

算法分析与设计

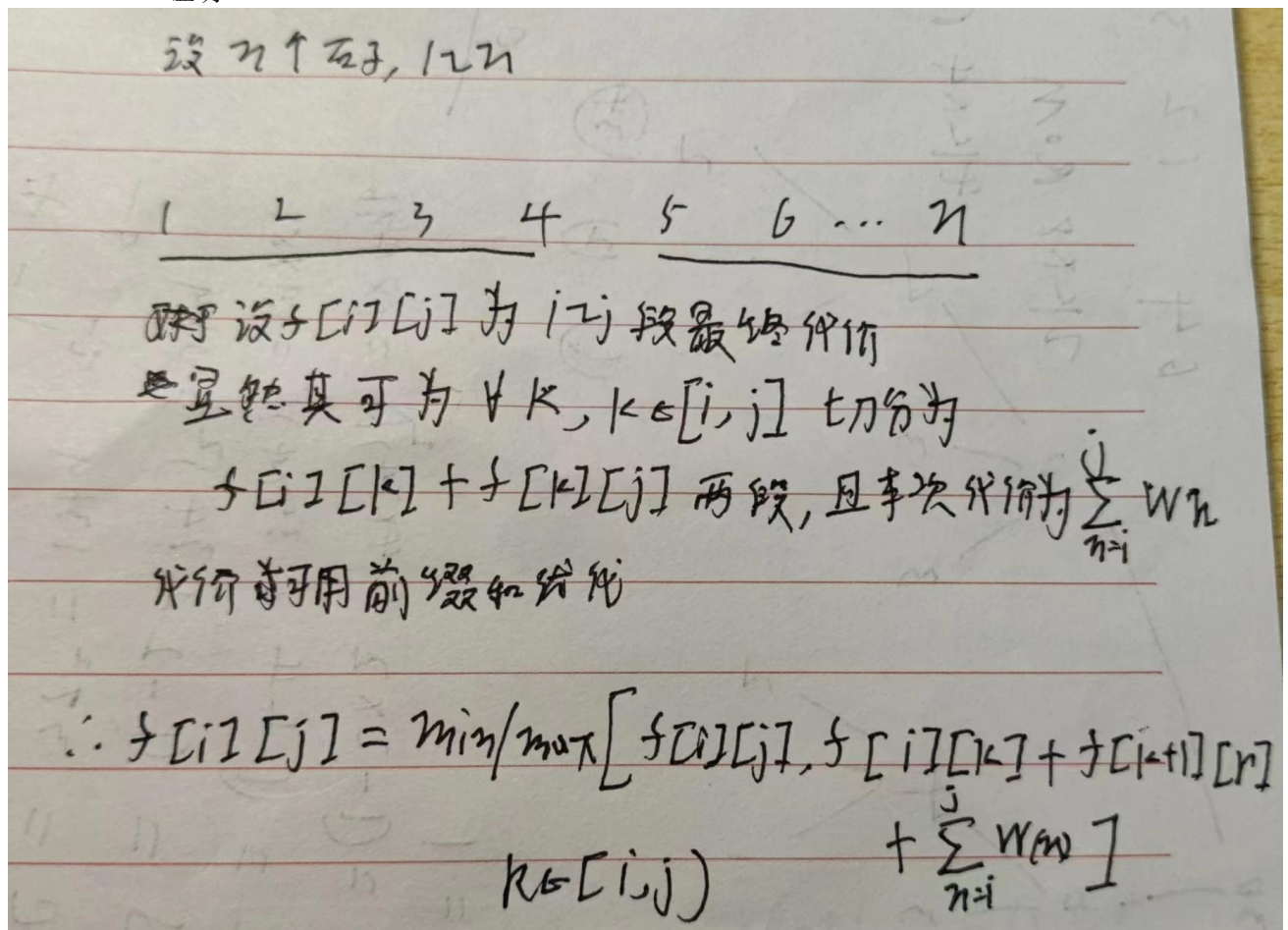
实验 03

实验目的

- 掌握动态规划的概念：最优子结构和子问题重叠性质。
- 掌握动态规划的解题步骤。
- 掌握采用动态规划解决矩阵连乘、最长公共子序列、凸多边形的最优三角剖分问题、简单 0-1 背包问题和最优二叉搜索树问题。

实训内容

- 1、熟练掌握动态规划的最优子结构性质和子问题重叠性质，并使用算法模拟软件辅助理解矩阵连乘的求解过程。
- 2、算法实现题：3-3 石子合并问题（P83）。编程实现，然后分析该问题的子问题重叠性质，并证明该问题的最优子结构性质。
 - 证明



● 区间 DP 实现:

```

1  #include<iostream>
2  #include<cstring>
3  #define ll long long
4  using namespace std;
5  const int N=310;
6  int n;
7  ll s[N],f[N][N],g[N][N];
8  int main(){
9      cin>>n;
10     //前缀和
11     for(int i=1;i<=n;i++){
12         cin>>s[i];
13         s[i]=s[i-1];
14     }
15     memset(f,0x3f,sizeof f);
16     for(int len=1;len<=n;len++){
17         for(int l=1;l+len-1<=n;l++){
18             int r=l+len-1;
19             if(len==1){
20                 f[l][r]=g[l][r]=0;
21                 continue;
22             }
23             for(int i=l;i<r;i++){
24                 f[l][r]=min(f[l][r],f[l][i]+f[i+1][r]+s[r]-s[l-1]);
25                 g[l][r]=max(g[l][r],g[l][i]+g[i+1][r]+s[r]-s[l-1]);
26             }
27         }
28     }
29     cout<<"最大分数: "<<g[1][n]<<endl;
30     cout<<"最小分数: "<<f[1][n];
31     return 0;
32 }

```

result.txt

```

1  Test 1 Passed
2
3  Input 1:
4  4
5  1 3 5 2
6
7  Expected Output :
8  最大分数: 29
9  最小分数: 22
10
11 Obtained Output :
12 最大分数: 29
13 最小分数: 22
14
15
16
17

```

3、算法实现题：3-6 租用游艇问题（P84）。

● 证明

设有 n 个点，编号 $0 \sim n-1$

显然，所有可行方案构成一个有向图，边数 n^2 ，点数 n 。

由此：① 可用 spfa 在期望 $O(m)$ 内求最短路

② 令 $f(i)$ 为 $0 \sim i$ 最小代价，显然对于 $\forall i, i \in [0, n]$ 所有中点 $k, k \in [0, i-1]$

$$f(i) = \min[f(i), f(k) + r(k, i)]$$

可得状态转移方程，复杂度 $O(n^2)$

● 线性 DP 实现

```
Test.cpp 算法分析与设计 3 > 实验 > D_More_Wrong > Test.cpp > solve()
1 #include<iostream>
2 #include<cstring>
3 #include<algorithm>
4 #include<cmath>
5 #include<queue>
6 #include<stack>
7 #include<set>
8 #include<map>
9 #include<unordered_set>
10 #include<unordered_map>
11 using namespace std;
12 #define ll long long
13 #define endl '\n'
14 #define pb push_back
15 #define inf 0x3f3f3f3f
16 #define fi first
17 #define se second
18 #define pll pair<ll,ll>
19 #define pii pair<int,int>
20 #define all(v) v.begin(), v.end()
21 const int N=5e5+10;
22 ll n,m;
23 void solve(){
24     cin>>n;
25     //初始化代价q[i][j]为i上j下的代价
26     //f[i]为0~i最小代价
27     vector<int> q[n],f(n,1e9);
28     f[0]=0;
29     for(int i=0;i<n;i++){
30         q[i].reserve(n);
31         for(int j=i+1;j<n;j++){
32             cin>>q[i][j];
33         }
34         for(int i=0;i<n;i++){
35             for(int j=0;j<i;j++){
36                 f[i]=min(f[i],f[j]+q[j][i]);
37             }
38         }
39     }
40     int main(){
41         solve();
42         return 0;
43     }
44 }
```

```
Test 1 Passed
1
2
3 Input 1:
4 3
5 5 15
6 7
7
8 Expected Output :
9 12
10
11 Obtained Output :
12 12
13
14
15
16
```

● spfa 图论实现

```
Test.cpp 算法分析与设计 3 > 实验 > D_More_Wrong > Test.cpp > spfa()
6 using namespace std;
7 #define ll long long
8 #define endl '\n'
9 #define pb push_back
10 #define all(v) v.begin(), v.end()
11 const int N=5e5+10;
12 ll n,m;
13 int e[N],ne[N],h[N],idx,w[N];
14 void add(int a,int b,int k){
15     e[idx]=b,ne[idx]=h[a],h[a]=idx++;
16     w[idx]=k;
17 }
18 int d[N],st[N];
19 void spfa(){
20     memset(d,0x3f,sizeof d);
21     int q[N],hh=0,tt=0;
22     q[0]=0,d[0]=0;
23     while(hh<=tt){
24         int t=q[hh++];
25         st[t]=0;
26         for(int i=h[t];i!=-1;i=ne[i]){
27             int j=e[i];
28             if(d[j]>d[t]+w[i]){
29                 d[j]=d[t]+w[i];
30                 if(!st[j]){
31                     q[++tt]=j;
32                     st[j]=1;
33                 }
34             }
35         }
36     }
37 }
38 void spfaSolve(){
39     cin>>n;
40     memset(h,-1,sizeof h);
41     for(int i=0;i<n;i++){
42         for(int j=i+1;j<n;j++){
43             int t;cin>>t;
44             add(i,j,t);
45         }
46     }
47     spfa();
48     cout<<"最小代价: "<<d[n-1];
49 }
50 void solve(){
51     cin>>n;
52     //初始化代价q[i][j]为i上j下的代价
53 }
```

```
Test 1 Passed
1
2
3 Input 1:
4 3
5 5 15
6 7
7
8 Expected Output :
9 最小代价: 12
10
11 Obtained Output :
12 最小代价: 12
13
14
15
16
```