四边形不等式优化

在 2D1D 动态规划中的应用

> aDbD 动态规划指状态是 n^a 的，转移是 n^b 的 dp.

对于形如：

$$f_{l,r}=\min_{k=l}^{r-1}\{f_{l,k}+f_{k+1,r}\}+w(l,r)$$

的转移方程，直接求解即是 $O(n^3)$ 的。但是倘若 $w(l,r)$ 满足某些性质，使得决策具有**单调性**，我们即可进行优化。具体地，若 w 满足：

- 对区间包含具有单调性：

$$l\leqslant l'\leqslant r'\leqslant r\implies w(l',r')\leqslant w(l,r)$$

- 四边形不等式（交叉小于包含）：

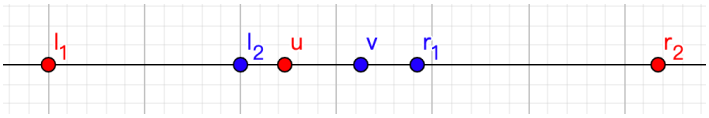
$$l_1\leqslant l_2\leqslant r_1\leqslant r_2\implies w(l_1,r_1)+w(l_2,r_2)\leqslant w(l_1,r_2)+w(l_2,r_1)$$

则：

- **Lemma**：若 w 满足以上两个条件，则 f 满足四边形不等式。

  证：对区间长度采用数学归纳法。边界是平凡的，现设 $g(k,l,r)=f_{l,k}+f_{k+1,r}+w(l,r)$，$u=\arg\min_{k=l_1}^{r_2-1}g(k,l_1,r_2)$，$v=\arg\min_{k=l_2}^{r_1-1}g(k,l_2,r_1)$，即 u,v 分别是 $[l_1,r_2),[l_2,r_1)$ 的最优决策点。分类讨论：

  - 若 $u\leqslant v$：

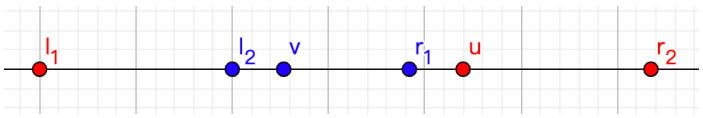
    

    则 $l_1\leqslant u<r_1$ 且 $l_2\leqslant v<r_2$，于是：$f_{l_1,r_1}\leqslant g(u,l_1,r_1)$, $f_{l_2,r_2}\leqslant g(v,l_2,r_2)$.

    又 $u+1\leqslant v+1\leqslant r_1\leqslant r_2$，由归纳假设，$f_{u+1,r_1}+f_{v+1,r_2}\leqslant f_{u+1,r_2}+f_{v+1,r_1}$，于是：

$$\begin{aligned}f_{l_1,r_1}+f_{l_2,r_2}&\leqslant g(u,l_1,r_1)+g(v,l_2,r_2)\\&=f_{l_1,u}+f_{u+1,r_1}+w(l_1,r_1)+f_{l_2,v}+f_{v+1,r_2}+w(l_2,r_2)\\&\leqslant f_{l_1,u}+f_{l_2,v}+f_{u+1,r_2}+f_{v+1,r_1}+w(l_1,r_2)+w(l_2,r_1)\\&=g(u,l_1,r_2)+g(v,l_2,r_1)\\&=f_{l_1,r_2}+f_{l_2,r_1}\end{aligned}$$

  - 若 $u>v$：

    则 $l_2\leqslant u<r_2$ 且 $l_1\leqslant v<r_1$，于是 $f_{l_2,r_2}\leqslant g(u,l_2,r_2)$, $f_{l_1,r_1}\leqslant g(v,l_1,r_1)$.

    又 $l_1\leqslant l_2\leqslant v<u$，由归纳假设，$f_{l_1,v}+f_{l_2,u}\leqslant f_{l_1,u}+f_{l_2,v}$，于是：

$$\begin{aligned}f_{l_1,r_1}+f_{l_2,r_2}&\leqslant g(v,l_1,r_1)+g(u,l_2,r_2)\\&=f_{l_1,v}+f_{v+1,r_1}+w(l_1,r_1)+f_{l_2,u}+f_{u+1,r_2}+w(l_2,r_2)\\&\leqslant f_{l_1,u}+f_{l_2,v}+f_{v+1,r_1}+f_{u+1,r_2}+w(l_1,r_1)+w(l_2,r_2)\\&=g(u,l_1,r_2)+g(v,l_2,r_1)\\&=f_{l_1,r_2}+f_{l_2,r_1}\end{aligned}$$

  证毕。

- **Theorem（决策单调性）**：若 f 满足四边形不等式，记 $m(l,r)=\arg\min_{k=l}^{r-1}g(k,l,r)$，表示最优决策点，则有：

$$m(l,r-1)\leqslant m(l,r)\leqslant m(l+1,r)$$

  证：反证法。为方便，记 $i=m(l,r),j=m(l,r-1),k=m(l+1,r)$.

  - 假若 $j>i$，则：$i+1\leqslant j+1\leqslant r-1\leqslant r$，则根据四边形不等式，有：$f_{i+1,r-1}+f_{j+1,r}\leqslant f_{i+1,r}+f_{j+1,r-1}$；

    又因为 i 是 $f_{l,r}$ 的最优决策点，所以 $f_{l,i}+f_{i+1,r}\leqslant f_{l,j}+f_{j+1,r}$.

    两式相加得到：

$$f_{l,i}+f_{i+1,r-1}\leqslant f_{l,j}+f_{j+1,r-1}$$

这与 j 是 $f_{l,r-1}$ 的最优决策点矛盾。

- 假若 $k<i$，则：$l\leqslant l+1\leqslant k\leqslant i$，则根据四边形不等式，有：$f_{l,k}+f_{l+1,i}\leqslant f_{l,i}+f_{l+1,k}$；

  又因为 i 是 $f_{l,r}$ 的最优决策点，所以 $f_{l,i}+f_{i+1,r}\leqslant f_{l,k}+f_{k+1,r}$.

  两式相加得到：

  $$f_{l+1,i}+f_{i+1,r}\leqslant f_{l+1,k}+f_{k+1,r}$$

  这与 k 是 $f_{l+1,r}$ 的最优决策点矛盾。

证毕。

根据上述定理，只要 w 满足区间包含单调性和四边形不等式，那么最优决策点位置具有单调性。所以我们在枚举 $f_{l,r}$ 的决策点位置时，只需从 $m(l,r-1)$ 枚举到 $m(l+1,r)$ 即可。如此，我们决策点的总枚举量为：

$$\sum_{1\leqslant l<r\leqslant n}m(l+1,r)-m(l,r-1)=\sum_{i=1}^{n}m(i,n)-\sum_{i=1}^{n}m(1,i)\leqslant n^2$$

即算法的时间复杂度下降至 $O(n^2)$.

Code

以合并石子为例：

```
for(int i = 1; i <= n; i++)
    dp[i][i] = 0, p[i][i] = i;
for(int l = 2; l <= n; l++){
    for(int i = 1; i <= n - l + 1; i++){
        int j = i + l - 1;
        dp[i][j] = INF;
        for(int k = p[i][j-1]; k <= min(j-1, p[i+1][j]); k++){
            if(dp[i][j] > dp[i][k] + dp[k+1][j] + s[j] - s[i-1]){
                dp[i][j] = dp[i][k] + dp[k+1][j] + s[j] - s[i-1];
                p[i][j] = k;
            }
        }
    }
}
```

在 1D1D 动态规划中的应用

对于形如：

$$f_r=\min_{l=1}^{r-1}\{f_l+w(l,r)\}$$

的转移方程，倘若 $w(l,r)$ 满足四边形不等式，则决策具有单调性。

Theorem（决策单调性）：若 w 满足四边形不等式，设 $k_r=\min\{l\mid f_l+w(l,r)\}$，即向 f_r 转移时的最优决策点，那么：

$$r_1\leqslant r_2\implies k_{r_1}\leqslant k_{r_2}$$

证：反证法。假若 $k_{r_1}>k_{r_2}$，则 $k_{r_2}<k_{r_1}<r_1\leqslant r_2$，根据四边形不等式，有：$w(k_{r_2},r_1)+w(k_{r_1},r_2)\leqslant w(k_{r_2},r_2)+w(k_{r_1},r_1)$；

又因为 k_{r_2} 是向 f_{r_2} 转移时的最优决策点，故 $f_{k_{r_2}}+w(k_{r_2},r_2)\leqslant f_{k_{r_1}}+w(k_{r_1},r_2)$.

两式相加得到：

$$f_{k_{r_2}}+w(k_{r_2},r_1)\leqslant f_{k_{r_1}}+w(k_{r_1},r_1)$$

这与 k_{r_1} 是向 f_{r_1} 转移时的最优决策点矛盾，故 $k_{r_1}\leqslant k_{r_2}$. 证毕。

与 2D1D dp 不同，这里决策单调性只给出了 l 的下界而没有上界，直接实现仍然是 $O(n^2)$ 的。此时我们可以使用单调栈+二分进行优化：

考虑决策序列，即一个长为 n、第 i 项表示 f_i 的最优决策点位置的序列，我们在 dp 过程中不断更新维护之。由于决策单调性，决策序列始终保持单调不减，因此每当我们得到一个新的 dp 值时，可以在决策序列中二分出第一个变得更优的地方，然后更新它和后面所有的最优决策点。实现时，采用单调栈记录三元组：（决策，左端点，右端点）即可。

Code

以「NOI2009诗人小G」为例：

```
LD dp[N];
int decision[N]; // position of best decision
deque<pair<int, pii>> deq;
inline LD calc(int l, int r){
    // calculate fr decided on l
    if(l >= r)  return INF;
    return dp[l] + fpow(abs(s[r] - s[l] - 1 - L), P);
}
void DP(){
    while(!deq.empty()) deq.pop_back();
    for(int i = 1; i <= n; i++) dp[i] = INF, decision[i] = 0;
    deq.push_back(mp(0, mp(0, n)));

    for(int i = 1; i <= n; i++){
        while(!deq.empty() && deq.front().second.second < i)    deq.pop_front();
        dp[i] = calc(deq.front().first, i);
        decision[i] = deq.front().first;
        while(!deq.empty() &&
            calc(i, deq.back().second.first) < calc(deq.back().first, deq.back().second.first))
            deq.pop_back();
        int l = deq.back().second.first + 1, r = n + 1;
        while(l < r){
            int mid = (l + r) >> 1;
            if(calc(i, mid) < calc(deq.back().first, mid))  r = mid;
            else    l = mid + 1;
        }
        if(l > n)   continue;
        deq.back().second.second = l - 1;
        deq.push_back(mp(i, mp(l, n)));
    }
}
```

关于四边形不等式的性质

性质1：若 w_1, w_2 都满足四边形不等式（或区间包含单调性），则对于任意 $c_1, c_2 > 0$，$c_1 w_1 + c_2 w_2$ 也满足四边形不等式（或区间包含单调性）；

性质2：若 w 能写作：$w(l,r) = f(r) - g(l)$ 的形式，则 w 满足四边形不等式。特别地，若 f, g 都是单调增加的，则 w 还满足区间包含单调性；

性质3：设 h 是一个单调增加的下凸函数，若函数 w 满足四边形不等式和区间包含单调性，则复合函数 $h(w(l,r))$ 也满足四边形不等式和区间包含单调性；

性质4：设 h 是一个下凸函数，若函数 w 满足四边形恒等式和区间包含单调性，则复合函数 $h(w(l,r))$ 也满足四边形不等式。