7. М и х л и н Г.З. Методы моделирования операций мини-ЭВМ и использование открытых подпрограмм. "Программирование", 1977, I.

С.В. Смирнов

ЗАДАЧА СЕГМЕНТАЦИИ И ОДИН СПОСОБ ЕЕ РЕШЕНИЯ

П о с т а н о в к а з а д а ч и

Пусть задана конечная упорядоченная последовательность $A=\{A_1, A_2, \dots, A_L\}$ элементов произвольной природы, и для элементов последовательности определены значения функционала F, так что $F(A_i)=c_i \geq 0$, $i=1,2,\dots,L$.

Определим следующую задачу сегментации последовательности:

Требуется "разрезать" исходную последовательность A на ряд подпоследовательностей (сегментов):

$\{A_1, A_2, \dots, A_{j_1}$;

$A_{j_1+1}, A_{j_1+2}, \dots, A_{j_2}$;

. . . . . .

$A_{j_{K-1}+1}, A_{j_{K-1}+2}, \dots, A_{j_K}$;

. . . . . .

. . . . . $A_L\}$,

или, что равносильно, указать набор индексов $J=\{j_1, j_2, \dots, j_K, \dots\}$ где $K<L$, так, чтобы

I) число элементов в каждом из сегментов было не меньше M и не больше B (предложение о допустимых соотношениях между числами L, M, B дается ниже);

2) сумма $\sum_{j\in J} c_j$ была минимальна. (Слагаемые суммы суть значения функционала F для так называемых K - элементов последовательности A, т.е. тех ее элементов, которые оказались непосредственно слева от сделанных "разрезов").

Корректность постановки задачи сегментации фиксирует сле-

дующая

т е о р е м а: существует, по крайней мере, одна определяемая условием (I) сегментация последовательности A, если тройка чисел L, M, B удовлетворяет любому из двойных неравенств:

$$1 \leqslant M \leqslant \frac{L}{R+1}, \text{ где } R = \left[\frac{L}{B}\right];$$

$$L \geqslant B \geqslant \frac{L}{S}, \text{ где } S = \left[\frac{L}{M}\right].$$

Здесь и далее квадратные скобки означают взятие целой части стоящего в них выражения.

Постановка задачи сегментации стимулируется многими практическими приложениями, а предлагаемый критерий оптимальности сегментации характерен для задач автоматизации проектирования пространственной или временной структуры сложных объектов при их декомпозиции.

К этой формальной схеме сводится, например, задача составления оптимального расписания измерений некоторого параметра, если в течение времени контроля детерминированно меняются свойства самого параметра (диапазон амплитуд, информативность и т.п.) и (или) другие связанные с контролем факторы, например, неравномерная по времени загруженность измерительной системы, а диапазон частот измерений параметра ограничен.

Особый интерес представляет интерпретация задачи сегментации, связанная с расчленением длинных программ с целью экономии времени их реализации под управлением операционных систем.

М е т о д и а л г о р и т м р е ш е н и я

Введем вектор $X = \{x_1, x_2, \ldots, x_L\}$, фиксирующий заданную сегментацию последовательности A:

$$x_i = \begin{cases} \text{I - если } A_i \text{ есть } K\text{-элемент в заданной сегментации} \\ \quad \text{последовательности } A; \\ \text{0 - в противном случае.} \end{cases}$$

Тогда задачу сегментации, описанную выше, можно сформулировать следующим образом:

минимизировать

$$\sum_{i=1}^{L} c_i x_i \qquad (1)$$

при условиях:

$$x_i = \begin{cases} 1, \\ 0, \end{cases} \quad i = 1,2,\dots,L; \qquad (2)$$

$$x_i = 0, \qquad i = 1,2,\dots,M-1, L-M+1, L-M+2,\dots,L; \qquad (3)$$

$$\sum_{k=0}^{M-1} x_{j+k} \leqslant 1, \quad j = M, 2M, 3M,\dots, nM,\dots, \text{где } n \leqslant \frac{L+1}{M} - 1; \qquad (4)$$

$$\sum_{k=0}^{B-M} x_{j+k} \geqslant 1, \quad j = M, 2M, 3M,\dots, nM,\dots, \text{где } n \leqslant \frac{L-B}{M} + 1; \qquad (5)$$

$$c_i \geqslant 0, \quad i = 1,2,\dots,L. \qquad (6)$$

Здесь условие (3) гарантирует, что длина первого и последнего сегмента в заданной сегментации будет не меньше M; (4) – то же для остальных сегментов; условие (5) ограничивает длину сегментов величиной B.

Для решения задачи (I) – (6) можно воспользоваться известными алгоритмами решения задач целочисленного линейного программирования с булевыми переменными общего вида [I]. Однако специфика условий (3) – (6), а также конечность числа допустимых вариантов сегментации последовательности A позволили предложить простой в вычислительном отношении и весьма экономный по памяти комбинаторный алгоритм, основанный на общей идее метода ветвей и границ [I].

В приложении к данной задаче метод ветвей и границ состоит в направленном движении по вершинам дерева – допустимым вариантам сегментации, – получаемого при фиксировании части переменных x_i.

Дерево ветвления строится следующим образом. Вершины первого уровня получаем, фиксируя поочередно допустимые варианты длины первого сегмента, вершины второго уровня – фиксируя поочередно (и отдельно для каждой вершины первого уровня) допустимые длины второго сегмента, и т.д. Каждая вершина дерева ветвления соответствует, таким образом, некоторой допустимой (возможно частичной, когда удовлетворена лишь часть условий (4), (5)), сегментации

последовательности A и оценивается суммой значений функционала F для K -элементов этой сегментации.

Вершины, где выполнены все условия (3) - (5), будем называть "висячими", остальные вершины являются "промежуточными". Очевидно, оценка висячей вершины равна значению критерия (I) для соответствующего допустимого решения задачи сегментации. Дальнейшее ветвление из висячей вершины либо невозможно, либо в силу выражения (6) приведет только к неменьшему значению критерия (I).

На первом шаге алгоритма производится наиболее короткая по числу пройденных вершин сегментация последовательности A . Оценка достигнутой при этом висячей вершины запоминается как минимальное значение критерия (I), а соответствующая сегментация - как оптимальная.

Полный перебор допустимых вариантов сегментации последовательности A отвечает просмотру всех вершин дерева ветвления. Сокращение перебора осуществляется путем запрета дальнейшего продвижения из достигнутой промежуточной вершины, если ее оценка больше минимального значения критерия, так как в силу (6) оценки висячих вершин, которые могут быть достигнуты из данной промежуточной, будут также больше минимального значения критерия.

Если при достижении некоторой висячей вершины ее оценка оказывается не больше минимального значения критерия, то последнему присваивается значение этой оценки, и соответствующая сегментация фиксируется как оптимальная.

Рассмотрим программу процедуры на языке ПЛ/I ДОС ЕС, реализующей описанный алгоритм сегментации:

```
I.  SEGM: PROCEDURE(L,M,B,C,Z,NK);
2.  DECLARE(L,M,B,PE,IG,IM,P,PP,NK(1))
                BINARY FIXED,
            (MINZ,Z(1),C(1)) DECIMAL FLOAT;
3.      MINZ=1E6; PE=L-M; IG=L-B;
6.      P=1; Z(1)=0; NK(1)=0;
9.  NP: PP=P; P=P+1; NK(P)=NK(P)+B;
I2.     DO WHILE(NK(P)>PE); NK(P)=NK(P)-1; END;
I5. ZP: IM=NK(PP)+M;
```

```
16.        DO  WHILE(NK(P)>=IM);
17.        Z(P)=Z(PP)+C(NK(P));
18.        IF  Z(P)<=MINZ THEN;
           IF  NK(P)<IG THEN GOTO NP;
19.        ELSE DO; MINZ=Z(P);
21. PUT    EDIT('РЕГИСТРАЦИЯ РЕЗУЛЬТАТА:')(SKIP(2),X(4),A);
22. PUT    EDIT('КРИТЕРИЙ=',MINZ)(SKIP,X(4),A,E(12,3));
23. PUT    EDIT('ОПТ. СЕГМЕНТАЦИЯ:')(SKIP,A);
24. DO     J=2 TO P; PUT EDIT(NK(J),' ')(F(3),A); END;
27.        END;
28.        NK(P)=NK(P)-1;
29.        END;
30.        P=P-1; PP=P-1;
32.        IF P>1 THEN DO; NK(P)=NK(P)-1; GOTO ZP; END;
36. PUT    EDIT('КОНЕЦ ГЛОБ MIN РАВЕН',MINZ)(SKIP,A,E(12,3));
37.        END.
```

Для удобства операторы программы пронумерованы:

1. Формальные параметры:

   L – длина последовательности A;

   M, B – минимально и максимально возможная длина сегмента, соответственно;

   C – массив значений функционала F для элементов последовательности A (достаточно передавать процедуре лишь значения функционала для A_i, где $i=M, M+1, \ldots, L-M$);

   Z, NK – рабочие массивы для размещения оценок вершин дерева ветвления и запоминания номеров K-элементов последовательности A в исследуемом варианте сегментации.

2. Описание переменных. Длины массивов C, Z, NK заданы формально. Истинные их значения – L для C и $\left[\frac{L}{M}\right]$

для Z , NK - должны определяться в вызывающей программе.

3. Переменной $MINZ$ - минимальному значению критерия - присваивается достаточно большое начальное значение (теоретически $+\infty$).

4. PE - максимально возможный номер для последнего, если отсчет начинать слева K -элемента (см. условие (3)).

5. IG - минимально возможный номер для последнего K-элемента. Если сегментация последовательности A производится так, что сначала выбирается допустимая длина первого сегмента, или, равно, допустимое положение первого K -элемента, затем второго K -элемента и т.д., то выбор положения очередного K -элемента, номер которого оказывается не меньше IG , означает получение сегментации, при которой выполнены все условия (3) - (5), т.е. в дереве ветвления достигается висячая вершина.

6-8. Определение характеристик "корня" дерева ветвления.

9,10. Переход на следующий уровень ветвления - к перебору длин следующего сегмента. Будем говорить, что P - номер этого, а PP - номер предыдущего уровня (сегмента), хотя в действительности эти номера на единицу меньше.

11. Выбор первой вершины на P -м уровне - номер K -элемента берется максимально возможным.

12-14. Номер K -элемента не должен превышать PE.

15. IM- наименьший из допустимых номеров K -элемента P -го сегмента.

16. Организуется поочередное исследование вершин P -го уровня, или перебор длин P -го сегмента.

17. Вычисление оценки исследуемой вершины.

18-27. Если оценка вершины не превышает минимального значения критерия, то производится проверка: является ли эта вершина промежуточной. Если "да", то исследование вершин данного уровня прерывается и выполняется переход на следующий уровень ветвления - передача управления оператору 9. Если вершина является висячей, то ее оценка присвои-

вается минимальному значению критерия и выводится на *SYSLST*. Там же фиксируется соответствующая сегментация последовательности A в виде номеров K-элементов. В общем случае оптимальная сегментация определяется лишь в последней регистрации результата, однако преимущество такой регистрации - экономия памяти и фиксирование нескольких оптимальных решений задачи, если таковые имеются, но в общем случае не всех имеющихся, так как не производится ветвления из висячей вершины (см. определение висячей вершины).

28. Переход к исследованию очередной вершины на P-м уровне - уменьшение длины P-го сегмента на единицу.

30,31. Подготовка возврата к прерванному ранее (оператор I8) исследованию вершин предыдущего уровня.

32. Если предыдущий уровень соответствует "корневой" вершине, то на *SYSLST* выводится сообщение о завершении исследования дерева ветвления и печатается минимальное значение критерия (I).

33,34. Выбор очередной вершины на P-м уровне - очередного варианта длины P-го сегмента - и переход к организации ветвления из этой вершины.

Пример расчета и заключительные замечания:

$L = 80$, $M = 6$, $B = 10$, $c_i = 2 + \sin(0{,}35i) + \sin(0{,}65i)$, $i = 1,2,\ldots,L$.

Минимальное значение критерия (I), равное 8,346, достигается в следующей сегментации (даны номера K-элементов):

8, I7, 27, 36, 46, 56, 66, 74.

Численные эксперименты проводились на ЭВМ М-4030 (расчет приведенного выше примера произведен за I2 с.). Малый вычислительный опыт не позволяет сделать надежных качественных выводов о работе алгоритма. Отметим только, что он более болезненно реагирует на увеличение разности $B - M$, чем на увеличение числа переменных

задачи, равного $i-2M+1$.

Литература

I. Корбут А.А., Финкельштейн Ю.Ю. Дискретное программирование. М., "Наука", 1969, 368 с.

В.В. Куликов, С.И. Трещев

МОДЕЛИРОВАНИЕ ИНФОРМАЦИОННЫХ МАССИВОВ

При отладке и тестовой проверке системы программного обеспечения (СПО) измерительно-вычислительного комплекса (ИВК) возникает задача моделирования информационных массивов с заданной структурой элементов. В самом общем виде задача заключается в генерации цепочек (строк) символов при условии, что между совокупностями цепочек символов и между отдельными символами цепочек заданы отношения.

СПО ИВК имеет в своем составе ряд подсистем, обеспечивающих преобразование данных, поступающих с регистров устройств согласования ЦВМ с системой сбора данных, в форму, используемую при вторичной обработке.

Подсистему СПО можно рассматривать как алгоритмический преобразователь входной информации в выходную. Описание алгоритма преобразования задано алгоритмическим языком, на котором реализована совокупность программ подсистемы. Рис.I поясняет взаимодействие подсистемы A_i с информационными массивами.

Входной информационный массив X_i от подсистемы A_{i-1} ($i \geq 1$) или внешней среды (A_0) поступает в подсистему A_i. В процессе преобразования подсистема A_i использует информационный массив G_i, генерирует массив Y_i и обновляет часть данных массива F_i. Массивы G_i и F_i составлены из данных, не входящих непосредственно в очередной экземпляр массива X_i или Y_i. Например, массив $G_i \cup F_i$ могут составлять константы, таблицы, номера алгоритмов обработки, списки обработанных параметров, логические шкалы и т.д., а массив $X_i \cup Y_i$ отдельные значения обрабатываемой реализации и другие необходимые для подсистемы A_i атрибуты параметра