第四章 动态规划

*动态规划是解决多阶段决策过程的最优化问题的一种方法。所谓多阶段决策过程是指这样一类决策过程，由于它的特殊性，我们可以把它按时间分为若干阶段(有时称为“时段”)，而在每一个阶段都需要作出决策，以便使整个过程取得最佳的经济效益。在多阶段决策问题中，各个阶段采取的决策一般说来是与时间有关系的。而且，前一阶段采取的策略如何，不但与该阶段的经济效益有关系，更直接影响以后各阶段的经济效益。可见，它是一个动态的问题，因此，处理的方法称为动态规划方法。
*动态规划也可以用来处理一些本来与时间没有关系的静态模型，这只要在静态模型中人为地引进“时间”因素，分成时段，就可以把它看作多阶段的动态模型，用动态规划方法去处理。

§4.1 最短路径问题与最优化原则

一、问题：

所谓最短路径问题是指给定始点和终点，并知道由始点到终点的各种可能的路径，问题是要找一条由始点到终点的最短路，即长度最短的路。

二．例子：

图 4.1 是表示由始点 A 到终点 E 的路线图。由 A 至 B(B_1, B_2或B_3)是第一阶段；由某$B_i(i = 1,2,3)$至 C(C_1, C_2或C_3)是第二阶段；由某$C_j(j = 1,2,3)$至 D(D_1或D_2)是第三阶段；由某$D_k(k = 1,2)$至 E 是第四阶段。

引进几个符号和概念：

(1) n 表示由某点(例如$A, B_i(i = 1,2,3), C_j(j = 1,2,3), D_k(k = 1,2)$)到终点之间的阶段数。例如：A 到 E 的阶段数$n = 4$。
(2) S 表示在任一阶段所处的位置，称 S 为状态变量(S 可取$A, B_i(i = 1, 2,3), C_j(j = 1,2,3), D_k(k = 1,2), E$)。
(3) $x_n(S)$称为决策变量，它表示当处于状态 S 还有 n 个阶段要走时，下一步所选取的点(例如：当$x_3(B_1) = C_1$时，它表示现在处于点B_1，还有三个阶段要走，下一步选取点C_1，即要走$B_1 \to C_1$的路线)。
(4)$f_n(S)$表示现在处于状态 S，还有 n 个阶段要走，由 S 至终点 E 的最短距离(例如：$f_4(A)$表示现在处于点 A，还有 4 个阶段要走，由 A 到终点 E 的最短距离)。
(5)$d(S, x_n(S))$表示点 S 到下一步所选的点$x_n(S)$的距离(例如：$d(A, B_1)$表示由 A 到B_1的距离，即$d(A, B_1) = 3$)。

分析：

由 A 到 B 的第一阶段有三种走法：

(1)$A \xrightarrow{3} B_1$，若选此走法，最短距离为

$$d(A,B_1)+f_3(B_1)$$

(2) $A \xrightarrow{5} B_2$，若选此走法，最短距离为

$$d(A,B_2)+f_3(B_2)$$

(3) $A \xrightarrow{4} B_3$，若选此走法，最短距离为

$$d(A,B_3)+f_3(B_3)$$

$f_4(A)$应当在上面三种走法中选取距离最小者，也即

$$f_4(A)=\min\begin{Bmatrix} d(A,B_1)+f_3(B_1) \\ d(A,B_2)+f_3(B_2) \\ d(A,B_3)+f_3(B_3) \end{Bmatrix}=\min\begin{Bmatrix} 3+f_3(B_1) \\ 5+f_3(B_2) \\ 4+f_3(B_3) \end{Bmatrix}$$

要求$f_4(A)$，必须先算出$f_3(B_1),f_3(B_2)$及$f_3(B_3)$。

现在算$f_3(B_1)$。有两种走法：

(1)$B_1 \xrightarrow{1} C_1$，若选此走法，最短距离为

$$d(B_1,C_1)+f_2(C_1)$$

(2) $B_1 \xrightarrow{5} C_2$，若选此走法，最短距离为

$$d(B_1,C_2)+f_2(C_2)$$

$f_3(B_1)$应当在上面两种走法中选取距离最小者。

$$f_3(B_1)=\min\begin{Bmatrix} d(B_1,C_1)+f_2(C_1) \\ d(B_1,C_2)+f_2(C_2) \end{Bmatrix}=\min\begin{Bmatrix} 1+f_2(C_1) \\ 5+f_2(C_2) \end{Bmatrix}$$

同理，$f_3(B_2)$和$f_3(B_3)$的算法如下：

$$f_3(B_2)=\min\begin{Bmatrix} d(B_2,C_1)+f_2(C_1) \\ d(B_2,C_2)+f_2(C_2) \\ d(B_2,C_3)+f_2(C_3) \end{Bmatrix}=\min\begin{Bmatrix} 8+f_2(C_1) \\ 4+f_2(C_2) \\ 6+f_2(C_3) \end{Bmatrix}$$

$$f_3(B_3)=\min\begin{Bmatrix} d(B_3,C_1)+f_2(C_1) \\ d(B_3,C_2)+f_2(C_2) \\ d(B_3,C_3)+f_2(C_3) \end{Bmatrix}=\min\begin{Bmatrix} 4+f_2(C_1) \\ 4+f_2(C_2) \\ 2+f_2(C_3) \end{Bmatrix}$$

$$f_2(C_1)=\min\begin{Bmatrix} d(C_1,D_1)+f_1(D_1) \\ d(C_1,D_2)+f_1(D_2) \end{Bmatrix}=\min\begin{Bmatrix} 4+f_1(D_1) \\ 2+f_1(D_2) \end{Bmatrix}$$

$$f_2(C_2)=\min\begin{Bmatrix} d(C_2,D_1)+f_1(D_1) \\ d(C_2,D_2)+f_1(D_2) \end{Bmatrix}=\min\begin{Bmatrix} 6+f_1(D_1) \\ 9+f_1(D_2) \end{Bmatrix}$$

$$f_2(C_3)=\min\begin{Bmatrix} d(C_3,D_1)+f_1(D_1) \\ d(C_3,D_2)+f_1(D_2) \end{Bmatrix}=\min\begin{Bmatrix} 7+f_1(D_1) \\ 5+f_1(D_2) \end{Bmatrix}$$

由于D_1(或D_2)经过一个阶段到终点 E 只有一种走法，因此

$f_1(D_1)=d(D_1,E)=1$

$f_1(D_2)=d(D_2,E)=2$

从后往前倒推：

$n=1$时，

$f_1(D_1)=1$，路径：$D_1 \to E$

$f_1(D_2)=2$，路径：$D_2 \to E$

$n=2$时，

$f_2(C_1)=4$，路径：$C_1 \to D_2 \to E$

$f_2(C_2)=7$，路径：$C_2 \to D_1 \to E$

$f_2(C_3)=7$，路径：$C_3 \to D_2 \to E$

$n=3$时，

$f_3(B_1)=5$，路径：$B_1 \to C_1 \to D_2 \to E$

$f_3(B_2)=11$，路径：$B_2 \to C_2 \to D_1 \to E$

$f_3(B_3)=8$，路径：$B_3 \to C_1 \to D_2 \to E$

$n=4$时，

$f_4(A)=8$，路径：$A \to B_1 \to C_1 \to D_2 \to E$

在例 1 中，我们主要利用了 n 个阶段与$n-1$个阶段的关系式(称为递推关系式或函数方程)：

$$f_n(S)=\min\{d(S,x_n(S))+f_{n-1}(x_n(S))\}$$

而$n=1$时，有 $f_1(S)=d(S,E)$

上述递推关系式可以用“最优化原则”来描述：

一个过程的最优策略具有这样的性质，即无论其初始状态和初始决策如何，其今后诸决策对以第一个决策所形成的状态作为初始状态的过程而言，必须构成最优策略。

当处于状态 S，还有 n 个阶段要走，由 S 至终点 E 的最短距离$f_n(S)$应该是：当第一步已采取决策(即走法)$x_n(S)$后，以后按最优路线走$n-1$个阶段的总距离$\{d(S,x_n(S))+f_{n-1}(x_n(S))\}$中的最小者。

§4.2 动态规划的算法本质

1.自上而下分治策略的复杂性

2.自下而上的动态规划填表法

例 1：以上例为例解释

例 2：世界系列差别赛

1.问题：有两个队 A 和 B 进行一系列比赛，看哪个队先赢得 n 场比赛。

设$P(i,j)$是 A 队需要再赢 i 场比赛(则 A 队赢), B 队需要再赢 j 场比赛(则 B 队赢)，这时 A 队赢的概率。

我们假设 A 和 B 队有相同的竞争力，各有 50%赢某场的概率。

例如：$n=4$，A 队需要再赢$i=2$场，B 队需要再赢$j=3$场，则

$P(2,3)=11/16$。

2.$P(i,j)$的递归式：

$$P(i,j)=\begin{cases}1, & \text{if } i=0 \text{ and } j>0\\ 0, & \text{if } i>0 \text{ and } j=0\\ (P(i-1,j)+P(i,j-1))/2, & \text{if } i>0 \text{ and } j>0\end{cases} \quad (4.1)$$

解以上递归方程，令$i+j=n$，$T(n)$是调用$P(i,j)$的最大次数，由(4.1)式

$$T(1)=c$$

$$T(n)=2T(n-1)+d, \quad c \text{ 和 } d \text{ 为常数}$$

解得 $T(n)\le 2^{n-1}c+(2^{n-1}-1)d$。即$T(n)$为 $O(2^n)=O(2^{i+j})$。

但递归计算$P(i,j)$有重复计算。例如：计算$P(2,3)$需要计算$P(1,3)$和

$P(2,2)$，而计算$P(1,3)$和 $P(2,2)$都需要计算$P(1,2)$。
3.列表计算(见图 4.2):

*解释计算规则
4.算法：

```
FUNCTION Odds (i, j : integer) : real;
VAR s, k : integer;
BEGIN
FOR s := 1 TO i+j DO
    BEGIN
   P[0, s] := 1.0;
   P[s, 0] := 0.0;
   FOR k := 1 TO s-1 DO
      P[k, s-k] := (P[k-1, s-k] + P[k, s-k-1])/2.0;
    END;
 RETURN (P[i, j]);
END;
```

5.算法分析：
$4-5$ 行：$O(s)$
$2-3$ 行：$O(1)$
总时间：$O(\sum_{s=1}^{n} s) = O(n^2)$。

§4.3 动态规划举例—Halin 图中的 TSP 问题

一．Halin 图的定义

1.Halin 图的定义：
在平面上嵌入一棵树 T，T 的每个内部顶点的度数至少为 3，并且至少有一个内部顶点。作一个圈 C 连接 T 的所有叶顶点。T 的所有叶顶点组成 C 上的所有顶点。这样得到的平面图 $H = T \cup C$称为 Halin 图。树 T 称为 Halin 图的特征树，圈 C 称为 Halin 图的伴随圈。(见图 4.3)

2.轮
设$H = T \cup C$是一个 Halin 图，其中树 T 是一颗星，即单个结点 v 连接到其它互不相邻的 n 个顶点，那么 H 就称为轮。一个轮是最简单的 Halin 图，记为W_p。(见图 4.4)

3. 扇
假设 T 至少有两个非叶子顶点，w 是树 T 中任意一个非叶子顶点，而且 w 仅与 T 中一个非叶子顶点相邻。不妨令 w 相邻的叶子顶点集合为 C(w)，注意到这些顶点在圈 C 上是一段连续的顶点。我们将导出子图$F = H[\{w\} \cup C(w)]$称为一个扇，w 为扇的中心。(见图 4.3)

4.扇收缩

假设 F 是 Halin 图$H = T \cup C$的一个扇，则把图 H 由收缩扇 F 成为一个伪点v_F，所得到的图记为$H\times F, H\times F$表示把 H 中的扇收缩成一点v_F后的图。

$V(H\times F) = \{v_F\} \cup (V(H)\backslash V(F))$，$E(H\times F)$被定义如下：

(1)如果一条边的两个端点都在 F 中，那么删除该边；

(2)如果一条边的两个端点都在$H - F$中，那么这条边保持不变；

(3)如果一条边的一个端点在$H - F$中，另一个端点在 F 中，那么现在这条边在$H - F$中的端点不变，另一端点为v_F。

*注意：

(1)任一 Halin 图 H 如果包含一个扇 F，那么 H 收缩扇 F 后得到的图$H' = H\times F$仍是一个 Halin 图。(见图 4.3)

(2)设$H = T \cup C$是一个 Halin 图，反复收缩 H 的扇，最后必然得到一个轮。

(3)存在求 Halin 图的特征树和伴随圈的 O(n)时间的算法，该算法也是判定一个图是否 Halin 图的算法。

二．求赋权 Halin 图中的旅行售货员问题(TSP)的算法

1.TSP 问题：设 G 是一个图，对任意边$e \in E(G), e$上有一个正的权$w(e)$ $\in R^+$，求 G 中一个哈密顿圈 C，使得 C 上的权$w(C) = \sum_{e\in E(C)} w(e)$在 G 的所有哈密顿圈中最小。

*TSP 问题对一般图而言是 NP-难的，但在 Halin 图上，却有 O(n)时间的算法求解。

2.在 Halin 图上求解 TSP 问题的算法：

设 H 是一个赋权 Halin 图。

(2.1)若 H 是一个轮(如图 4.5)

令$w(C') = \sum_{i=0}^{n-1} w(u_i u_{i+1}) + w(u_n u_0)$

则最短哈密顿圈：

$$C = \min_{0\le i\le n}\{w(C') - w(u_i u_{i+1}) + w(u_i V) + w(V u_{i+1})\}$$

(2.2)若 H 不是轮，则 H 包含一个扇 F(见图 4.6)

令

$$\bar{C}_{jk} = \sum_{i=1}^{r-1} w(u_i u_{i+1}) + w(u_r w)$$

$$\bar{C}_{kl} = \sum_{i=1}^{r-1} w(u_i u_{i+1}) + w(u_1 w)$$

$$令\ w(P) = \sum_{i=1}^{r-1} w(u_i u_{i+1})$$

$$\bar{C}_{jl} = \min_{1\le i\le r-1}\{w(P) - w(u_i u_{i+1}) + w(u_i w) + w(w u_{i+1})\}$$

收缩扇 F，得$H' = H\times F$。给 H′的各边一个新的赋权c_h'如下，其中c_j表示原来 H 的边 j 上的权值，c_h'为 H′的边 h 上的新权值。

$$c'_h = \begin{cases} c_h, & \text{for } h \in E(H')\backslash\{j,k,l\} \\ c_j + \frac{1}{2}\left(\bar{C}_{jl} + \bar{C}_{jk} - \bar{C}_{kl}\right), & \text{if } h = j \\ c_k + \frac{1}{2}\left(\bar{C}_{kl} + \bar{C}_{jk} - \bar{C}_{jl}\right), & \text{if } h = k \\ c_l + \frac{1}{2}\left(\bar{C}_{jl} + \bar{C}_{kl} - \bar{C}_{jk}\right), & \text{if } h = l \end{cases}$$

那么，H’中最短哈密顿圈的权与 H 中最短哈密顿圈的权相等。

*举例说明：

算法：

1.输入赋权 Halin 图$H = T \cup C$；

2.以 T 的某个内部顶点 v 为根，后序遍历 T;

3.每当遍历到一个扇的中心 w 时，与 w 相邻的叶子$u_1, u_2, \cdots, u_r$已遍历完。令$F = H[\{w\} \cup \{u_1, u_2, \cdots, u_r\}]$为扇，构造$H \times F$，按(2.2)方法给 H’的边赋权。若 H’是轮则执行步骤 4；否则继续后序遍历 T, 执行步骤 3；

4.若 H’是轮，按照(2.1)的方法，求 H’的最短哈密顿圈 C’；

5.按上述过程的逆序过程，逐层将 H’中的最短哈密顿圈 C’恢复成 H 中的最短哈密顿圈 C”，最后求得原图 H 中的最短哈密顿圈。