SCHOOL OF SOFTWARE, TSINGHUA UNIVERSITY

dp

孟骋

July 11, 2017

使用dp[i][j]表示已经询问了子树的所有关键节点,人车的一个状态,其中:

- j==0:人去,不管人是否回来
- j==1:人去,人一定要回来
- j==2:人车都去,人车都要回来
- j==3:人车都去,人一定要回来,车不管
- j==4:人车都去,不管人车最后是否回来

这里显然有dp[i][1]>=dp[i][0],dp[i][2]>=dp[i][3]>=dp[i][4]
假设某个结点为fa,其某一子节点为son,w1(fa,son)表示步行的代价,w2(fa,son)表示开车的代价,则有:

$$dp[fa][1] = \sum \left(dp[son][1] + 2 * w1(fa, son)\right)$$

即遍历所有的子节点(w1(fa,son)),然后遍历子节点的所有子节点,并且回到子节点(dp[son][1]),然后从子节点回来(w1(fa,son))。注意,这是一棵树,不存在环。

$$dp[fa][2] = \sum \left(\min(2 * w1(fa, son) + dp[son][1], dp[son][2] + 2 * w2(fa, son))\right)$$

分为两种情况,一种是人步行去,然后步行回来,第二种是人车都去,人车都回来。

$$\begin{aligned} temp &= \min(dp[son][0] - dp[son][1] - w1(fa, son)) \\ temp &= \min(temp, 0) \\ dp[fa][0] &= dp[fa][1] + temp \end{aligned}$$

分两种情况:

- 人不回来,一个边选择为dp[son][0]+w1(fa,son),其它的都选择dp[son][1]+2*w1(fa,son)
- 人回来, 所有边都为dp[son][1]+2*w1(fa,son)=dp[fa][1]

$$\begin{aligned} t2 &= \min(dp[son][1] + 2 * w1(fa, son), dp[son][2] + 2 * w2(fa, son)) = dp[fa][2] \\ t3 &= 0 \\ t3 &= \min(t3, w1(fa, son) + w2(fa, son) + dp[son][3] - t2) \\ dp[fa][3] &= dp[fa][2] + t3 \end{aligned}$$

dp[fa][3]的决策方式:

- 选择一个子节点开车去,然后步行回来(w2(fa,son)),其它的选择t2.
- 都选择t2

等价于:

$$dp[fa][3]=\min(dp[fa][2],dp[fa][2]-t2+w1(fa,son)+w2(fa,son)+dp[son][3])$$

最后是dp[fa][4],从走的最后一棵子树考虑:

- 走最后一棵子树时,还有车:最后一棵子树的选择是 min()
-