

ZY1806418 赵嘉 算法作业
Assignment 1

一、用动态规划解工厂生产问题

每月需求量如表:	时期(月)	需要量(产品单位)
	1	2
总需求 = $2+3+2+4=11$	2	3
	3	2
	4	4

对每个月来讲,若当月生产 x 单位,则生产成本 $P(x) = \begin{cases} 0 & x=0 \\ 3+x & x>0 \end{cases}$ (不生产无成本)
(固定成本 3 加上每单位 1 成本)

对每个月来讲,若上月结束时库存为 y ,则将其存储一个单位产生库存费用为 $S(y) = 0.5y$ (每单位每月 0.5 费用)

设前 i 个月的总需求为 $D(i)$, 可得: $D(1)=2$ $D(2)=5$ $D(3)=7$ $D(4)=11$, $D(0)=0$

设状态 $T(i, j)$ 表示第 i 个月 ($1 \leq i \leq 4$) 结束时, 产品的总生产量为 j 时, 所能花费的最低总成本

显然到第 4 月结束为止的总生产量 j 必须不小于前 4 个月的总需求 $D(i)$, 并且不大于所有月份的总需求 $D(4)=11$.

即: $D(i) \leq j \leq D(4)=11$

则有状态转移方程:

$$\begin{cases} T(1, j) = P(j) & D(1) \leq j \leq D(4) \\ T(i, j) = \min \{ T(i-1, j') + S(j' - D(i-1)) + P(j - j') \}, & i > 1, D(i-1) \leq j' \leq j, D(i) \leq j \leq D(4) \end{cases}$$

解释:

到第 i 个月的总花费可以看作是到上个月的总花费加上这个月的花费, 假设到上个月总生产量为 j' , 则到上个月的总花费就是 $T(i-1, j')$, 而这个月的花费有两部分, 一是维持上个月的库存所产生的库存费用, 上个月留下的库存为到上个月为止的总生产量减去到上个月为止的总需要量, 即 $j' - D(i-1)$, 则库存费用为 $S(j' - D(i-1))$, 二是新生产的产品所产生的生产费用, 假设到本月为止总生产量达到 j , 则在这个月新生产了 $j - j'$ 个, 生产费为 $P(j - j')$ 放到第 i 个月总花费就是这三项的和: $T(i-1, j') + S(j' - D(i-1)) + P(j - j')$

其中的 j' 是不固定的, 我们求的就是对于所有可能的 j' , 总花费所能取到的最小值, 故取 \min

至于第一个月, 即 $i=1$ 时, 由于是最开始的月份所以没有 $T(i-1, j')$, 由于一开始没有库存也就没有 $S(j')$ 项, 故只算 $P(j)$

以上即为状态转移方程的推理过程

计算:

$$i=1: T(1, 2) = 5 \quad T(1, 3) = 6 \quad T(1, 4) = 7 \quad T(1, 5) = 8 \quad T(1, 6) = 9 \quad T(1, 7) = 10 \\ T(1, 8) = 11 \quad T(1, 9) = 12 \quad T(1, 10) = 13 \quad T(1, 11) = 14$$

$$i=2: T(2, 5) = \min \{ T(1, 2) + S(0) + P(3), T(1, 3) + S(1) + P(2), T(1, 4) + S(2) + P(1), T(1, 5) + S(3) + P(0) \} \\ = \min \{ 11, 11.5, 12, 9.5 \} = 9.5 \text{ (对应上月总产量为 5)}$$

$$T(2, 6) = \min \{ T(1, 2) + S(0) + P(4), T(1, 3) + S(1) + P(3), T(1, 4) + S(2) + P(2), T(1, 5) + S(3) + P(1), T(1, 6) + S(4) + P(0) \} \\ = \min \{ 12, 12.5, 13, 13.5, 11 \} = 11 \text{ (对应上月总产量为 6)}$$

$$T(2, 7) = \min \{ T(1, 2) + S(0) + P(5), T(1, 3) + S(1) + P(4), T(1, 4) + S(2) + P(3), T(1, 5) + S(3) + P(2), \\ T(1, 6) + S(4) + P(1), T(1, 7) + S(5) + P(0) \} \\ = \min \{ 13, 13.5, 14, 14.5, 15, 12.5 \} = 12.5 \text{ (对应上月总产量 7)}$$

$$\begin{aligned}
T(2,8) &= \min \{ T(1,2) + S(0) + P(6), T(1,3) + S(1) + P(5), T(1,4) + S(2) + P(4), T(1,5) + S(3) + P(3), \\
&\quad T(1,6) + S(4) + P(2), T(1,7) + S(5) + P(1), T(1,8) + S(6) + P(0) \} \\
&= \min \{ 14, 14.5, 15, 15.5, 16, 16.5, 17 \} = 14 \text{ (对应上月总生产 8)} \\
T(2,9) &= \min \{ T(1,2) + S(0) + P(7), T(1,3) + S(1) + P(6), T(1,4) + S(2) + P(5), T(1,5) + S(3) + P(4), T(1,6) + S(4) + P(3), \\
&\quad T(1,7) + S(5) + P(2), T(1,8) + S(6) + P(1), T(1,9) + S(7) + P(0) \} \\
&= \min \{ 15, 15.5, 16, 16.5, 17, 17.5, 18, 18.5 \} = 15 \text{ (对应到上月总生产 2)} \\
T(2,10) &= \min \{ T(1,2) + S(0) + P(8), T(1,3) + S(1) + P(7), T(1,4) + S(2) + P(6), T(1,5) + S(3) + P(5), T(1,6) + S(4) + P(4), \\
&\quad T(1,7) + S(5) + P(3), T(1,8) + S(6) + P(2), T(1,9) + S(7) + P(1), T(1,10) + S(8) + P(0) \} \\
&= \min \{ 16, 16.5, 17, 17.5, 18, 18.5, 19, 19.5, 17 \} = 16 \text{ (对应到上月总生产 2)} \\
T(2,11) &= \min \{ T(1,2) + S(0) + P(9), T(1,3) + S(1) + P(8), T(1,4) + S(2) + P(7), T(1,5) + S(3) + P(6), T(1,6) + S(4) + P(5), \\
&\quad T(1,7) + S(5) + P(4), T(1,8) + S(6) + P(3), T(1,9) + S(7) + P(2), T(1,10) + S(8) + P(1), T(1,11) + S(9) + P(0) \} \\
&= \min \{ 17, 17.5, 18, 18.5, 19, 19.5, 20, 20.5, 21, 18.5 \} = 17 \text{ (对应到上月总生产 2)}
\end{aligned}$$

$$\begin{aligned}
T(3,7) &= \min \{ T(2,5) + S(0) + P(2), T(2,6) + S(1) + P(1), T(2,7) + S(2) + P(0) \} \\
&= \min \{ 14.5, 15.5, 13.5 \} = 13.5 \text{ (对应到上月总生产 7)} \\
T(3,8) &= \min \{ T(2,5) + S(0) + P(3), T(2,6) + S(1) + P(2), T(2,7) + S(2) + P(1), T(2,8) + S(3) + P(0) \} \\
&= \min \{ 15.5, 16.5, 17.5, 15.5 \} = 15.5 \text{ (对应到上月总生产 5)} \\
T(3,9) &= \min \{ T(2,5) + S(0) + P(4), T(2,6) + S(1) + P(3), T(2,7) + S(2) + P(2), T(2,8) + S(3) + P(1), T(2,9) + S(4) + P(0) \} \\
&= \min \{ 16.5, 17.5, 18.5, 19.5, 17 \} = 16.5 \text{ (对应到上月总生产 5)} \\
T(3,10) &= \min \{ T(2,5) + S(0) + P(5), T(2,6) + S(0) + P(4), T(2,7) + S(2) + P(3), T(2,8) + S(3) + P(2), \\
&\quad T(2,9) + S(4) + P(1), T(2,10) + S(5) + P(0) \} \\
&= \min \{ 17.5, 18.5, 19.5, 20.5, 21, 18.5 \} = 17.5 \text{ (对应到上月总生产 5)} \\
T(3,11) &= \min \{ T(2,5) + S(0) + P(6), T(2,6) + S(1) + P(5), T(2,7) + S(2) + P(4), T(2,8) + S(3) + P(3), \\
&\quad T(2,9) + S(4) + P(2), T(2,10) + S(5) + P(1), T(2,11) + S(6) + P(0) \} \\
&= \min \{ 18.5, 19.5, 20.5, 21.5, 22, 22.5, 20 \} = 18.5 \text{ (对应到上月总生产 5)}
\end{aligned}$$

$$\begin{aligned}
T(4,11) &= \min \{ T(3,7) + S(0) + P(4), T(3,8) + S(1) + P(3), T(3,9) + S(2) + P(2), T(3,10) + S(3) + P(1), T(3,11) + S(4) + P(0) \} \\
&= \min \{ 20.5, 22, 22.5, 23, 20.5 \} = 20.5 \text{ (对应到上月总生产 11)}
\end{aligned}$$

即得最低总成本为 20.5(千元)，根据推导时的“上月总产量记录”倒推出每月总产量为：
 第一月：5 第二月：0 第三月：6 第四月：0

二、用动态规划求解TSP问题

已知距离矩阵：

$$D = \begin{matrix} V_1 & 0 & 10 & 20 & 30 & 40 & 50 \\ V_2 & 12 & 0 & 18 & 30 & 25 & 21 \\ V_3 & 23 & 19 & 0 & 5 & 10 & 15 \\ V_4 & 34 & 32 & 4 & 0 & 8 & 16 \\ V_5 & 45 & 27 & 11 & 10 & 0 & 18 \\ V_6 & 56 & 22 & 16 & 20 & 12 & 0 \end{matrix}$$

从 V_1 出发，经过 V_2, V_3, V_4, V_5, V_6 且只经过一次，最后返回 V_1 ，问如何选择路线，使总行程最短。
设状态变量为 $T(I, V_j)$ ，其中 I 是一个集合，是从 V_1 开始经过的其它节点的集合， V_j 代表推销员当前的位置是 V_j 。
 $T(I, V_j)$ 的值就代表了经过集合 I 中的节点后最终落脚于节点 V_j 时所花费的行程的最小值。

举例说明：比如 $T([V_2, V_3, V_5], V_3) = 50$ 就代表推销员从 V_1 出发经过了 V_2, V_3, V_5 之后（并不知道具体的经过顺序，只知道经过了），现在位于 V_3 位置，那么推销员所花费行程最少是 50。

我们要求的结果其实就是 $T([V_1, V_2, V_3, V_4, V_5, V_6], V_1)$

设从 V_k 到 V_j 的距离为 d_{kj} ，设 $I - V_j$ 代表从集合 I 中去除掉节点 V_j 剩下的集合
有状态转移方程：

$$\left\{ \begin{array}{l} \text{当 } I \text{ 中只有一个节点时: } T([V_j], V_j) = d_{jj} \\ \text{当 } I \text{ 中有两个以上节点时: } T(I, V_j) = \min \{ T(I - V_j, V_k) + d_{kj} \}, \text{ 其中 } V_k \text{ 是属于集合 } (I - V_j) \text{ 的任一节点} \end{array} \right.$$

解释：

当 I 中只有一个节点 V_j 时，说明推销员只从 V_1 出发走了一步，到了 V_j ，因此行程就是从 V_1 到 V_j 的距离 d_{jj}

当 I 中有多个节点时，说明推销员走了很多步后现在落脚于 V_j ，既然现在落脚于 V_j ，那么之前肯定没有走过 V_j ，故上一步的经过节点集合里肯定没有 V_j ，将其排除，则上一步的经过节点集合为 $I - V_j$ ，假设上一步的落脚点为 V_k ，那么 V_k 是属于集合 $I - V_j$ 的任意一个节点，则从上一状态 $T(I - V_j, V_k)$ 到这一状态 $T(I, V_j)$ 其实就是增加了从 V_k 到 V_j 的一段路，因此到这一状态的总行程为： $T(I - V_j, V_k) + d_{kj}$

其中 V_k 是在集合 $I - V_j$ 中任选的，我们求的就是对于所有可能的 V_k ，总行程能取到的最小值，故取 \min

未完，接下页

伪代码：

算法 TSP(D, n)

//用动态规划法计算TSP问题

//输入：距离矩阵D，节点的总个数n

//输出：从节点V₁开始经过且只经过一次其它n-1个节点后返回V₁所能花费的最短行程

for j <= 2 to n:

T[E[j], j] <- D[1, j]

for x <= 2 to (n-1): //节点集合I中的节点数从2到n-1遍历

C = getCombination(x, n-1) //选取从n-1个节点中选x个的所有可能组合，共C_{n-1}^x个节点集合

//C中为由节点构成的集合的集合，例如：{[2,3][2,5],[4,6]}

for I in C:

for j in I: //选取V_j, V_j是属于经过节点集合I的，由此可表示状态T(I, j)

E <- I.sub(E[j]) //E为从集合I中除去节点V_j剩下的集合

T[I, j] <- +∞ //先将T(I, V_j)赋为+∞，之后通过与其它情况比较求出最小值

for k in E: //V_k从集合E中选取

T[I, j] = min(T[I, j], T[E, k] + D[k, j]) //求得T[I, j]

End for

End for

End for

End for

//到此为止计算了经过除V₁外所有节点后落脚于除V₁外其它任一节点时所能得到的最小行程

//最终只需要再计算最后一步回到V₁时的最小行程

I <- [1, 2, ..., n]

//I赋为包含所有节点的集合

E <- I.sub(I[1])

//E为包含除V₁外所有节点的集合

T[I, 1] <- +∞

for k in E:

T[I, 1] = min(T[I, 1], T[E, k] + D[k, 1])

End for

return T[I, 1]

算法复杂性分析：

I中的节点数是从1到n遍历的

其中，当I中节点数为x时，集合I有C_{n-1}^x种可能，V_j有n种可能，V_k有n-1种可能

故算法复杂度为 $\sum_{x=1}^{n-1} (C_{n-1}^x \cdot x \cdot (n-1)) < \sum_{x=1}^{n-1} (n^x \cdot n \cdot (n-1)) < n^2 \cdot \frac{n}{n-1} (C_n^n = n^2 - 2^n)$, 即 $O(n) = n^2 \cdot 2^n$