УДК 658.52.011.56

МИНИМИЗАЦИЯ ВРЕМЕНИ ФОРМИРОВАНИЯ ИНФОРМАЦИОННЫХ МАССИВОВ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ УПРАВЛЕНИЯ

БУРКОВ В. Н., КЛЕТИН В. А.

(Москва)

Рассматривается методика минимизации времени формирования информационных массивов в АСУ, базирующаяся на оптимизации последовательности работы с ними.

Дается формальная постановка задачи, приводятся случаи, допускающие применение эффективных процедур и соответствующие им алгоритмы. Описываются результаты предлагаемой методики применительно к построению математического обеспечения конкретной АСУ.

1. Введение

Функционирование автоматизированных систем управления (АСУ), как правило, бывает связано с обработкой большого числа информационных массивов (ИМ) и формированием потока выходных документов, что требует существенных затрат машинного времени. Попытки сократить затраты такого рода обычно опираются на использование банков данных (БД), тщательную проработку вопросов размещения массивов на машинных носителях (магнитные ленты, диски, барабаны и т. п.), способов организации массивов (индексно-последовательный, относительный, прямой, последовательный) и систем кодирования [1—3].

Основными элементами информационной базы АСУ являются ИМ, предназначенные для длительного или кратковременного хранения информации. Массивы данных, размещаемые в БД системы для длительного хранения, централизованного их обновления и использования различными потребителями информации, называются основными. Все другие массивы, предназначенные для кратковременного хранения и использования какого-то массива каким-то конкретным потребителем информации, называются рабочими.

Благодаря тому, что содержимое различных рабочих ИМ часто пересекается, указанные выше подходы к минимизации времени обработки информации можно дополнить построением такого порядка формирования ИМ, который бы сократил время их формирования за счет взаимодействия друг с другом. Иными словами, часть рабочих массивов формируется не непосредственно на основании основных ИМ, хранимых в БД, а с использованием ранее сформированных и «более доступных» массивов. При этом допустимо формирование промежуточных ИМ, которые не необходимы для работы АСУ, но на основании которых сокращается время формирования рабочих массивов, необходимых для работы АСУ.

Далее приводятся формальная постановка этой задачи, случаи, допускающие применение эффективных процедур, и соответствующие им алгоритмы, описание пакета программ, реализующих этот подход при разработке АСУ завода.

2. Формальная постановка задачи

Пусть задан взвешенный, ориентированный граф $G(X, U)$, где X — множество вершин ($|X|=n$), U — множество дуг ($|U|\leqslant n^2$). Каждой дуге $(i, j)\in U$ присвоен вес $t(i, j)\geqslant 0$. На множестве вершин X графа $G(X, U)$

выделены вершина $s \in X$, не имеющая заходящих в нее дуг, и подмножество вершин $Q \subset X \setminus \{s\}$, такое, что любая вершина $q \in Q$ достижима из s по дугам графа $G(X, U)$. На $G(X, U)$ требуется выделить подграф $G^1(X^1, U^1)$, такой, что [4—6] $G^1(X^1, U^1)$ является деревом с корнем в s, $Q \subseteq X^1$ и

$$\sum_{(i,j) \in U^1} t(i,j) \to \min. \tag{1}$$

Покажем, что указанная выше задача формирования ИМ может быть сведена к задаче (1). Поставим в соответствие каждому ИМ вершину графа $G(X, U)$. Среди ИМ присутствуют как ИМ, необходимые для работы АСУ, так и промежуточные ИМ, которые не необходимы для работы АСУ, но на основании которых сокращается время формирования рабочих массивов. ИМ, необходимые для работы АСУ, будем называть обязательными, т. е. эти массивы обязательно должны быть сформированы. Две вершины соединим дугой $(i, j) \in U$, если j-й массив можно сформировать на основании i-го. Каждой дуге $(i, j) \in U$ присвоен вес $t(i, j)$, равный времени формирования j-го массива на основании i-го. На множестве вершин X графа $G(X, U)$ выделено подмножество вершин $Q \subset X$, отвечающее обязательным ИМ, и выделена вершина $s \in X \setminus Q$, отвечающая основным ИМ. Таким образом задача формирования ИМ, необходимых для работы АСУ, за минимальное время сводится к задаче отыскания минимального дерева с корнем в вершине s на графе $G(X, U)$, отвечающем данной задаче.

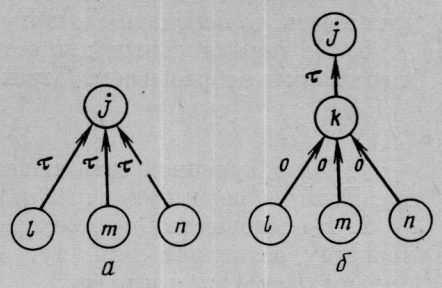

Рис. 1

Можно показать, что к (1) сводится и случай, когда j-й массив формируется на основании нескольких массивов за время τ. В этом случае в граф $G(X, U)$ вводится фиктивная вершина k, достижимая из тех же вершин, что и j, за нулевое время. На новом графе $G^1(X^1, U^1)$ вершина j достижима только из k, причем $t(k, j) = \tau$ (рис. 1, *а*, *б*).

Можно показать, что к сформулированной выше задаче (1) могут быть сведены следующие задачи: прокладка линий электропередач, ниток газо- и нефтепроводов, принятие решений о разработке месторождений, передача сообщений и т. п. [5—7].

3. Алгоритмы решения задачи

Сформулированная задача является задачей дискретного программирования, что требует для ее решения в общем случае использования различного рода переборных процедур, таких, как методы типа ветвей и границ, алгоритм Балаша и т. п. Однако можно выделить ряд условий, при которых возможно применение эффективных, т. е. гарантирующих глобально-оптимальное решение и исключающих полный перебор, процедур. Тривиальным примером задачи такого рода является случай, когда в (1) из s должна быть достижима лишь одна вершина из $X \setminus \{s\}$; очевидно, что в этом случае можно воспользоваться методом кратчайшего пути [8].

Если «обязательными» являются все вершины подмножества $X \setminus \{s\}$, то возможно использование ряда процедур, эффективных на различного рода графах, содержательное описание которых приводится ниже.

Поиск на сетях. Пусть $G(X, U)$ в (1) является сетью, т. е. не содержит контуров [s], а $Q = X \setminus \{s\}$. В этом случае для решения (1) можно воспользоваться следующим алгоритмом [5].

Алгоритм 1

1. На полученной сети выделяем вершину, принадлежащую первому ярусу вершин, считая от корневой.

2. Выделенную вершину стягиваем в корневую.

3. Вес стянутой дуги добавляем к ранее полученному, а индекс запоминаем.

4. Если граф исчерпан, т. е. все вершины стянуты в корневую вершину, то алгоритм закончен, на шаге 3 последней итерации получено искомое решение, в противном случае переходим к шагу 5.

5. Проверяем наличие на полученной сети параллельных дуг. Если такие есть, то переходим к шагу 6, в противном случае — к шагу 1.

6. Из параллельных дуг оставляем лишь ту, вес которой минимален, а остальные отбрасываем. Переходим к шагу 1.

Алгоритм 2

1. На полученной сети выделяем ранее не просматривавшуюся вершину. Если таковых нет, перейти к шагу 3, в противном случае — к шагу 2.

2. На множестве дуг, заходящих в выделенную на предыдущем шаге вершину, оставляем лишь дугу с минимальным весом, а остальные отбрасываем. Переходим к шагу 1.

3. Конец алгоритма. Полученное дерево является искомым. Доказательства конечности алгоритма и оптимальности полученного решения тривиальны. Первое следует из конечности графа, второе — из свойств сетей.

Если рассмотреть более общий случай, а именно $Q=X^1\subset X\setminus\{s\}$, то алгоритмы 1, 2, вообще говоря, оказываются неверны, они позволяют получать лишь верхние границы суммарного веса дуг минимального дерева. Для вычисления нижней границы можно воспользоваться следующей процедурой.

Алгоритм 3

1. Все вершины подмножества $X^1\subset X$ помечаем, а переменной S присваиваем вес, равный нулю.

2. Если все помеченные вершины стянуты в корневую, то перейти к шагу 9, нет — к следующему шагу.

3. На множестве помеченных вершин $X\setminus\{s\}$ выбираем произвольную.

4. На множестве дуг, заходящих в выбранную на шаге 3 последней итерации вершину, выбираем дугу с минимальным весом Δ.

5. Вес всех дуг, заходящих в выбранную вершину, уменьшить на величину Δ.

6. Величину S увеличить на Δ.

7. «Стянуть» вершины, соединяемые дугой с нулевым весом, и пометить вновь образованную вершину.

8. Если при стягивании вершин образовались параллельные дуги, то оставить лишь одну из них, вес которой является наименьшим, а другую отбросить. Перейти к шагу 2.

9. Конец алгоритма. Величина S равна нижней оценке суммарного веса дуг минимального дерева, а стянутые дуги соответствуют дугам подмножества U^1.

Поиск на планарных нуль-графах. Если исходный граф $G(X, U)$ таков, что каждой дуге $(i, j)\in U$ соответствует дуга $(j, i)\in U$, причем $t(i, j)\cdot t(j, i)=0$, то такой граф будем называть нуль-графом.

Задачи оптимизации траектории движения частицы в силовых полях часто могут быть сведены к решению (1) на графах такого рода, где веса дуг $t(i, j)$ отображают затраты энергии на переход из i-й точки поля в j-ю. Пусть $U^1\subset U$ — подмножество дуг с ненулевым весом, а $U''=U\setminus U^1\subset U$. Можно показать, что поиск на $G(X, U)$ может быть сведен к поиску на графе $G_1(X_1, U_1)$, такому, что $G_1(X_1, U_1^1)$ является сетью, симметричной сети $G_1(X_1, U_1'')$, стягиванием в одну вершину контуров, суммарный вес дуг которых равен нулю. Если $G(X_1, U_1)$ является планарным, то двойственной (1) является задача поиска минимального разреза на двойствен-

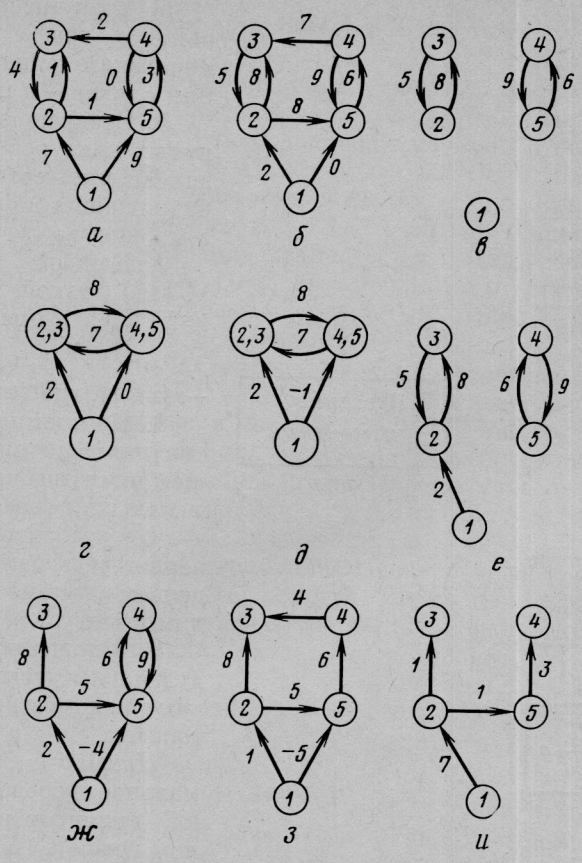

Рис. 2

ном $G_1(X_1, U_1^1)$ бисвязном планарном орграфе [9]. Но в [9] показано, что величина минимального разреза на планарных графах равна максимальной циркуляции и является целочисленной величиной. Отсюда следует, что если исходный граф является нуль-графом, удовлетворяющим приведенным выше условиям, то поиск минимального дерева может осуществляться методами линейного программирования.

Поиск на графах с бикомпонентами. Если $Q=X\setminus\{s\}$, но $G(X, U)$ является орграфом с бисвязными компонентами, то можно воспользоваться более сложным алгоритмом, предназначенным для поиска максимальных деревьев на произвольных графах [7]. Решение задачи (1) в этом случае предваряется простой процедурой: выбирается число $M=\max t(i, j)$, после чего каждой дуге $(i, j)\in U$ присваивается вес $r(i, j)=M-t(i, j)$. Затем задача решается алгоритмом, содержательное описание которого приводится ниже.

Алгоритм 4

1. На полученном графе для каждой вершины $j\in X\setminus\{s\}$ выбираем дугу (i, j), такую, что $r(i,j)=\max_k r(k,j)$, и помечаем ее. Если выбранные дуги не образуют бикомпоненты, то переходим к шагу 4, в противном случае — к шагу 2.

2. Выбираем произвольный контур, образованный помеченными дугами и стягиваем его в одну из принадлежащих ему вершин.

3. Если в полученном графе нет бисвязных компонент, переходим к шагу 1, в противном случае — к шагу 2.

4. Если на графе были стянуты контуры, то переходим к шагу 5, если же такие контуры отсутствуют — к шагу 9.

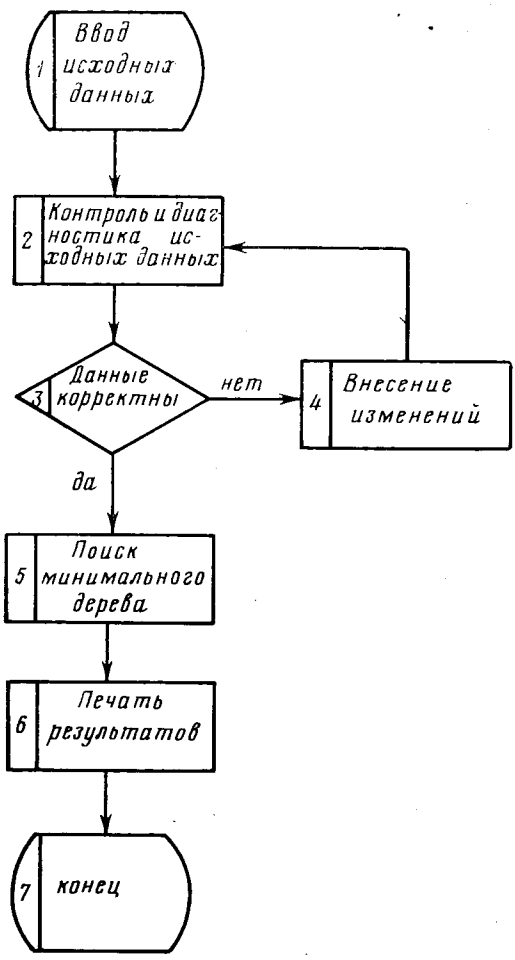

Рис. 3

5. Если число вершин полученного графа равно $|X|$, то переходим к шагу 9, в противном случае — к шагу 6.

6. Возвращаем один из стянутых контуров V_q в граф, обозначая множество вершин, принадлежащих контуру V_q, $X(V_q)$, а множество дуг — $U(V_q)$.

7. Каждой дуге $(i, j) \in U \setminus U(V_q)$, такой, что $i \in X \setminus X(V_q)$, $j \in X(V_q)$, присваиваем вес $r^1(i, j) = r(i, j) + \min\limits_{(k,l) \in U(V_q)} r(k, l) - r(m, j)$, где $(m, j) \in U(V_q)$.

8. На подмножестве дуг, рассматривавшихся на предыдущем шаге, выбираем дугу с максимальным весом и помечаем ее, а дугу, заходящую в ту же вершину и помеченную ранее, делаем непомеченной. Переходим к шагу 5.

9. Множество помеченных дуг на полученном графе образуют искомое дерево. Конец алгоритма.

Пример 1. Определить минимальное дерево с корнем в x_1 на ориентированном взвешенном графе $G(X, U)$, изображенном на рис. 2, *а*. Граф, на котором ищется максимальное дерево, полученный из исходного, изображен на рис. 2, *б*, последовательность преобразований, отображающая реализацию алгоритма 4,— на рис. 2, *в* — *з*, а конечный граф, являющийся минимальным деревом,— на рис. 2, *и*.

В общем случае, как отмечалось выше, при решении (1) приходится пользоваться переборными процедурами, подробный анализ эффективности которых содержится в [10]. На основании этого анализа был разработан пакет программ, описание которого приводится ниже.

4. Пакет программ Delta

Общее описание пакета. Назначением пакета программ является решение задачи (1), предваряемое вводом, контролем и диагностикой исходных данных. Вывод результатов счета осуществляется в виде матрицы смежности вершин полученного дерева и величины рекорда, равного сумме весов дуг, принадлежащих дереву. Исходными данными являются матрица смежности вершин первичного графа и список «обязательных» вершин конечного дерева.

Основу пакета составляет программа Delta 3, осуществляющая поиск оптимального решения (1) методом типа ветвей и границ [10].

Оценка $\Delta(\pi)$ частичного плана π, содержащего подмножество вершин $Q \subset X$ исходного графа $G(X, U)$, вычисляется на основании выражения

$$\Delta(\pi) = \sum_{x_j \in Q} \min_{x_i \in X_i^1} t(i, j) + \sum_{x_k \in X} \min_{x_l \in X \setminus Q} t(k, l),$$

где $X_1^1 \subset Q$, причем вершины подмножества X_1^1 в упорядочении π стоят до вершины x_j.

Агрегированная блок-схема алгоритма работы пакета приведена на рис. 3. Программы ввода, вывода, контроля и диагностики исходных данных реализованы на языке Кобол, а программа поиска оптимального решения — на языке Фортран IV. Пакет программ Delta предназначен для работы на ЭВМ серии ЕС, операционная система ОС ЕС, требуемый объем оперативной памяти составляет 100 Кбайт. Ограничением на использование пакета является требование хранения всех исходных данных в оперативной памяти ЭВМ. Пакет Delta был использован для разработки математического обеспечения первой очереди АСУ. Ниже приводятся основные характеристики решавшихся задач.

Анализ эффективности пакета при решении реальных задач. В состав подсистем первой очереди АСУ завода входят технико-экономическое планирование (ТЭП), техническая подготовка производства (ТПП), сбыт, реализация, оперативное управление (ОУ), охватывающие весь цикл контроля производства от формирования портфеля заказов до выдачи документации на отгружаемую продукцию и осуществляющие взаимный обмен информацией посредством связующих массивов. Исходный граф $G(X, U)$, отображающий все множество массивов и связей между ними, содержит $|X|=41$ вершину и $|U|=89$ дуг. Время поиска локально оптимального решения группой экспертов составило 4 ч, пакетом Delta — 2 ч, причем программная реализация последнего позволяет получить ежедневный выигрыш во времени счета 1,5÷3 ч по сравнению с решением, предложенным экспертами, что соответствует ежегодному экономическому эффекту 50 тыс. руб.

ЛИТЕРАТУРА

1. *Мамиконов А. Г., Пискунов А. Н., Цвиркун А. Д.* Модели и методы проектирования информационного обеспечения АСУ. М.: Статистика, 1978.
2. *Бурков В. Н., Соколов В. Б.* Оптимальное размещение информационных массивов в памяти на магнитных лентах для случая двунаправленного поиска.— Автоматика и телемеханика, 1969, № 4, с. 107–117.
3. *Соколов Р. В.* Экономико-информационное моделирование процессов преобразования информации в АСУП. Л.: Изд-во Ленинградск. гос. ун-та, 1980.
4. *Зыков А. А.* Теория конечных графов. Т. 1. М.: Наука, 1969.
5. *Гроппен В. О.* Задача о минимальном каркасе сети.— В кн.: Современные проблемы управления. М.: Наука. 1974, с. 167–168.
6. *Романовский И. В.* Алгоритмы решения экстремальных задач. М.: Наука, 1977.
7. *Mipieka E.* Optimisation algorithms for networks and graphs. N. Y.: Dekker, 1978.
8. *Бурков В. Н., Горгидзе И. А., Ловецкий С. Е.* Прикладные задачи теории графов. Тбилиси: Мецниереба, 1974.
9. *Гроппен В. О.* Связь задач о максимальной циркуляции и минимальном разрезе в сильносвязанном графе с задачей о неоднородном потоке.— В кн.: Научн-техн. сб. Электронная техника, сер. 9, Автоматизированные системы управления. М., 1972, вып. 3, с. 117–124.
10. *Клетин В. А.* Анализ эффективности комбинаторных алгоритмов поиска экстремальных деревьев на взвешенных орграфах.— Автоматика и телемеханика, 1979, № 11, с. 134–141.

Поступила в редакцию
8.XII.1980

MINIMIZING THE TIME OF DATA FILE COMPILATION IN MANAGEMENT INFORMATION SYSTEMS

BURKOV V. N., KLETIN V. A.

A methodologyis described for minimizing the time of data file compilation in MIS by optimizing the sequence of their processing.

The problem is stated in formal terms; cases are described where effective procedures and associated algorithms are applicable. The results of using the methodology in design of a specific MIS software are described.
