЗ БОЛЬШИХ СИСТЕМ УПРАВЛЕНИЯ. УПОРЯДОЧЕНИЕ ИНФОРМАЦИОННОЙ СТРУКТУРЫ И ОЦЕНКА ВРЕМЕНИ ПОДГОТОВКИ ДОКУМЕНТОВ

Цель работы – приобретение навыков описания и анализа потоков информации в больших системах, включая алгоритмизацию задачи с выделением уровней в структуре и определения их взаимосвязи, а также определения максимальных и минимальных путей прохождения документов.

Задание рассчитано на 1 занятие.

1.1. Цели и задачи структурного анализа

Одной из важнейших характеристик системы является структура, которая у больших систем часто является многоуровневой. Конкретизация структуры дается на стольких уровнях, сколько требуется для полного представления об основных свойствах системы. При этом большие системы управления в первую очередь отличает информационная сложность, т.е. для принятия решений требуется сбор, обработка и анализ больших объемов информации, часто в реальном масштабе времени. В связи с этим актуальны задачи описания и анализа структуры потоков информации в системе.

В дальнейшем будем рассматривать данный круг вопросов на примере автоматизированных систем управления производством (АСУП).

При создании АСУП структурная модель рассматривается на нескольких крупных уровнях, в том числе [1]:

- организации,
- функций управления,
- технических средств.

и вводятся понятия организационной, функциональной и технической структур.

При анализе *организационной структуры* предприятия как объекта управления решаются задачи:

- описание состава организации и построение ее структурной схемы;
- определение функций отдельных подразделений, раскрытие их структурной схемы;
- описание материальных, вещественных и информационных связей;
- построение обобщенной структурной информационной модели предприятия.

При анализе *функциональной структуры*:

- изучаются функции управления в структурных подразделениях существующей системы;
- выбирается состав автоматизируемых функций;
- определяются их взаимосвязи;

■ составляется обобщенная функциональная структура задач управления АСУП.

При анализе *технической структуры*:

■ определяются основные элементы, участвующие в основных информационных процессах регистрации, подготовки, сбора, передачи, хранении, обработке, воспроизведении и выдаче информации;

■ составляется формальная структурная модель системы технических средств с учетом топологии расположения элементов системы и их информационного и энергетического взаимодействия, как между собой, так и с внешней средой.

Независимо от уровня рассмотрения задача структурного анализа состоит в том, чтобы, исходя из заданного описания элементов системы и непосредственных связей между ними, получить заключение о структурных свойствах системы в целом и основных ее подсистем. При решении задач структурного анализа больших систем обычно принимаются три уровня описания связей между элементами:

1. наличие связи;
2. направление связи;
3. вид и направление сигналов, определяющих взаимодействие элементов.

На первом уровне исходят лишь из наличия или отсутствия связей между элементами. Изучаемой системе может соответствовать неориентированный граф, вершинами которого являются элементы системы, а ребрами - связи между элементами. Задачи структурного анализа на этом уровне сводятся к следующим:

■ определение связности (целостности) системы; если система не является связной, то ставят задачу выделения изолированных связных подсистем со списками входящих в них элементов;

■ выделение циклов;

■ определение минимальных и максимальных последовательностей элементов (цепей), разделяющих элементы друг от друга.

На втором уровне, когда задано направление связи, системе соответствует ориентированный граф, направления дуг которого совпадают с направлениями связей. К задачам структурного анализа в этом случае относят:

■ топологическую декомпозицию с выделением сильно связных подсистем;

■ перечисление входных и выходных полюсов и в соответствии с этим выделение узлов приема и выдачи информации;

■ выделение уровней в структуре и определение их взаимосвязи;

■ определение максимальных и минимальных путей и др.

На третьем уровне описания связей между элементами системы учитывается не только направленность связи, но и раскрываются состав и характер сигналов взаимодействия элементов (входные, выходные и управляющие). Указанные выше задачи структурного анализа первого и второго уровней расширяются за счет включения следующих:

■ выделение местных и общих контуров управления;

■ при многорежимном характере функционирования - выделение типичных структурных конфигураций для каждого из режимов;

■ выделение путей передачи входных сигналов и путей передачи управляющих сигналов и др.

Источником информации в АСУП является документ. Взаимодействие элементов в системе приводит к тому, что одни документы формируются на основании других, т.е. происходит движение информации с целью получения определенного функционального результата. Как для системы в целом, так и для любой подсистемы все документы могут быть классифицированы на:

1. *исходные* - поступающие в систему;

2. *внешние* - результаты переработки исходных;

3. *промежуточные* - результаты переработки исходных, которые используются для вычисления внешних документов, но сами из системы не выдаются.

Совокупность исходных и внешних документов составляет информационный базис системы, который не зависит от программ обработки информации, а определяется функциями системы (подсистемы). Между документами, входящими в поток, существуют отношения вхождения и порядка.

Отношение вхождения означает

$$x_i = x_{i_1} x_{i_2} \ldots x_{i_n},$$

т.е. документ x_j образуется непосредственно из документов $x_{i_1} x_{i_2} \ldots x_{i_n}$.

Отношение порядка:

”x_i следует за x_j“

означает, что документ x_i может быть образован только после документа x_j.

Для решения задач анализа и оптимизации документооборота в больших системах эффективно используется аппарат теории графов.

Принцип представления структуры в виде графа достаточно прост:

■ элементами системы при решении одних задач ставят в соответствие ребра графа, а связям - вершины графа.

■ при решении других задач поступают иначе: элементам ставят в соответствие вершины графа, а связям - ребра графа.

Первый граф - *реберный*, второй - *вершинный.*

Тогда, если документам сопоставить вершины графа, а дугам - соответствующие отношения вхождения и порядка, то получится структура, отражающая информационное взаимодействие элементов системы, называемая *информационным графом*.

Введение порядковой функции на информационном графе позволяет выявить его многоуровневую структуру и классифицировать документы по уровням формирования.

Рассмотрим подробнее задачи структурного анализа второго уровня, когда задано направление связи. Такая формализация структур заложена в теории графов, причем, рассматриваемой задаче соответствует структура в виде ориентированного графа.

1.2. Формализация описания структуры на основе теории графов

Приведем основные определения и понятия теории графов.

Пусть определено некоторое множество элементов *V*. Граф

$$G = G(V)$$

считается определенным, если задано некоторое семейство сочетаний элементов или пар вида

$$E = (a,b),$$

где

$$a, b \in V$$

указывает, какие элементы считаются связанными.

Пара

$$E = (a,b)$$

называется *ребром*, а элементы *a*, *b* - *концевыми точками ребра* или *вершинами.*

Если порядок расположения концов важен, то *E* называют ориентированным ребром - *дугой*; при этом *a* называют начальной вершиной, а *b* - конечной вершиной. Говорят также, что ребро *E* *инцидентно* вершинам *a*, *b*, а вершины *a*, *b* инцидентны ребру *E* (т.е. есть связь).

Способы формализованного задания графа

А. Графическое представление

Наиболее наглядно, но не может быть использовано для решения задач структурного анализа на ЭВМ.

Б. Матричное представление.

Существуют различные формы матричного представления графа

$$G = G(V).$$

Матрица смежности вершин для ориентированного графа имеет вид

$$A = \{a_{ij}\},\ i = 1,2,\dots,n;\ j = 1,2,\dots,n,$$

где элементы a_{ij} определяются следующим образом:

$$a_{ij} = \begin{cases} 1, \text{если из вершины } i \text{ можно перейти в вершину } j; \\ 0, \text{в противном случае.} \end{cases}$$

Вид матрицы смежности ориентированных графов будет существенным образом зависеть от выбранного алгоритма нумерации вершин. Кроме того, выбрав определенный принцип нумерации вершин на графе для некоторых типов графов (графов без контуров), можно свести матрицу *A* к треугольному виду, где

$$a_{ij} = 0, \text{ при } j > i,$$

что следует учитывать при задании исходной информации о структуре системы для ЭВМ.

Матрица инцидентности

$$B = \{b_{ij}\},\ i = 1,2,\dots,n;\ j = 1,2,\dots,m,$$

где n - число вершин, m - число ребер, определяется для ориентированного графа следующим образом:

$$b_{ij} = \begin{cases} 1, \text{ если } i - \text{я вершина является началом } j - \text{го ребра;} \\ -1, \text{ если } i - \text{я вершина является концом } j - \text{го ребра;} \\ 0, \text{если } i - \text{я вершина не инцендентна } j - \text{му ребру.} \end{cases}$$

Для многих практических задач, связанных с анализом структурных свойств больших систем, большинство элементов в матрицах *A* и *B* оказываются нулевыми, что приводит к неэффективному использованию памяти ЭВМ. В этом случае целесообразно использовать множественные способы формализации структуры.

B. Множественное представление.

В этом случае для ориентированного графа *G(V)* задается множество вершин *V* и соответствие *G*, которое показывает, как связаны между собой вершины.

Для каждой вершины i соответствие G определяет множество вершин $G(i)$, в которые можно непосредственно попасть из вершины i. В ряде случаев $G(i)$ называется *множеством правых инциденций*.

Множество $G^{-1}(i)$ определяет все вершины, из которых можно непосредственно попасть в вершину i, и поэтому называется *обратным соответствием (отображением)*. По аналогии с $G(i)$ множество $G^{-1}(i)$ называется множеством *левых инциденций*.

Таким образом, ориентированный граф задается перечислением (списком):

- либо множеств вида $G(i)$,
- либо множеств вида $G^{-1}(i)$ для всех вершин графа.

Такой способ оказывается значительно более компактным и эффективным при задании исходной информации о структуре для решения задач анализа и синтеза, особенно для иерархических систем.

Порядковая функция на графе. Понятие уровня

Целью введения порядковой функции на графе без контуров является разбиение множества вершин графа на непересекающиеся подмножества, упорядоченные так, что если вершина входит в подмножество с номером i, то следующая за ней вершина - в подмножество с номером больше i. Полученные непересекающиеся подмножества называются *уровнями*.

Алгоритм упорядочения сводится к следующему [2]. В подмножество нулевого уровня N_0 включаются все вершины i, у которых

$$G^{-1}(i)=\varnothing,$$

т.е. вершины, в которые не входит ни одна дуга.

Проводится последовательная нумерация вершин: $i=1,2,..,l$.

В подмножество первого уровня N_1 включаются все вершины i, у которых

$$G^{-1}(i)\subset N_0,$$

т.е. вершины, в которые входят дуги только из вершин, принадлежащих N_0.

Проводится последовательная нумерация вершин: $i=l+1,l+2,\ldots, l+r$.

В подмножество второго уровня N_2 включаются все вершины i, у которых

$$G^{-1}(i)\subset(N_0 \cup N_1),$$

т.е. вершины, в которые входят дуги только из вершин, принадлежащих N_0 и N_1.

Проводится последовательная нумерация вершин: $i=l+r+1,l+r+2,\ldots, l+r+p$.

В подмножество третьего уровня N_3 включаются все вершины i, у которых

$$G^{-1}(i)\subset(N_0 \cup N_1 \cup N_2),$$

после чего также проводится нумерация вершин.

Данный процесс повторяется до тех пор, пока не будут пронумерованы все вершины графа.

Изложенная процедура нумерации приводит к тому, что в матрице смежности вершин графа

$$a_{ij}=0 \ \textit{при} \ i>j.$$

Пример. Исходный неупорядоченный граф приведен на рис.1.1.

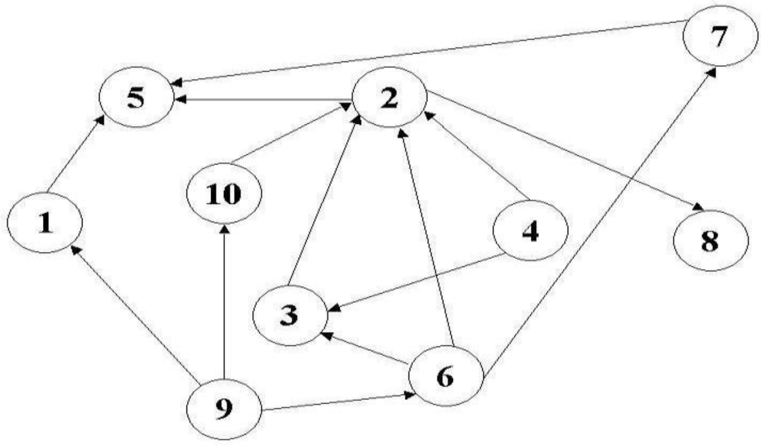

Рис. 1.1. Исходный неупорядоченный граф системы.

В результате применения рассмотренного выше алгоритма получаем упорядоченный граф (рис.1.2)., которому соответствуют множества:

N_0 = {1(4), 2(9)}, N_1 = {3(10), 4(6), 5(1)}, N_2 = {6(3), 7(7)},

N_3 = {8(2)}, N_4 = {9(8), 10(5)},

где в скобках указан исходный номер вершины.

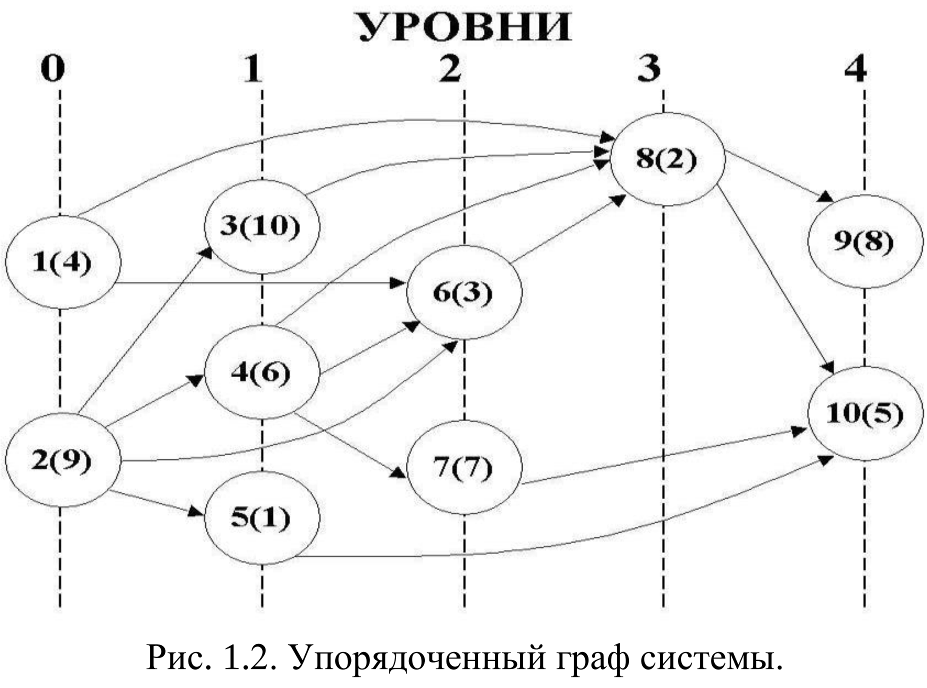

Рис. 1.2. Упорядоченный граф системы.

Числовая функция на графе

Числовую функцию на графе задают обычно либо на вершинах, либо на дугах (ребрах) графа.

Числовая функция на вершинах графа считается заданной, если каждой i -ой вершине a_i графа $G(V)$, $a_i \in V$, ставится в соответствие некоторое число

$$I_i = I(a_i) \tag{1.1}$$

из некоторого множества *L*.

Числовая функция на дугах (ребрах) для ориентированного графа считается заданной, если каждой дуге (a_i,a_j) или ребру ставится в соответствие число

$$q = q\,(a_i,a_j) \tag{1.2}$$

из некоторого множества *Q*.

В этом случае числовая функция может быть задана матрицей смежности, в которой вместо единицы, определяющей только факт наличия связи между соответствующими вершинами, ставится число $q = q\,(a_i,a_j)$. Такую матрицу часто называют *матрицей цены*.

В некоторых случаях числовая функция на графе задается комбинированным способом, как на вершинах, так и на дугах.

Значение функции на пути S через вершины $a_1, a_2, ..., a_i, ..., (a_i \in S)$ при задании числовой функции на вершинах графа определяется либо в соответствии с *аддитивной формой*

$$I_s = \sum_{a \in S} I(a_i) \tag{1.3}$$

либо в соответствии с *мультипликативной формой*

$$I_s = \prod_{a \in S} I(a_i) \tag{1.4}$$

Аналогичным образом определяется значение функции на пути через вершины $a_1, a_2, ..., a_i, ...$ при задании числовой функции на дугах (ребрах) графа:

$$q_s = \sum_{(a_i a_j) \in S} q(a_i a_j)\,, \tag{1.5}$$

$$q_s = \prod_{(a_i a_j) \in S} q(a_i a_j)\,. \tag{1.6}$$

В соответствии с данными определениями можно поставить задачу нахождения путей через множество вершин (дуг), обладающих определенным свойством, с максимальным (минимальным) значением числовой функции. Такие пути называются *максимальными* (*минимальными*). Определение максимальных (минимальных) путей на графе чаще всего формализуется в виде задачи динамического программирования. Так, определение максимального пути на графе без контуров в соответствии с (5) реализуется на основе функционального уравнения

$$q_s^{\max}(a_i a_j) = \max_{a_i \in G^{-1}(a_j)} [q_s^{\max}(a_1 a_j) + q(a_i a_j)], \qquad (1.7)$$

где:

$q_s^{\max}(a_1 a_j)$ - максимальное значение функции на путях s из некоторой начальной вершины a_1 в вершину a_j (аналогично $q_s^{\max}(a_i a_j)$)**;**

$G^{-1}(a_j)$ - множество левых инциденций для вершины a_j;

$q = q\ (a_i,a_j)$ - значение функции на дуге (a_i,a_j).

Определение максимальных и минимальных путей имеет многочисленные приложения при проектировании больших систем:

- задачи сетевого и календарного планирования для определения критического пути;
- транспортные задачи;
- задачи контроля и диагностики и др.

1.3. Задание и порядок выполнения работы

1. Граф системы, заданный приведенной ниже матрицей смежности и значениями числовой функции на дугах,

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1									2								
2																	2
3					4		5				3						
4									2								
5				3		4											
6												3					
7	2			2													
8														2			
9																3	
10															1		
11	3	4															
12																2	
13															2		
14															2		
15																	
16										3			3				
17								2									

необходимо дополнить дугами в соответствии с номером бригады по журналу:

Вар-т	1	2	3	4	5	6	7	8	9	10	11	12
Дуга	(12,8)	(5,2)	(11,4)	(7,2)	(3,4)	(5,12)	(6,9)	(1,17)	(11,17)	(4,12)	(2,9)	(1,12)
Цена	4	4	5	3	6	5	4	3	5	4	3	4
Дуга	(9,10)	(4,16)	(6,17)	(9,8)	(7,9)	(4,17)	(1,8)	(8,10)	(12,13)	(2,12)	(3,1)	(17,16)
Цена	4	5	5	2	5	6	5	1	5	4	8	5

2. Разработать блок-схему алгоритма решения задач:
- упорядочения вершин;
- определения минимального пути.

Алгоритм должен обеспечивать возможность изменения исходных данных (элементов матрицы смежности, значений числовых функций на графе).

3. Реализовать программу решения указанных задач.

1.4. Контрольные вопросы

1. Приведите примеры больших систем управления из области производства, транспорта, экономики и т.д., для которых возможна постановка подобного рода задач.
2. Какие уровни описания связей между элементами используются при решении задач структурного анализа больших систем? Какое описание используется для каждого уровня? В чем состоят задачи структурного анализа на каждом из уровней?
3. Какие способы формализованного задания графа используются, в чем их достоинства и недостатки.
4. Какие способы задания числовой функции на графе существуют?

1.5. Указания по оформлению отчета

Отчет по лабораторной работе должен содержать:

1. Матрицу смежности графа с индивидуально дополненными дугами.
2. Исходное графическое представление системы.
3. Графическое представление системы, упорядоченной по уровням, с отображением минимального пути.
4. Блок-схему решения задач упорядочения вершин и определения минимального пути.
5. Листинг программы.
6. Распечатку результатов решения.

Лабораторная работа №2

СИНТЕЗ СТРУКТУРЫ БОЛЬШИХ СИСТЕМ УПРАВЛЕНИЯ. РАСПРЕДЕЛЕНИЕ ЗАДАЧ ПО УЗЛАМ УПРАВЛЕНИЯ

Цель работы – приобретение навыков оптимизации структуры больших систем управления на примере решения задачи оптимального распределения решаемых в большой системе задач по узлам и уровням системы в соответствии с выбранным критерием эффективности.

Работа рассчитана на 1 занятие.

2.1. Общая постановка задачи. Метод решения

В процессе определения структуры БСУ часто приходится решать задачу распределения выбранных задач по узлам и уровням системы. При этом выбранная структура системы считается оптимальной, если достигается оптимум выбранного показателя эффективности, отражающего основные свойства системы с точки зрения выполнения поставленных задач.

Типовая задача распределения выбранных задач по узлам управления системы имеет вид [1,3]:

- известна предполагаемая топология размещения возможных узлов (центров) управления,
- функции управления перечислены в виде последовательности задач, которые необходимо распределить между узлами управления.

При построении структуры системы управления необходимо распределить функции (задачи) $(i=1,..., I)$ между узлами управления $(j=1,..., J)$ в соответствии с выбранным показателем эффективности $f(\bar{x})$.

Данная задача является моделью дискретного программирования и носит комбинаторный характер. В свою очередь, большая группа комбинаторных методов базируется на достаточно общей схеме методов, которые объединены под одним названием - **методы «ветвей и границ».**

2.2. Алгоритмическая схема метода «ветвей и границ»

Для определенности будем считать, что ищется минимум целевой функции.

В основу метода «ветвей и границ» положены следующие этапы [4]:

1. Определение нижней оценки целевой функции на множестве допустимых решений X_0.

Под нижней границей оценки целевой функции $f_0(x)$ понимается такое значение $V(X_0)$**,** при котором для всех допустимых решений

$$x^T = (x_1, x_2, ..., x_n)$$

имеет место соотношение:

$$f(x^{\min}) \geq V(X_0), x \in X_0 \ . \tag{2.1}$$

Оценка $V(X_0)$ должна также удовлетворять следующему свойству: оценка по множеству допустимых решений должна быть не хуже оценки по любому входящему в него подмножеству, т.е.

$$V(X_0) \leq V(X_1), X_1 \subset X_0. \tag{2.2}$$

2. Ветвление, т.е. построение дерева решений или разбиение всего множества решений на подмножества (см. рис.2.1) в соответствии с выбранным признаком, причем

$$X_0 = \cup X_{i_1}, X_{i_1} = \cup X_{i_1 i_2} \ldots$$

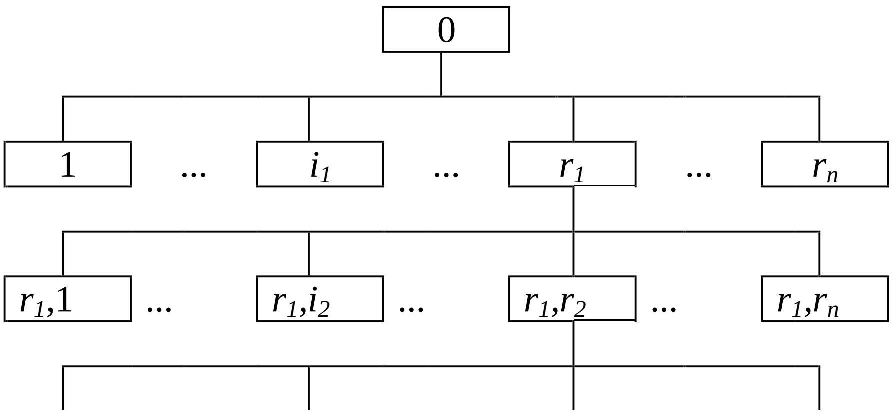

Рис. 2.1. Построение дерева решений.

На первом шаге при определенном выше правиле определения оценок ищется

$$\min_{i_1} V(X_{i_1}) = V(X_{r_1}) \ .$$

На втором шаге производится ветвление подмножества X_{r_1} и вычисление соответствующих оценок $V(X_{r_1, i_2})$ на подмножествах X_{r_1, i_2}. Далее ищется

$$\min_{i_2} V(X_{r_1, i_2}) = V(X_{r_1, r_2})$$

Процесс ветвления продолжается до тех пор, пока полностью не будет определено решение $x^{(0)}$, «подозреваемое» на оптимальность. В этом случае выделенное подмножество решений включает в себя только один вариант $x^{(0)}$.

3. Определение оптимальности решения $x^{(0)}$.

Для тех случаев, когда оценка $V(X_0)$ построена таким образом, что позволяет определить точную нижнюю границу для значений целевой функции $f(x)$ на множестве допустимых решений, то найденное решение $x^{(0)}$ является строго оптимальным решением $x^{\min}$.

Если построенная оценка $V(X_0)$ является только грубой оценкой нижней границы для значений целевой функции $f(x)$, т.е. неравенство (2.1) является строгим, то решение $x^{(0)}$ в общем случае является приближенно оптимальным решением (субоптимальным), хотя может быть и строго оптимальным.

В задачах дискретного программирования, носящих комбинаторный характер, чаще используются грубые оценки, ибо они проще строятся и вычисляются, а поэтому получение в случае применения грубой оценки решения $x^{(0)}$ еще не гарантирует его оптимальности.

В случае применения точных оценок $V(X_0)$ получение оптимального решения $x^{(0)}$ заканчивает работу алгоритма метода «ветвей и границ». При использовании грубых оценок возможно возникновение такой ситуации, когда полученное значение $V(x^{(0)})$ оказывается хуже, чем значения оценок на некоторых неразветвленных подмножествах построенного дерева решений. В этом случае обычно организуется дополнительное ветвление, позволяющее улучшить значение $x^{(0)}$, алгоритм которого сводится к следующему.

Пусть уровень последнего ветвления, на котором определено решение

$$x^{(0)T} = (x_{r_1}, x_{r_2}, \dots, x_{r_n}),$$

есть уровень с номером $(n - 1)$. Обозначим значение $V(x^{(0)})$ через R и назовем рекордом.

Поднимаемся на уровень $(n - 2)$ и ищем

$$\min_{i_{n-2}} V(X_{r_1 r_2 \dots r_{n-3} i_{n-2}}), i_{n-2} \neq r_{n-2}.$$

Если оказывается, что

$$R \leq \min_{i_{n-2}} V(X_{r_1, r_2, \dots, i_{n-2}})$$

то это означает, что на данном уровне нет подмножеств, которые могли бы дать при дополнительном ветвлении решение лучшее, чем найденное $x^{(0)}$, в силу ранее принятого свойства оценки (2.2). Поэтому переходим на уровень ($n-3$).

Если получили, что

$$R > \min_{i_{n-2}} V(X_{r_1, r_2, \dots, i_{n-2}}), \; i_{n-2} \neq r_{n-2},$$

то определяем s_{n-2}, на котором достигается

$$V(X_{r_1, r_2, \dots, r_{n-3}, s_{n-2}}) = \min_{i_{n-2}} V(X_{r_1, r_2, \dots, r_{n-3}, i_{n-2}})$$

и производим дополнительное ветвление подмножества $X_{r_1, r_2, \dots, r_{n-3}, s_{n-2}}$ до тех пор, пока не происходит ухудшение получаемых оценок по сравнению с найденным значением рекорда R. В противном случае переходим на уровень ($n-3$). Если в процессе дополнительного ветвления определен новый рекорд, который оказывается лучше, чем

$$R \; (R_1 < R),$$

то рассмотренная выше процедура повторяется для нового рекорда R_1. Если

$$R_1 \geq R,$$

то переходим на уровень с номером ($n-3$), сохраняя старое значение рекорда R, и рассмотренная выше процедура улучшения значения $x^{(0)}$ повторяется, начиная с уровня ($n-3$) и далее.

Данный алгоритм улучшения значения $x^{(0)}$ при использовании грубых оценок заканчивает работу, если найденное значение рекорда оказывается не хуже (меньше либо равно), чем значения оценок на всех неразветвленных подмножествах построенного дерева решений.

В общем случае алгоритм дополнительного ветвления, реализуемый в возвратно-поступательном режиме, может привести к полному перебору вариантов и, чем грубее оценка, тем больше вероятность приблизиться к полному перебору при поиске оптимального решения. Однако можно ограничиться поиском приближенного оптимального решения, если ввести оценку точности приближенного оптимального решения ε. В этом случае ветвление любого подмножества X не проводится, если

$$|R - V(X)| \leq \varepsilon.$$

Рассмотренная схема применения метода «ветвей и границ» является достаточно общей. Однако при реализации решения конкретных задач дискретного программирования требуется, исходя из максимального учета специфики решаемой задачи, конкретизировать способы вычисления оценок и методы ветвления.

2.3. Постановки задачи

Задано:

- множество задач, реализуемых в системе управления $(i = 1,..., I)$;
- множество узлов системы управления $(j = 1,..., J)$.

Необходимо так распределить задачи по узлам системы, чтобы достигнуть максимального значения эффективности, не выходя из области допустимых ограничений.

Пусть:

- c_{ij} - затраты на реализацию i-ой задачи в j-ом узле;
- t_{ij} - время решения задачи в j-ом узле.

Введем дополнительную переменную x_{ij}:

$$x_{ij} = \begin{cases} 1, \textit{если } i - \textit{ая задача выполняется в } j - \textit{ом узле}, \\ 0, \textit{в противном случае}. \end{cases}$$

Оптимизация распределения задач по узлам может производиться по одной из следующих целевых функций:

- минимизации затрат

$$\min_{x_{ij}} \sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij} x_{ij} \tag{2.3}$$

- минимизации общего времени решения задачи

$$\min_{x_{ij}} \sum_{i=1}^{I} \sum_{j=1}^{J} t_{ij} x_{ij} \tag{2.4}$$

- минимизации максимального времени решения задачи в узле

$$\min_{x_{ij}} [\max_{j} (\sum_{i=1}^{I} t_{ij} x_{ij})] \tag{2.5}$$

При этом могут учитываться следующие ограничения:

- связи между задачами (обычно задаются графом *G(I)*, где *I* - множество задач);
- связи между узлами, в которых находятся элементы системы (обычно задаются графом *G(J)*, где *J* - множество узлов);
- ограничение на общее время решения всех задач, если минимизируются затраты

$$\sum_{i=1}^{I} \sum_{j=1}^{J} t_{ij} x_{ij} \le T_з$$

- ограничение на общие затраты по реализации задач в системе управления, если минимизируется время решения всех задач,

$$\sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij} x_{ij} \le C_з$$

- ограничение на загрузку каждого узла,

$$\sum_{i=1}^{I} \lambda_i t_{ij} x_{ij} \le \rho_{зj}$$

где: λ_i - интенсивность поступления i-ой задачи на решение,

$\rho_{зj}$ - допустимая загрузка j-го узла;

- если каждая задача решается только в одном узле системы, то

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

В зависимости от того, как выбрана целевая функция и какие ограничения учитываются, возникает ряд частных постановок задачи оптимального распределения задачи по узлам.

Минимизация общих затрат (общего времени) при ограничениях на загрузку каждого из узлов и при условии, что каждая задача решается только в одном узле системы, т.е.

$$\min_{x_{ij}} \sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij} x_{ij} \; ,$$

либо

$$\min_{x_{ij}} \sum_{i=1}^{I} \sum_{j=1}^{J} t_{ij} x_{ij}$$

при

$$\sum_{i=1}^{I} \lambda_i t_{ij} x_{ij} \le \rho_{зj} \; ,$$

$$\sum_{j=1}^{J} x_{ij} = 1, i = 1, 2,..., I.$$

Минимизация общих затрат (общего времени) при ограничениях на общее время (общие затраты) и при условии, что каждая задача решается только в одном узле системы, т.е.

$$\begin{cases} \min\limits_{x_{ij}} \sum\limits_{i=1}^{I} \sum\limits_{j=1}^{J} c_{ij} x_{ij} \\ \sum\limits_{i=1}^{I} \sum\limits_{j=1}^{J} t_{ij} x_{ij} \le T_з, \\ \sum\limits_{j=1}^{J} x_{ij} = 1, i = 1, 2, ..., I, \\ x_{ij} = (0,1). \end{cases} \tag{2.6}$$

В дальнейшем будем рассматривать только задачу (2.6). Дерево ветвления строится следующим образом:

- подмножество первого уровня разбиения формируем, фиксируя соответствие первой задачи различным узлам $(X_1, ..., X_{i_1}, ..., X_J)$. Множество X_{j_1} включает в себя все варианты, где первая задача решается в узле j_1, а распределение остальных задач по узлам произвольное.
- аналогично, множества второго уровня формируем, фиксируя соответствие второй задачи различным узлам. Множество $X_{j_1 j_2}$ включает в себя все варианты решений, где первая задача решается в узле j_1, вторая задача - в узле j_2, а остальные задачи имеют произвольное распределение по узлам и т.д.

Для каждого из подмножеств (вершин дерева) необходимо построить оценки целевой функции и ограничения. Общее выражение оценки целевой функции для множества вариантов $X_{j_1 j_2 ... j_i ... j_l}$ в данной задаче может быть построено следующим образом:

$$V_c(X_{j_1 j_2 ... j_i ... j_l}) = \sum_{i \le l} c_{ij_i} + \sum_{i > l} \min_j c_{ij}, \tag{2.7}$$

$$V_t(X_{j_1 j_2 ... j_i ... j_l}) = \sum_{i \le l} t_{ij_i} + \sum_{i > l} \min_j t_{ij} \tag{2.8}$$

где j_1, j_2, ..., j_i ,..., j_l - множество узлов системы, закрепленных за соответствующими задачами $1, 2, ..., i, ..., l$.

Оценка для ограничения (2.8) необходима в данном случае для исключения из процесса ветвления множества заведомо не подходящих вариантов с учетом принятого ограничения на общее время решения задач.

Для ускорения процесса поиска используются всякие дополнительные приемы, учитывающие специфику задачи.

Пример. Пусть заданы [1]:

- матрица стоимости затрат

$$C = \begin{vmatrix} 3 & 7 & 2 & 2 \\ 4 & 8 & 1 & 3 \\ 5 & 9 & 6 & 2 \\ 6 & 10 & 7 & 1 \\ 7 & 5 & 3 & 1 \end{vmatrix}$$

- матрица временных затрат

$$T = \begin{vmatrix} 1.5 & 3 & 2 & 9 \\ 2 & 6 & 5 & 10 \\ 3 & 7 & 6 & 11 \\ 4 & 8 & 7 & 12 \\ 4 & 9 & 8 & 5 \end{vmatrix}$$

Значение $T_з = 20$.

Так как каждая задача может решаться только в одном узле системы, то

$$\min_{x_{ij}} \sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij} x_{ij} = \sum_{i=1}^{I} \min_{x_{ij}} \sum_{j=1}^{J} c_{ij} x_{ij}$$

Используя это условие и принимая во внимание ограничение по времени, можно исключить из матриц C и T элементы, которые не влияют на выбор оптимального решения. Показано, что такими элементами в каждой строке являются элементы, удовлетворяющие одновременно двум условиям:

$$c_{ij} > \min_{j} c_{is};$$

$$t_{ij} > t_{is}.$$

После такого преобразования матрицы $C^{(0)}$ и $T^{(0)}$ принимают вид:

$$C^{(0)} = \begin{vmatrix} 3 & --- & 2 & --- \\ 4 & --- & 1 & --- \\ 5 & 9 & 6 & 2 \\ 6 & 10 & 7 & 1 \\ 7 & --- & --- & 1 \end{vmatrix}$$

$$T^{(0)} = \begin{vmatrix} 1.5 & --- & 2 & --- \\ 2 & --- & 5 & --- \\ 3 & 7 & 6 & 11 \\ 4 & 8 & 7 & 12 \\ 4 & --- & --- & 5 \end{vmatrix}$$

Принимая во внимание ограничение по времени

$$T_з = 20,$$

можно из матриц $C^{(0)}$ и $T^{(0)}$ также вычеркнуть элементы, при которых всегда будет нарушено ограничение во времени. Для выделения таких элементов используется ограничение:

$$\sum_{i=1}^{r-1} \min t_{ij} + t_{rj} + \sum_{i=r+1}^{I} \min_j t_{ij} \le T_з \,. \qquad (2.9)$$

Изменяя индекс строки r **(**$r = 1,2,...,i,..., I$**)** и просматривая все элементы этой строки в соответствии с условием (2.9), можно еще раз преобразовать матрицы $C^{(0)}$ и $T^{(0)}$, вычеркнув такие элементы, если они имеются. Проведем такое преобразование матриц $C^{(0)}$ и $T^{(0)}$. В результате получим:

$$C^{(1)} = \begin{vmatrix} 3 & --- & 2 & --- \\ 4 & --- & 1 & --- \\ 5 & 9 & 6 & --- \\ 6 & 10 & 7 & --- \\ 7 & --- & --- & 1 \end{vmatrix}$$

$$T^{(1)} = \begin{vmatrix} 1.5 & --- & 2 & --- \\ 2 & --- & 5 & --- \\ 3 & 7 & 6 & --- \\ 4 & 8 & 7 & --- \\ 4 & --- & --- & 5 \end{vmatrix}$$

Из матриц $C^{(1)}$ и $T^{(1)}$ видно, что исходные матрицы значительно упростились после проведения дополнительных преобразований.

Перейдем теперь непосредственно к решению задачи на основе алгоритма ветвей и границ. Дерево решения с учетом построенных оценок (2.7) и (2.8) представлено на рис. 2.2. В узлах дерева слева представлена оценка целевой функции, справа - оценка ограничения. Зачеркнуты узлы, не удовлетворяющие ограничению.

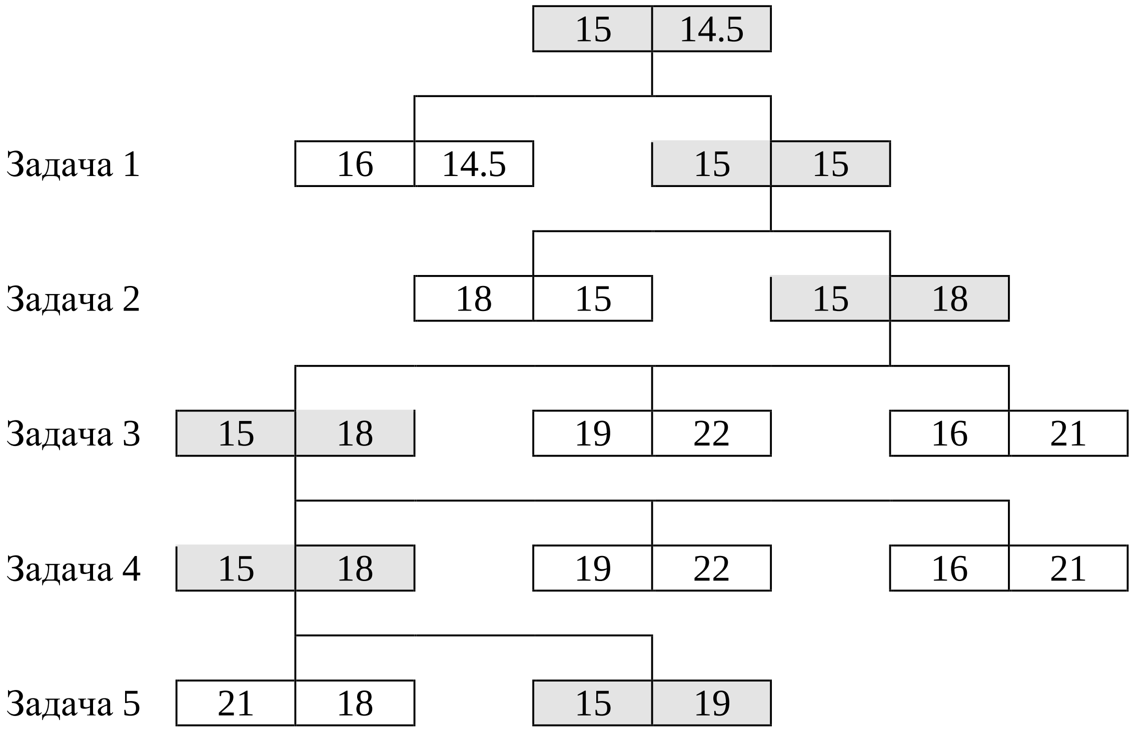

Рис. 2.2. Дерево решения задачи.

В результате решения получили следующее оптимальное распределение задач по узлам (задача - узел): 1 - 3; 2 - 3; 3 - 1; 4 - 1; 5 - 4.

Заштрихованными областями представлен путь поиска оптимального решения на дереве вариантов.

2.4. Задание

5.1. Разработать подробную блок-схему алгоритма.

5.2. Преобразовать исходные матрицы стоимости затрат и временных затрат.

5.3. Составить программу и решить задачу.

5.4. Составить дерево решения.

Варианты для бригад определяются следующим образом: в матрицах стоимости затрат *C* и временных затрат *T* в исходных матрицах из выше приведенного примера столбец 1 изменяется в соответствии с соотношением:

(№ группы + номер бригады + номер строки матрицы)/2.

2.5. Контрольные вопросы

1. Содержание задачи структурного синтеза больших систем управления.
2. Постановка задачи структурного синтеза. Примеры.
3. Постановки задач структурной оптимизации. Примеры.

4. Основные методы решения задачи структурной оптимизации и их сравнительный анализ.

2.6. Указания по оформлению отчета

Отчет по лабораторной работе должен содержать:

1. Исходные матрицы стоимости затрат *C* и временных затрат *T*, необходимые ограничения и текущие преобразования матриц.
2. Дерево решения.
3. Таблицу с указанием номера задачи и узла, в котором она решается.
4. Блок-схему решения задачи.
5. Листинг программы.
6. Распечатку результатов решения.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. **Денисов А.А., Колесников Д.Н.** Теория больших систем управления -Л.: Энергоиздат, 1982. – 288с.
2. **Майника Э.** Алгоритмы оптимизации на сетях и графах. - М.: Мир, 1981. – 323 с.
3. **Перегудов Ф.И., Тарасенко Ф.П.** Введение в системный анализ – М.: Высш. шк., 1989. – 367 с.
4. **Ху Т.** Целочисленное программирование и потоки в сетях. – М.: Мир, 1974. – 519 с.
5. **Черемных С.В., Семёнов И.О., Ручкин В.И.** Структурный анализ систем: IDEF-технология. – М.: Финансы и статистика, 2003. – 208 с.