ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1988
Seria:AUTOMATYKA z.94 Nr kol.970

Qin Yuyuan
Wuhan Iron and Steel University
Wuhan, Hubei, China

A TABLEAU STRUCTURE OF DYNAMIC PROGRAMMING

Summary. Dynamic programming is a useful tool to solve multistage decision problems. But there are still several points worth to be improved. One of them may be how to calculate those results for numerical problems in a reasonable way.

In this paper we suggest a tableau structure which can be used to treat such problems of finite type of which the state set at each stage is either finite or infinite in number.

1. Type 1 Finite State Set

The most practical problems of this type can always be converted into solving shortest path problems on weighted multi- n- stage digraphs [1].

Suppose we have a multi-stage digraph G. Its vertex set has an (n+1) partition

$$V^{(0)}V^{(1)} \dots \dots V^{(n)}$$

where
$$V^{(i)} = \left\{ v_t^{(i)} \mid t = 1,2,\dots t_i \right\}$$

$$|V^{(i)}| = t_i, \quad i = 0,1,\dots n.$$

The length of the link $v_r^{(i-1)} v_s^{(i)}$ is denoted by $d(v_r^{(i-1)}, v_s^{(i)})$ or $a_{rs}^{(i)}$. If there is no link from $v_p^{(i-1)}$ to $v_q^{(i)}$, we may imagine that it does havea link with the length $+\infty$. The i-th stage can be written as a modi-matrix /also called revised matrix by T.C.Hu and mini-add matrix by A.Shimbel/, denoted by STAGE$\left(V^{(i-1)}, V^{(i)}\right)$, or STAGE(i):

$$
\mathrm{STAGE}(i) = \begin{array}{c} \\ v_1^{(i-1)} \\ v_2^{(i-1)} \\ \vdots \\ v_h^{(i-1)} \\ \vdots \\ v_{t_{i-1}}^{(i-1)} \end{array}
\begin{array}{c} \begin{array}{ccccccc} v_1^{(i)} & v_2^{(i)} & \cdots & v_k^{(i)} & \cdots & v_{t_i}^{(i)} \end{array} \\
\left[\begin{array}{ccccccc} a_{11}^{(i)} & a_{12}^{(i)} & \cdots & a_k^{(i)} & \cdots & a_{1t_i}^{1} \\ & & & & & \\ & & & & & \\ \cdots & \cdots & & a_{hk}^{(i)} & \cdots & \\ & & & & & \\ \cdots & & \cdots & & & \end{array} \right] \end{array}
$$

If the length of a path is defined in the form of the sum of all lengths on it, the lengths of shortest paths from any state /vertex/ in $V^{(0)}$ to another in $V^{(n)}$ can be found as:

$$\prod_{i=1}^{n} \mathrm{STAGE}(i)$$

Since the associative law for modi-product of modi-matrices is valid, there are several ways for performing the computations. The most straight ways may be the forward process and the backward one. When you need to solve a numerical problem by hand, and once you have decided to take the forward or backward process, the following two tabular forms are recommended, which will help you to save yourself considerable writting effort by organizing the computations in a convenient and compact form.

Forward process tableau for computing $d(V^{(0)}, V^{(n)})$

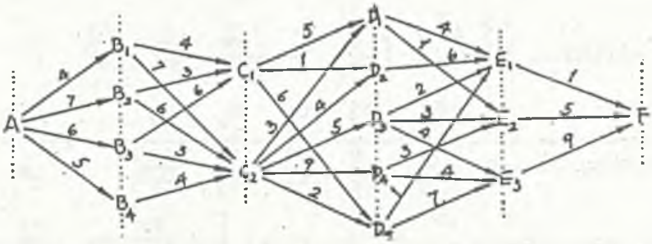
STAGE (1)	STAGE (2)
$\prod_{i=1}^{2}$ STAGE (i)	STAGE (3)
$\prod_{i=1}^{3}$ STAGE (i)	STAGE (4)
⋮	⋮
$\prod_{i=1}^{n-1}$ STAGE (i)	STAGE (n)
$\prod_{i=1}^{n}$ STAGE (i)	$= d(V^{(0)}, V^{(n)})$

Backward process tableau for computing $d\left(v^{(0)}, v^{(n)}\right)$

STAGE(n−1)	STAGE(n)
STAGE(n−2)	$\prod_{i=n-1}^{n}$ STAGE(i)
⋮	⋮
STAGE(1)	$\prod_{i=2}^{n}$ STAGE(i)
$d\left(v^{(0)}, v^{(n)}\right) =$	$\prod_{i=1}^{n}$ STAGE(i)

The terms within the rectangles in the tableaux are the given modi-matrices and the others are intermediate and final results of the computations which you must f i l l in.

Example 1. Find the shortest paths and their length on the following figure.



Solution: Write down the modi-matrices of these stages:

$$\text{STAGE}(A,B) = A\ \begin{matrix} B_1 & B_2 & B_3 & B_4 \\ (4 & 7 & 6 & 5) \end{matrix}, \quad \text{STAGE}(B,C) = \begin{matrix} & C_1 & C_2 \\ B_1 & 4 & 7 \\ B_2 & 3 & 6 \\ B_3 & 6 & 3 \\ B_4 & & 4 \end{matrix}$$

$$\mathrm{STAGE}(C,D) = \begin{array}{c} \\ C_1 \\ C_2 \end{array} \begin{array}{c} \begin{array}{ccccc} D_1 & D_2 & D_3 & D_4 & D_5 \end{array} \\ \begin{bmatrix} 5 & 1 & & 6 & \\ 3 & 4 & 5 & 9 & 2 \end{bmatrix} \end{array}, \quad \mathrm{STAGE}(D,E) = \begin{array}{c} \\ D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \end{array} \begin{array}{c} \begin{array}{ccc} E_1 & E_2 & E_3 \end{array} \\ \begin{bmatrix} 4 & 1 & \\ 6 & & \\ 2 & 3 & 4 \\ & 3 & 4 \\ 1 & & 7 \end{bmatrix} \end{array}$$

and

$$\mathrm{STAGE}(E,F) = \begin{array}{c} \\ E_1 \\ E_2 \\ E_3 \end{array} \begin{array}{c} F \\ \begin{bmatrix} 1 \\ 5 \\ 9 \end{bmatrix} \end{array}$$

We define

$$\mathrm{STAGE}(A,B) \otimes \mathrm{STAGE}(B,C) \equiv \mathrm{STAGE}(A,B,C) \equiv \mathrm{STAGE}(A,C),$$

$$\mathrm{STAGE}(A,B) \otimes \mathrm{STAGE}(B,C) \otimes \mathrm{STAGE}(C,D) \equiv$$
$$\equiv \mathrm{STAGE}(A,B,C,D) \equiv \mathrm{STAGE}(A,D),$$

and so on. We have

$$\mathrm{STAGE}(A,\ B) = A \begin{array}{c} \begin{array}{cc} C_1 & C_2 \end{array} \\ \left[\dfrac{8}{B_1} \quad \dfrac{9}{B_3,\ B_4} \right] \end{array}$$

$$\mathrm{STAGE}(A,\ D) = A \begin{array}{c} \begin{array}{ccccc} D_1 & D_2 & D_3 & D_4 & D_5 \end{array} \\ \left[\dfrac{12}{C_2} \quad \dfrac{9}{C_1} \quad \dfrac{14}{C_2} \quad \dfrac{14}{C_1} \quad \dfrac{11}{C_2} \right] \end{array}$$

We have made two conventions above. The first one is that the only elements omitted are positive infinity. Second, the vertices under a number divided by a short line are those through which the shortest path passes. They are not entries of the modi-matrix, hence do not take part in the computation.

For example, on STAGE(A,C), we can read the path from A to C_2 via B_3 or B_4 is a shortest one among all /four/ possible paths from A to C_2, and the length is 9. Again, on STAGE(A,D), we can read that the shortest path from A to D_4 passes through C_1 which has the length 14. As for the shortest path from A to C_1, we can look at STAGE(A,C) and that it must pass through vertex B_1. Similarly, we have

$$\mathrm{STAGE}(A,E) = A \begin{array}{c} \begin{array}{ccc} E_1 & E_2 & E_3 \end{array} \\ \left[\dfrac{12}{D_5} \quad \dfrac{13}{D_1} \quad \dfrac{18}{D_3, D_4, D_5} \right] \end{array},$$

$$\text{and} \qquad \mathrm{STAGE}(A,F) = A \begin{array}{c} F \\ \left[\dfrac{13}{E_1} \right] \end{array}$$

By using the forward process tableau, we have

$\begin{array}{ccccc} & B_1 & B_2 & B_3 & B_4 \\ A & 4 & 7 & 6 & 5 \end{array}$	$\begin{array}{ccc} & C_1 & C_2 \\ B_1 & 4 & 7 \\ B_2 & 3 & 6 \\ B_3 & 6 & 3 \\ B_4 & & 4 \end{array}$
$\begin{array}{ccc} & C_1 & C_2 \\ A & \dfrac{8}{B_1} & \dfrac{9}{B_3,B_4} \end{array}$	$\begin{array}{cccccc} & D_1 & D_2 & D_3 & D_4 & D_5 \\ C_1 & 5 & 1 & & 6 & \\ C_2 & 3 & 4 & 5 & 9 & 2 \end{array}$
$\begin{array}{cccccc} & D_1 & D_2 & D_3 & D_4 & D_5 \\ A & \dfrac{12}{C_2} & \dfrac{9}{C_1} & \dfrac{14}{C_2} & \dfrac{14}{C_1} & \dfrac{11}{C_2} \end{array}$	$\begin{array}{cccc} & E_1 & E_2 & E_3 \\ D_1 & 4 & 1 & \\ D_2 & 6 & & \\ D_3 & 2 & 3 & 4 \\ D_4 & & 3 & 4 \\ D_5 & 1 & & 7 \end{array}$
$\begin{array}{cccc} & E_1 & E_2 & E_3 \\ A & \dfrac{12}{D_5} & \dfrac{13}{D_1} & \dfrac{18}{D_3,D_4,D_5} \end{array}$	$\begin{array}{cc} & F \\ E_1 & 1 \\ E_2 & 5 \\ E_3 & 9 \end{array}$
$\begin{array}{cc} & F \\ A & \dfrac{13}{E_1} \end{array}$	

Certainly, the result is the same as that obtained above.

Therefore the distance /shortest length/ from A to F is 13 and the shortest paths are found from STAGE(A,F), STAGE(A,E), STAGE(A,D) and STAGE(A,C) successively. We have

$$A \quad \begin{matrix} B_3 \\ B_4 \end{matrix} C_2 \quad D_5 \quad E_1 \quad F$$

Hence we have two shortest paths with the length 13.

2. Type 2 Infinite State Set

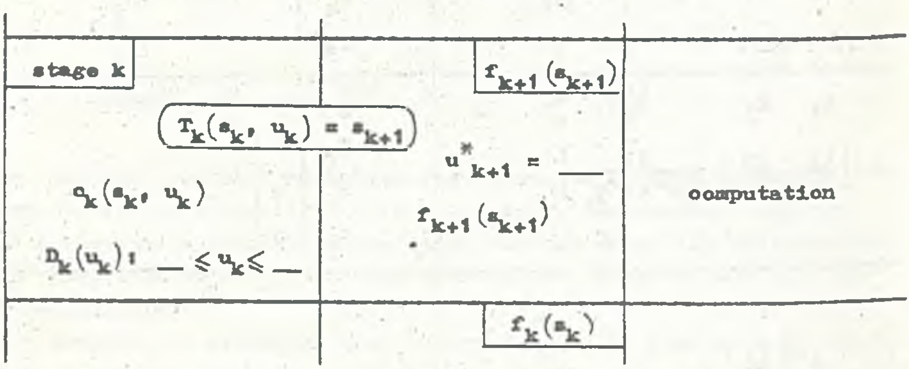
The basic recursive relationship for backward induction is

$$\begin{cases} f_k(s_k) = \underset{u_k \in D_k(u_k)}{\text{opt.}} \left\{ c_k(s_k, u_k) \ , \ f_{k+1}(s_{s+1}) \right\} \\ \qquad\qquad\qquad k = 1, 2, \ldots N \\ f_{N+1}(s_{N+1}) = 0 \end{cases} \qquad /1/$$

where on the k-th stage, s_k is the state variable u_k is the decision variable subject to the restrictions $D_k(u_k)$. The transition function from s_k to s_{k+1} is

$$s_{k+1} = T_k(s_k, u_k) \ . \qquad /2/$$

When the state set is /or, at least, theoretically/ not finite, we cannot use modi-matrices as computational tool. We suggest to solve the problems by /1/ and /2/ on the tableau as following:



Surely, we also have an analogous tableau for forward induction. We shall see the role of the tableau method playing in solving the following numerical example.

Example 2. An agriculture product company that sells a single product would like to consider a six month period inventory problem. At the beginning of each period, k, the company reviews the inventory level, meet the demand requirement in the period, d_k, and then decides how many units to buy from its supplier for next period. The price c_k of the product changes violently from time to time, but the company has an accurate forecast of d_k and c_k, they are

period k	0	1	2	3	4	5	6
demand d_k	0	8	5	3	2	7	4
price c_k	11	18	13	17	20	10	

Suppose the initial and final inventory levels are 2 and 0 respectively. The capacity of the warehouse: H, is 9 units.

The objective of the company is to devise a schedule that minimizes the total buy cost subject to the restriction that all the demand requirements are satisfied on time.

Solution: we take the inventory level at the beginning of stage k as the state variable, s_k, and the amount to buy as the decision variable, u_k. According to the given conditions, we have

(i) transition function: $s_{k+1} = s_k - d_k + u_k$, /3/

$s_7 = 0$,

(ii) constraints: $d_{k+1} \leq s_{k+1} \leq H$, /4/

where $H = 9$.

By /3/, when $k = 6$, we have $u_6 = 0$, and then $s_6 = d_6 = 4$.

By /3/, /4/, we have

$$d_{k+1} + d_k - s_k \leq u_k \leq H + d_k - s_k .$$

But u_k must be non-negative, so we get

$$D_k(u_k): \max(0, d_{k+1} + d_k - s_k) \leq u_k \leq H + d_k - s_k . \quad /5/$$

The recursive relationship for backward induction is

$$f_k(s_k) = \min_{u_k} \left\{ c_k u_k + f_{k+1}(s_{k+1}) \right\} .$$

Then we can solve the numerical example on the following tableau.

Tableau		Computation
stage 5 $d_5 = 7$ $(s_5 - 7 + u_5 = s_6 = 4)$ $10\ u_5$ $11-s_5 \le u_5 \le 16-s_5$	$f_6(s_6)$ stage 6 0	$\min\limits_{u_5} \{10\ u_5 + 0\} = 10(11-s_5)$, $u_5^* = 11 - s_5$
stage 4 $d_4 = 2$ $(s_4-2+u_4 = s_5)$ $20\ u_4$ $9-s_4 \le u_4 \le 11-s_4$	$f_5(s_5)$ $u_5^* = 11-s_5$ $110-10s_5$	$\min\limits_{u_4} \{20u_4+110-10(s_4-2+u_4)\} =$ $= \min \{130+10u_4-10s_4\}$ $= 130+10(9-s_4)-10s_4 = 220-20s_4$ $u_4^* = 9-s_4$
stage 3 $d_3 = 3$ $(s_3-3+u_3 = s_4)$ $17\ u_3$ $\max(0,5-s_3) \le u_3 \le 12-s_3$	$f_4(s_4)$ $u_4^* = 9-s_4$ $220-20s_4$	$\min\limits_{u_3} \{17u_3+220-20(s_3-3+u_3)\} =$ $= \min \{280-3u_3-20s_3\} =$ $= 280-3(12-s_3)-20s_3 = 244-17s_3$ $u_3^* = 12-s_3$
stage 2 $d_2 = 5$ $(s_2-5+u_2 = s_3)$ $13\ u_2$ $\max(0,8-s_2) \le u_2 \le 14-s_2$	$f_3(s_3)$ $u_3 = 12-s_3$ $244-17s_3$	$\min\limits_{u_2} \{13u_2+244-17(s_2-5+u_2)\}$ $= \min \{329-4(14-s_2)-17s_2\} =$ $= 273-13s_2$ $u_2^* = 14-s_2$
stage 1 $d_1 = 8$ $(s_1-8+u_1 = s_2)$ $18\ u_1$ $13-s_1 \le u_1 \le 17-s_1$	$f_2(s_2)$ $u_2 = 14-s_2$ $273-13s_2$	$\min\limits_{u_1} \{18u_1+273-13(s_1-8+u_1)\} =$ $= \min \{377+5u_1-13s_1\} =$ $= 377+5(13-s_1)-13s_1 = 442-18s_1$ $u_1^* = 13-s_1$
stage 0 $d_0 = 0$ $(2 + u_0 = s_1)$ $11\ u_0$ $6 \le u_0 \le 7$	$f_1(s_1)$ $u_1 = 13-s_1$ $442-18s_1$	$\min\limits_{u_0} \{11u_0+442-18(2+u_0)\} =$ $= \min \{406-7u_0\} =$ $= 406 - 7 \times 7 = 357$ $u_0^* = 7$

Tableau		Computation
Answer: The minimun total buy cost: 357	$f_o(s_o)$ $u_o^* = 7$ 357	optimal strategy: 7, 4, 9, 3, 0, 4, 0. and its inventory level sequence: 9, 5, 9, 9, 7, 4.

3. Conclusions

In this paper has been suggested a tableau structure for computations of dynamic programming problems. It is an appendix to the theory presented in [1]. The tableau structure can be used to treat finite or infinite problems.

LITERATURA

[1] Qin Yuyuan: On Jar - Metric Principle A Unified Approach to Solve Optimum Paths Problems on Multistage Directed Graph

Recenzent: Dr.hab.inż.F.Marecki

Wpłynęło do Redakcji do dnia 1988-04-30.

TABELARYCZNA STRUKTURA PROGRAMOWANIA DYNAMICZNEGO

S t r e s z c z e n i e

Artykuł stanowi uzupełnienie do opublikowanej wcześniej na VIKKADPP pracy "ON JAR - METRIC PRINCIPLE" [1]. Przedstawiono w nim tabelaryczną formę obliczeń dla problemu "najkrótszej drogi". Ponadto wskazano możliwość wykorzystania proponowanego podejścia do rozwiązywania innych problemów programowania dynamicznego.

ТАБЛИЧНАЯ СТРУКТУРА ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Р е з ю м е

Статья является приложением к ранее опубликованной на У Конференции АДПП работе " On JAR - METRIX Principle " [I] . Представлена табличная форма расчетов для проблемы " кратчайшего пути " . Показаны возможности использования программного подхода к решению других проблем динамического программирования.