E3-D题-小水獭和切钢条

21371258 冯睿冰

钢条切割问题（一维）

给定长度为 n 的钢条和长度为 i 的钢条价格表p_i（$i=1, \cdots, n$）
求切割钢条方案使得收益r_n最大

转化为规模更小的子问题：

$$r_n = \max(p_n, r_1 + r_{n-1}, r_2 + r_{n-2}, \cdots, r_{n-1} + r_1)$$

简化版本：（最优解只包含一个相关子问题，而不是两个）

$$r_n = \max_{1 \leqslant i \leqslant n}(p_i + r_{n-i})$$

BOTTOM-UP-CUT-ROD(p, n)

```
let r[0..n] be a new array
r[0] = 0
for j = 1 to n
    q = -∞
    for i = 1 to j
        q = max(q, p[i] + r[j-i])
    r[j] = q
return r[n]
```

E3-D题：钢条切割问题（二维）

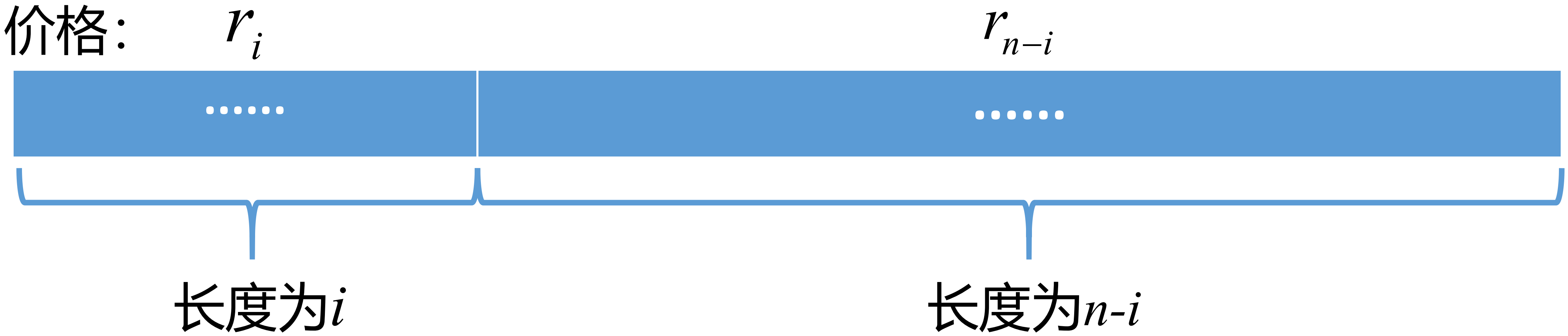
给定长度为 n 的钢条和长度不超过 m 的钢条价格表 p_i（$i=1$，2，$\cdots m$），求将钢条切割为$\boxed{\left\lceil \frac{n}{m} \right\rceil, \left\lceil \frac{n}{m} \right\rceil + 1, \ldots, n \text{ 段}}$的钢条方案使得收益最大

输出格式

对于每组数据，输出一行 $n - \left\lceil \frac{n}{m} \right\rceil + 1$ 个正整数，表示将钢条切割为 $\left\lceil \frac{n}{m} \right\rceil, \left\lceil \frac{n}{m} \right\rceil + 1, \ldots, n$ 段的最大总销售价格。

一维

$$r_n = \max(p_n, r_1 + r_{n-1}, r_2 + r_{n-2}, \cdots, r_{n-1} + r_1)$$

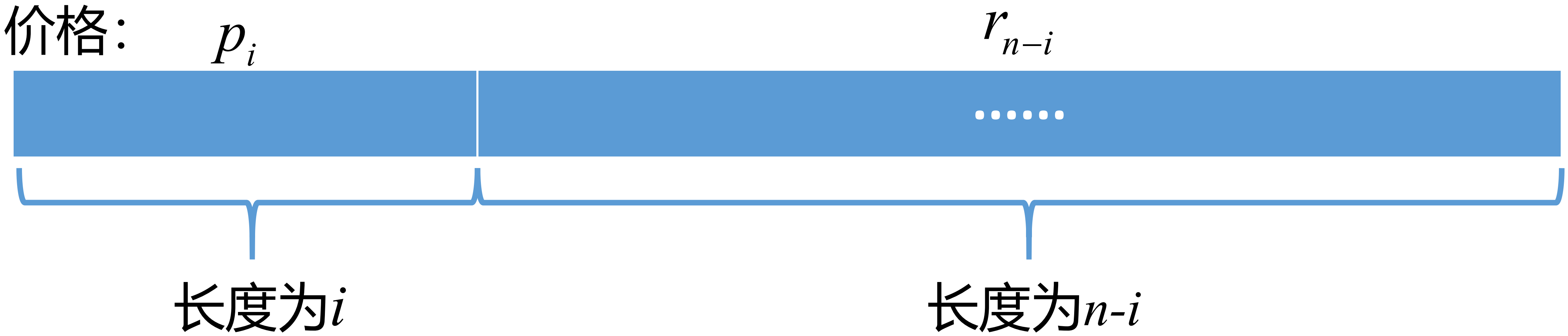


$$r_n = \max_{0 \le i \le n}(r_i + r_{n-i})$$

简化版本：（最优解**只包含一个相关子问题**，而不是两个）

一维

$$r_n = \max_{1 \leqslant i \leqslant n}(p_i + r_{n-i})$$



二维

定义 r_{ij} 为将长度为 i 的钢条切割为 j 段的最大收益（切下来的钢条长度不超过m）

$$\boxed{\left\lceil \frac{i}{m} \right\rceil \le j \le i}$$

价格：r_{xy} $r_{i-x,j-y}$

长度为x，段数为y　　长度为i-x，段数为j-y

则有 $r_{ij} = \max\limits_{0\le x\le i,0\le y\le j}(r_{xy} + r_{i-x,j-y})$ 且 $\left\lceil \frac{x}{m} \right\rceil \le y \le x$, $\left\lceil \frac{i-x}{m} \right\rceil \le j-y \le i-x$

简化版本：（最优解**只包含一个相关子问题**，而不是两个）

二维

定义 r_{ij} 为将长度为*i*的钢条切割为*j*段的最大收益（切下来的钢条长度不超过*m*）

$$\boxed{\left\lceil \frac{i}{m} \right\rceil \le j \le i}$$

价格: $\boxed{r_{x1} = p_x (1 \le x \le m)}$

$r_{i-x,j-1}$

长度为*x*，段数为1 　　　　长度为*i*-*x*，段数为*j*-1

则有 $r_{ij} = \max\limits_{1 \le x \le \min(i,m)} (p_x + r_{i-x,j-1})$ 　且 $\left\lceil \frac{i-x}{m} \right\rceil \le j-1 \le i-x$

```
q=max(q,p[x]+dp[i-x][j-1]);
```

$\text{CUT}(n, m)$

let $dp[1 \cdots n, 1 \cdots n]$ be a new table

for $i = 1$ **to** m

 $dp[i,1] = p[i]$

 if $i == n$

  print $dp[i,1]$

for $i = 1$ **to** n **n**

 for $j = 2$ **to** i **n**

  if $\left\lceil \frac{i}{m} \right\rceil \le j \le i$

   $q = -\infty$

   for $x = 1$ **to** $\min(m, i-1)$ **10**

    if $\left\lceil \frac{i-x}{m} \right\rceil \le j-1 \le i-x$

     $\boxed{q = \max(q, p[x] + dp[i-x, j-1])}$

   $dp[i, j] = q$

   if $i == n$

    print $dp[i, j]$

一维

$$r_n = \max_{0 \le i \le n}(r_i + r_{n-i})$$

$$r_n = \max_{1 \le i \le n}(p_i + r_{n-i})$$

$$\Theta(n^2)$$

二维

$$r_{ij} = \max_{0 \le x \le i, 0 \le y \le j}(r_{xy} + r_{i-x,j-y})$$

$$r_{ij} = \max_{1 \le x \le \min(i,m)}(p_x + r_{i-x,j-1})$$

$$\Theta(10n^2)$$

注意到题目中n和m的范围：

$1 \le n \le 5 \times 10^3$, $1 \le m \le \min(n, 10)$

$$r_{ij} = \max_{1 \le x \le \min(i,m)} (p_x + r_{i-x,j-1})$$

```
long long dp[5010][5010];
q=max(q,p[x]+dp[i-x][j-1]);
```

MLE

```
long long dp[15][5010]
q=max(q,p[x]+dp[(i-x)%11][j-1]);
```

AC

(先遍历长度)

```
long long dp[5010][2]
q=max(q,p[x]+dp[i-x][(j-1+2)%2]);
```

AC

(先遍历段数)

先遍历长度

```
long long max(long long a,long long b){
	return a>b?a:b;
}
int my_ceil(int x,int y){
	return (x/y*y==x)?x/y:x/y+1;
}
void cut(int n,int m) {
	int i,j;
	int x;
	long long q;
	for(i=1; i<=m; i++) {
		dp[i%11][1]=p[i];
		if(i==n){
			printf("%lld ",dp[i%11][1]);
		}
	}
	for(i=1; i<=n; i++) {
		for(j=2; j<=i; j++) {
			int s=my_ceil(i,m);
			if(j>=s&&j<=i) {
				q=-1;
				for(x=1;x<=m&&x<=i-1;x++){
					int r=my_ceil(i-x,m);
					if(j-1>=r&&j-1<=i-x)
						q=max(q,p[x]+dp[(i-x)%11][j-1]);
				}
				dp[i%11][j]=q;
				if(i==n){
					printf("%lld ",dp[i%11][j]);
				}
			}
		}
	}
}
```

先遍历段数

```
long long max(long long a,long long b){
	return a>b?a:b;
}
int my_ceil(int x,int y){
	return (x/y*y==x)?x/y:x/y+1;
}
void cut(int n,int m) {
	int i,j;
	int x;
	long long q;
	for(i=1; i<=m; i++) {
		dp[i][1]=p[i];
		if(i==n){
			printf("%lld ",dp[i][1]);
		}
	}
	for(j=2; j<=n; j++) {
		for(i=j; i<=n; i++) {
			int s=my_ceil(i,m);
			if(j>=s&&j<=i) {
				q=-1;
				for(x=1;x<=m&&x<=i-1;x++){
					int r=my_ceil(i-x,m);
					if(j-1>=r&&j-1<=i-x)
						q=max(q,p[x]+dp[i-x][(j-1+2)%2]);
				}
				dp[i][j%2]=q;
				if(i==n){
					printf("%lld ",dp[i][j%2]);
				}
			}

		}
	}
}
```