BÀI TOÁN LUỒNG VÀ ỨNG DỤNG TRONG THỰC TẾ

VŨ ĐÌNH HÒA, ĐOÀN PHÚC, ĐÀO ĐÌNH KHẢ

1. MỞ ĐẦU

Trong số báo 3 − 1988 Tạp chí Khoa học Tính toán và điều khiển, chúng tôi đã giới thiệu bài toán luồng và một số các bài toán có thể giải được bằng cách đưa nó về dạng bài toán luồng và sử dụng thuật toán Fork−Funkerson để giải. Bài toán số 5 được nêu trong số báo đó là một bài toán liên quan đến việc dự báo số máy bay thực tế có thể sử dụng để thực hiện những lịch biểu cất cánh và hạ cánh cho trước trên một số sân bay.

Với thuật toán Fork−Funkelson chúng ta còn có thể bố trí lịch biểu hành trình cho mỗi máy bay để đảm bảo số máy bay của ta ít nhất có thể.

Trong bài này còn có một chương trình máy vi tính được viết theo ngôn ngữ Basic cho thuật toán tìm luồng lớn nhất của Fork−Funkelson.

Trước khi trình bày bài toán bố trí lịch biểu trên sân bay chúng ta nhắc lại rằng những bài toán sau có thể giải bằng cách vận dụng thuật toán luồng và cách thức giải chúng đã được trình bày trong bài báo số 3−1988 :

− Bài toán phân công công việc (còn gọi là bài toán vợ chồng).

− Bài toán tìm số cực tiểu các dòng phủ của một ma trận.

− Bài toán tìm số cực đại các phần tử khác không không cùng nằm trên một hàng hoặc một cột.

− Bài toán bố trí công nhân trên một băng chuyền sản xuất.

2. BÀI TOÁN LỊCH BIỂU SÂN BAY

Giả sử ta có n sân bay và i và j là hai sân bay khác nhau, thời gian t_{ij} là thời gian một chiếc máy bay phải bay từ sân bay i sang sân bay j. Trên mỗi sân bay i có một lịch biểu F_i như sau : máy bay hạ cánh vào thời điểm a_i và cất cánh vào thời điểm b_i ($a_i \leqslant b_i$). ^ bi).

Bài toán 5 : Tìm số máy bay tối thiểu để thỏa mãn các lịch biểu F_i, biết rằng $t_{ij} > 0$ với $i \neq j$ và $t_{ij} = 0$ với $i = j$, ngoài ra ta có bất đẳng thức tam giác :

$$t_{ij} \leqslant t_{ik} + t_{kj} \quad \text{cho mọi } k, i, j \in \overline{1, n} \tag{1}$$

Giải :

Ta nói rằng lịch biểu F_i được sắp trước lịch biểu F_j nếu :

$$b_i + t_{ij} \leqslant a_j \tag{2}$$

và ta ký hiệu $F_i > F_j$.

Khi đó quan hệ $>$ có tính chất phản đối xứng và bắc cầu.

Tức là :

− Nếu $F_i > F_j$ thì không có $F_j > F_i$. (3)

− Nếu $F_i > F_j$ và $F_j > F_k$ thì $F_i > F_k$ (4)

Ngoài ra ta gọi các lịch biểu $F_{i_1}, F_{i_2}, \ldots, F_{i_k}$ lập thành một chuỗi nếu :

$$F_{i_j} > F_{i_{j+1}}, \text{ cho mọi } j = 1, 2, \ldots, k-1.$$

Và ta ký hiệu $F_{i_j} > F_{i_2} > F_{i_3} > \ldots > F_{i_k}$.

Như vậy có thể thấy rằng số máy bay tối thiểu ứng với số chuỗi tối thiểu có thể phân tích từ hệ lịch biểu đã cho. Nếu gọi hai lịch biểu F_i và F_j là hai lịch biểu độc lập nếu như chúng không thể so sánh được với nhau, thì số lịch biểu độc lập ít nhất của một hệ lịch biểu tối đa (tức là không thể bổ sung hơn được) là n − k, với k là số lớn nhất các phần tử 1 không cùng nằm trên cùng hàng và cột của ma trận được thiết lập bởi quy tắc :

$$\alpha_{ij} = \begin{cases} 1 \text{ nếu } F_i > F_j \\ 0 \text{ trong trường hợp khác.} \end{cases}$$

Trên cơ sở của nhận xét trên ta có thuật toán xác định số máy bay cực tiểu như sau:

4. Thiết lập ma trận $Q = (\alpha_{ij})$ với $\alpha_{ij} = 1$ nếu $F_i > F_j$, $\alpha_{ij} = 0$ cho mọi trường hợp khác.

2. Giải bài toán tìm dòng phủ nhỏ nhất cho ma trận Q, và số cực đại các phần tử 1 độc lập.

3. Giả sử các phần tử độc lập là (i_1, i_2), (i_3, i_4), ..., (i_{2k-1}, i_{2k}), Khi đó n−k máy bay của ta dự định chọn là các máy bay thực hiện:

— các chuỗi lịch biểu tạo thành từ $F_{i_1}, F_{i_2}, ..., F_{i_{2k}}$,

— các lịch biểu độc lập còn lại trong số F_t với $t = i_r$, $r \in \overline{1, 2k}$

Để kết thúc bài này, phần còn lại chúng tôi xin giới thiệu một chương trình máy vi tính viết bằng ngôn ngữ Basic cho thuật toán tìm luồng lớn nhất:

```
10 REM THUẬT TOÁN FORD–FULKERSON TÌM LUỒNG LỚN NHẤT TRÊN MẠNG VẬN TẢI
20 PRINT « Thông số của mạng phải đưa vào dưới dạng 2 ma trận A (N, N) và C (N, N), ở
dây A (I, J) = 1 nếu có cùng (I, J) với khả năng thông qua C (I, J). Khi đó A (J, I) = – 1)
và C (J, I) = 0. Ma trận F (N, N) chỉ giá trị của luồng »
30 PRINT « Đầu vào là đỉnh số 1, đầu ra là đỉnh số N »
40 INPUT « Số đỉnh của đồ thị », N
50 DIM A (N, N), F (N, N), EX (N), C (N, N), AA (N, N)
60 DIM LU (N, N*N), B (N, N), O (N, N), DU (N)
70 INPUT « Vào số liệu từ đĩa hay không ? (Y/N) » ; AS
80 IF AS = « Y » OR AS = « y » THEN 280 ELSE 90
90 FOR I = 1 TO N
100 FOR. J = 1 TO N
110 PRINT « A (« ; I ; », « ; J ; ») = » ;
120 INPUT A (I, J)
130 PRINT « C(« ; I ; », « ; J ; ») = » ;
140 INPUT C (I, J)
150 F (I, J) = 0
160 NEXT
170 NEXT
180 INPUT « ghi số liệu ra đĩa hay không ? (Y/N) » ; AS
190 IF AS « Y » OR AS = « y » THEN 200 ELSE 370
200 INPUT « vào tên tệp : » , AS
210 OPEN « o »,#1, AS
220 FOR I = 1 TO N
230 FOR J = 1 TO N
240 WRITE#1, A (I, J), C (I, J)
250 NEXT
260 NEXT
270 GOTO 370
280 INPUT « vào tên tệp : », AS
290 OPEN « i »,#1, AS
300 FOR I = 1 TO N
310 FOR J = 1 TO N
320 INPUT #1, A (I, J), C (I, J)
330 AA (I, J) = A (I, J)
340 NEXT
350 NEXT
360 CLOSE #1
```

```
370 PRINT «CHƯƠNC TRÌNH CON» : MASO = 0
380 S = 1
390 FOR I = 1 TO N
400 FOR J = 1 TO N
410 Z (I, J) = ABS (A (I, J)) : PRINT Z (I, J) ;
420 B (I, J) = Z (I, J) : LU (I, J) = Z (I, J)
430 NEXT J
440 PRINT
450 NEXT I
460 FOR I = 1 TO N
470 FÒR J = 1 TO N
480 O (I, J) = 0
490 FOR K = 1 TO N
500 O (I, J) = O (I, J) + B (I, K) * Z (K, J)
510 NEXT K
520 NEXT J
530 NEXT I
540 S = S + 1
550 IF S = N THEN PRINT «LUỒNG ĐÃ ĐƯỢC ĐẦY. CHƯƠNG TRÌNH DỪNG Ở ĐÂY»
GOTO 1040
560 FOR I = 1 TO N
570 FOR J = 1 TO N
580 B (I, J) = O (I, J)
590 LU (I, (S-1) * N + J) = O (I, J)
600 NEXT
610 NEXT
620 IF LU (1, (S-1) * N + N) = O THEN 460 ELSE MASO = 1 : S = S + 1 : PRINT «SO DINH
OF DUƠNG = » ; S
630 DU (1) = 1 : DU (S) = N : PRINT DU (1) ; DU(3)
640 P = S - 1 : W = N
650 FOR K = 2 TO N-1
660 IF LU (1, (P-2) * N + K) * LU (K, W) = O THEN 680 ELSE 670
670 DU (P) = K : GOTO 690
680 NEXT
690 P = P - 1 : W = K
700 IF P>1 THEN 650 ELSE 710
710 FORE E = 1 TO S
720 PRINT « DU («; E ;») = » ; DU (E)
730 NEXT
740 EX (1) = 1E + 37
750 FOR K = 1 TO S-1
760 IF A (DU (K), DU (K+1)) = 1 THEN 770 ELSE 820
770 IF F (DU (K), DU (K+1)) < C (DU) (K), DU (K+1)) THEN 780 ELSE 860
780 IF EX (DU (K)) < C (DU (K), DU (K+1)) - F (DU (K), DU (K+1)) THEN 790 ELSE
800
790 EX (DU(K+1)) = EX (DU(K)) : GOTO 860
800 EX (DU(K+1)) = C(DU(K), DU(K+1)) - F(DU(K), DU(K+1))
810 GOTO 860
820 IF A (DU (K), DU (K+1)) = - 1 THEN 830 ELSE 860
```

```
830 IF F (DU (K) DU (K+1)) > O THEN 840 ELSE 860
840 IF EX (DU (K)) < F (DU (K+1)) THEN EX (DU (K+1)) EX (DU (K) : GOTO 860 ELSE 850
850 EX (DU (K+1)) = F (DU (K), DU (K+1))
860 NEXT K
870 PRINT « GIÁ TRỊ CỦA GIA SỐ EX (N) = » ; EX (N)
880 FOR I = 1 TO S-1
890 IF A (DU (I), DU (I+1))=1 THEN F (DU (I), DU (I+1))=F (DU(I) (DU(I+1))+EX (N): GOTO910
900 F (DU (I), DU (I+1)) = F (DU (I), DU (I+1)) - EX (N)
910 NEXT
920 PRINT « BẢNG PHÂN LUỒNG »
930 U = O
940 FOR I = 1 TO N
950 U = U + F (I, N)
960 FOR J = 1 TO N
970 PRINT F (I, J);
980 IF F (I, J) = C (I, J) THEH 990 ELSE 1000
990 IF F (I, J) > O THEN A (I, J) = O : A (J, I) = O
```

Nhận ngày 29-10-1988

TÀI LIỆU THAM KHẢO

1. **Sachs H.**, Einfuhrung in die Theorie der endlichen Graphen Leipzig 1970.

2. **Berge C.**, Théorie des graphes et ses applications, Paris 1967.

3. **Wesley W. Chu, Leslie J. Holloway, Min—Tsung Lan and Kemal Efe**, « Task Allocation in distributed data Processing » Computer, Now 1980, p. 57-69, Vol. 13, Nr. 11 - ISSN-0018-9162.

4. **Harary. F.**, Proof techniques in Graph Theory. New—york and London 1969.

5. **Roberts F.**, Discrete Matthematical models, 1981. London — Sydney — Toronto — Naw Delhi—Tokyo - Singapore.

6. **Vũ Đình Hòa và Đoàn Phúc**, Bài toán luồng và ứng dụng trong thựa tế. Tạp chí Khoa học tính toán và điều khiển 3/1988.

ABSTRACT

THE PROBLEM OF FLOW AND ITS APPLICATION

The algorithm for finding maximal network flows are used for finding solutions of some another problems and one programm writing in language Basic for the algorithm.