



# ACM/ICPC Template

AsZ VincentLDL

August 15, 2015

# Contents

<b>1</b>	<b>java</b>			
1.1	读写	2	3.6	费用流
1.2	高精	2	3.7	弦图
		2	3.8	最小树形图
<b>2</b>	<b>dp 优化</b>		<b>3</b>	<b>4 数据结构</b>
2.1	决策单调性优化	3	4.1	splay
2.2	单调队列优化以及写仙人掌图	4	4.2	lct
2.3	斜率优化	6	4.3	可持久化线段树
2.3.1	斜率以及 $x$ 维都单调	6	4.4	点分治
2.3.2	随便什么情况: cdq 分治优化	7	4.5	树链剖分
<b>3</b>	<b>图论</b>		<b>9</b>	<b>5 字符串</b>
3.1	two sat	9	5.1	后缀数组
3.2	割顶, 点双联通分量	9	5.2	后缀自动机
3.3	桥, 边双联通分量	9		
3.4	平面图	9	<b>6</b>	<b>其他算法</b>
3.5	dinic	9	6.1	cdq 分治与读入优化

# Chapter 1

## java

1.1 读写

1.2 高精

## Chapter 2

# dp 优化

### 2.1 决策单调性优化

- 形式： $f[i] = f[j] + w[j, i]$  形式决策单调。
- 一般打表找规律看决策是否单调。
- 四边形不等式： $w[i, j] + w[i + 1, j + 1] \leq w[i + 1, j] + w[i, j + 1]$ , 则满足决策单调性。
- 有时候不满足决策单调性，但是去掉完全不合格状态之后可以满足。

```
1 #include <bits/stdc++.h>
2 #define MAXN 51234
3
4 using namespace std;
5 typedef long long arrayN[MAXN];
6
7 deque < pair< pair<int, int> , int> > deq;
8 arrayN f, sum, c;
9 long long L;
10
11 long long sqr(long long x)
12 {
13     return x * x;
14 }
15
16 long long trans(int l, int r)
17 {
18     return sqr(1LL * r - (l + 1) - L + sum[r] - sum[l]) + f[l];
19 }
```

```
20 int myLowBound(pair <int, int> pr, int ori, int now)
21 {
22     int l = pr.first, r = pr.second;
23     for (; l < r; )
24     {
25         int mid = l + r >> 1;
26         if (trans(ori, mid) <= trans(now, mid)) l = mid + 1;
27         else r = mid;
28     }
29     return l;
30 }
31
32 int main()
33 {
34     int n;
35     freopen("toys.in", "r", stdin);
36     cin >> n >> L;
37     for (int i = 1; i <= n; ++i)
38     {
39         cin >> c[i];
40         sum[i] = sum[i - 1] + c[i];
41     }
42     deq.push_back(make_pair(make_pair(1, n), 0));
43     for (int i = 1; i <= n; ++i)
44     {
45         for (; deq.front().first.second < i; deq.pop_front());
46         f[i] = trans(deq.front().second, i);
47         if (i == n) break;
48         deq.front().first.first = i + 1;
49         if (deq.front().first.second < i + 1) deq.pop_front();
50         for (; !deq.empty() && trans(deq.back().second, deq.back().
51 first.first) >= trans(i, deq.back().first.first); deq.pop_back
52 ());
53         if (deq.empty()) deq.push_back(make_pair(make_pair(i + 1, n
54 ), i));
55         else
56         {
57             int x = myLowBound(deq.back().first, deq.back().second,
58 i);
59             if (trans(i, x) >= (trans(deq.back().second, x))) x++;
60             deq.back().first.second = x - 1;
61             if (x <= n) deq.push_back(make_pair(make_pair(x, n), i)
62 );
63         }
64     }
65 }
```

```

61     cout << f[n] << endl;
62     return 0;
63 }

```

Listing 2.1: hnoi2008toys.cpp

## 2.2 单调队列优化以及写仙人掌图

- 题目背景：仙人掌图上最长链
- 形式： $f[i] = \max(g[j]) + w[i]$ ， $w[i]$  单调，可见，如果  $j < k$ ， $g[j] < g[k]$ ，则  $j$  可以直接不考虑，所以此时维护  $g$  单调减的队列即可。
- 仙人掌图找环：首先形成 bfs 树，发现有环，记  $pt$ ,  $ph$ ，然后选  $pt$  沿着  $pre$  走到跟，一路打时间戳；再从  $ph$  沿着  $pre$  走，就可以找到  $lca$ 。  $pt$ ,  $ph$  到  $lca$  的路径，加上  $pt \rightarrow ph$  就是基环了。

```

1  #include <bits/stdc++.h>
2  #define MAXN 1123456
3  #define MAXM 2123456
4
5  typedef int arrayN[MAXN], arrayM[MAXM];
6
7  using namespace std;
8
9  arrayN fir, cost, t, pre, vis;
10 arrayM e, nxt, c;
11 long long ans, dst[MAXN];
12 int num, now, visNow;
13
14 void link(int u, int v, int w)
15 {
16     e[++num] = v, nxt[num] = fir[u];
17     fir[u] = num, c[num] = w;
18 }
19
20 vector<int> bfsFindCycle(int x)
21 {
22     ++now;
23     vector<int> cyc;
24     deque<int> deq;
25     int pt = 0, ph = 0, last;
26     deq.push_back(x);
27     t[x] = now;
28     for (; !deq.empty() && !pt;)

```

```

29 {
30     int u = deq.front();
31     deq.pop_front();
32     for (int p = fir[u]; p && !pt; p = nxt[p])
33         if (e[p] != pre[u])
34             if (t[e[p]] == now)
35             {
36                 pt = u, ph = e[p];
37                 last = c[p];
38             }
39             else
40             {
41                 t[e[p]] = now;
42                 pre[e[p]] = u;
43                 cost[e[p]] = c[p];
44                 deq.push_back(e[p]);
45             }
46     }
47     vector<int> cycTmp;
48     if (pt)
49     {
50         ++now;
51         int tmp = pt;
52         for (; tmp != x; tmp = pre[tmp])
53             t[tmp] = now;
54         t[x] = now;
55         int lca = ph;
56         for (; t[lca] != now; lca = pre[lca]);
57         for (tmp = pt; tmp != lca; tmp = pre[tmp])
58         {
59             swap(last, cost[tmp]);
60             cyc.push_back(tmp);
61         }
62         cyc.push_back(lca);
63         cost[lca] = last;
64
65         for (tmp = ph; tmp != lca; tmp = pre[tmp])
66             cycTmp.push_back(tmp);
67         for (; !cycTmp.empty(); cycTmp.pop_back())
68             cyc.push_back(cycTmp.back());
69     } else cyc.push_back(x);
70
71     ++now;
72     for (int i = 0; i < cyc.size(); ++i)
73         t[cyc[i]] = now;
74     return cyc;

```

```

75 }
76
77 struct node
78 {
79     long long w;
80     long long lst, f;}g[MAXN * 2];
81
82 long long bfsLongest(int rt, int &nrt)
83 {
84     long long lst = 0;
85     deque <int> deq;
86     deq.push_back(rt);
87     nrt = rt;
88     vis[rt] = ++visNow;
89     dst[rt] = 0;
90     for (; !deq.empty(); )
91     {
92         int u = deq.front();
93         deq.pop_front();
94         for (int p = fir[u]; p; p = nxt[p])
95             if (vis[e[p]] != visNow && t[e[p]] != now)
96             {
97                 vis[e[p]] = visNow;
98                 dst[e[p]] = dst[u] + c[p];
99                 if (dst[e[p]] > lst)
100                 {
101                     lst = dst[e[p]];
102                     nrt = e[p];
103                 }
104                 deq.push_back(e[p]);
105             }
106     }
107     return lst;
108 }
109
110 long long solve(int x)
111 {
112     long long ans = 0;
113     vector <int> cyc = bfsFindCycle(x);
114     int n = cyc.size();
115     for (int i = 0; i < n; ++i)
116     {
117         int pt, pp;
118         t[cyc[i]] = 0;
119         g[i].lst = bfsLongest(cyc[i], pt);
120         ans = max(ans, bfsLongest(pt, pp));

```

```

121         t[cyc[i]] = now;
122         if (n == 1) return g[i].lst;
123         g[i].w = cost[cyc[i]];
124         g[i].f = 0;
125         g[i + n] = g[i];
126     }
127     g[0].w = 0;
128     for (int i = 1; i < 2 * n; ++i)
129         g[i].w += g[i - 1].w;
130     g[0].f = g[0].lst;
131     deque <int> deq;
132     deq.push_back(0);
133     for (int i = 1; i < 2 * n; ++i)
134     {
135         for (; deq.front() + n <= i; deq.pop_front());
136         g[i].f = g[i].lst + g[i].w + g[deq.front()].lst - g[deq.
front()].w;
137         for (; !deq.empty() && g[deq.back()].lst - g[deq.back()].w
<= g[i].lst - g[i].w; deq.pop_back());
138         deq.push_back(i);
139     }
140     for (int i = 0; i < 2 * n; ++i)
141         ans = max(ans, g[i].f);
142     return ans;
143 }
144
145 int main()
146 {
147     freopen("island.in", "r", stdin);
148     int n;
149     num = 1;
150     scanf("%d", &n);
151     for (int i = 1; i <= n; ++i)
152     {
153         int v, len;
154         scanf("%d%d", &v, &len);
155         link(i, v, len);
156         link(v, i, len);
157     }
158     long long ans = 0;
159     for (int i = 1; i <= n; ++i)
160         if (!vis[i])
161             ans += solve(i);
162     printf("%lld\n", ans);
163     return 0;

```

## 2.3 斜率优化

- $f[i] = \min(a[i] * x[j] + b[i] * y[j])$
- 更好的理解：设  $P=f[i]$ , 则  $y = (-a/b)x + P/b$ . 求满足要求的最小截距。或者通过各种转化，最优决策就是从无穷远朝原点移动，第一个碰上的点为最优决策点。
- 很好的性质：所有最优决策一定在当前所有点构成的凸包上。（例如，在最优决策点划一条相应斜率的线，其余点均在该线上方，）

### 2.3.1 斜率以及 x 维都单调

想像斜率越来越大的直线往 y 正方向移动，第 i 次移动首次碰上 k。对于以后的决策，因为斜率更大，那么在 k 之前，第 i 次移动没有碰上的点必然再也用不上了，所以可以维护一个单调队列。下面例题是：把一个序列切开，每个部分权值是和平方加常数，求权值和最小值

```

1 #include <deque>
2 #include <cstdio>
3 #include <cstring>
4 #include <iostream>
5 #include <cstdlib>
6
7 #define MAXN 512345
8
9 using namespace std;
10 typedef long long arrayN[MAXN];
11
12 struct node
13 {
14     long long x, y, f;
15     node (long long tx = 0, long long ty = 0, long long tf = 0)
16     {
17         x = tx, y = ty, f = tf;
18     }
19     //y = f + sum^2, x = sum
20 }g[MAXN];
21
22 long long sqr(long long x)
23 {

```

```

24     return x * x;
25 }
26
27 long long cross(long long x1, long long y1, long long x2, long long
    y2)
28 {
29     return x1 * y2 - x2 * y1;
30 }
31
32 deque < int > deq;
33
34 int main()
35 {
36     freopen("hdu3507.in", "r", stdin);
37     int N, M;
38     for (; scanf("%d%d", &N, &M) != EOF; )
39     {
40
41         deq.clear();
42         g[0] = node(0, 0, 0);
43         deq.push_back(0);
44         for (int i = 1; i <= N; ++i)
45         {
46             int x;
47             scanf("%d", &x);
48             g[i].x = g[i - 1].x + x;
49             long long lim = g[i].x << 1;
50             for (; deq.size() > 1; deq.pop_front())
51             {
52                 node u = g[deq[0]];
53                 node v = g[deq[1]];
54                 if ((v.y - u.y) > lim * (v.x - u.x))
55                     break;
56             }
57             node pt = g[deq.front()];
58             g[i].f = pt.f + sqr(g[i].x - pt.x) + M;
59             g[i].y = sqr(g[i].x) + g[i].f;
60             for (; deq.size() >= 2; deq.pop_back())
61             {
62                 node A = g[deq[deq.size() - 2]];
63                 node B = g[deq[deq.size() - 1]];
64                 node C = g[i];
65                 if (cross(B.x - A.x, B.y - A.y, C.x - B.x, C.y - B.
y) > 0) break;
66             }
67             deq.push_back(i);

```



```

68     }
69     cout << g[N].f << endl;
70 }
71 return 0;
72 }

```

Listing 2.3: hdu3507.cpp

### 2.3.2 随便什么情况：cdq 分治优化

- 排序的顺序，凸壳的方向写之前一定要画清楚。
- 这里归并排一维的序可以节省一个  $\log$  的复杂度
- cdq 分治的顺序至关重要，千万不能乱。
- $f[i]$  表示第  $i$  天手上的券全换成现金最多多少，其中  $x[j], y[j]$  分别表示用  $f[j]$  的钱换成 A, B 券分别能有多少。
- $f[i] = \max(\max(A[i] * x[j] + B[i] * x[j], f[j]))$
- 就是经典的斜率优化问题咯。不用平衡树的话可以离线用 cdq 分治。先按照  $A[i]/B[i]$  排序（具体大小顺序画一画就知道了）。solve(l, r) 时需要按照下标 lab 大小分为两部分。然后 solve(l, mid)，同时主义归并把递散维  $x$  排好序。l ~ mid 至 mid + 1 r 转移。最后 solve(mid + 1, r)，接着归并排好  $x$  就行了。

```

1 #include <bits/stdc++.h>
2 #define MST(a, b) memset((a), (b), sizeof(a))
3 #define MAXN 112345
4 #define esp 1e-8
5
6 using namespace std;
7
8 struct node
9 {
10     double A, B, rate; //A/B
11     double x, y;
12     double f;
13 }g[MAXN];
14
15 int lab[MAXN], a[MAXN];
16
17 int cmp(double x)
18 {
19     if (x < -esp) return -1;
20     if (x > esp) return 1;

```

```

21     return 0;
22 }
23
24 int smaller(int u, int v)
25 {
26     int tx = cmp(g[u].x - g[v].x);
27     int ty = cmp(g[u].y - g[v].y);
28     return tx < 0 || (tx == 0 && ty < 0);
29 }
30
31 void mergeSortX(int al, int ar, int bl, int br)
32 {
33     int Na = 0;
34     for (int i = al; i <= ar; ++i)
35     {
36         while (bl <= br && smaller(lab[bl], lab[i]))
37             a[++Na] = lab[bl++];
38         a[++Na] = lab[i];
39     }
40     for (; bl <= br; ++bl)
41         a[++Na] = lab[bl];
42     for (int i = 1; i <= Na; ++i)
43         lab[al + i - 1] = a[i];
44 }
45
46 double cross(int A, int B, int C)
47 {
48     return (g[B].x - g[A].x) * (g[C].y - g[B].y) - (g[B].y - g[A].y)
49         * (g[C].x - g[B].x);
50 }
51
52 double comRate(int A, int B, int C)
53 {
54     return (g[B].y - g[A].y) * g[C].B + g[C].A * (g[B].x - g[A].x);
55 }
56
57 void getRightPartF(int al, int ar, int bl, int br)
58 {
59     int Na = 0;
60     double lim = 0;
61     for (int i = al; i <= ar; ++i)
62     {
63         lim = max(lim, g[lab[i]].f);
64         while (Na >= 2 && cmp(cross(a[Na - 1], a[Na], lab[i])) >=
0)

```

```

65     —Na;
66     a[++Na] = lab[i];
67 }
68 int La = 1;
69 for (int i = bl; i <= br; ++i)
70 {
71     int p = lab[i];          g[p].f = max(g[p].f, lim);
72     for (; La + 1 <= Na && cmp(comRate(a[La], a[La + 1], p)) >= 16
73         0; ++La);
74     g[p].f = max(g[p].f, g[a[La]].x * g[p].A + g[a[La]].y * g[p].B);
75 }
76
77 void solve(int l, int r)
78 {
79     if (l == r)
80     {
81         int p = lab[l];
82         //g[p].f = max(g[p].f, g[p - 1].f);
83         g[p].x *= g[p].f;
84         g[p].y *= g[p].f;
85         return ;
86     }
87     int Na = r - l + 1;
88     int upLim = 0, downLim = MAXN;
89     for (int i = l; i <= r; ++i)
90     {
91         upLim = max(upLim, lab[i]);
92         downLim = min(downLim, lab[i]);
93     }
94     int midLim = (upLim + downLim) >> 1;
95     int pLow = 0;
96     for (int i = l; i <= r; ++i)
97         if (lab[i] <= midLim)
98             a[++pLow] = lab[i];
99     int pHigh = pLow;
100    for (int i = l; i <= r; ++i)
101        if (lab[i] > midLim)
102            a[++pHigh] = lab[i];
103    for (int i = 1; i <= Na; ++i)
104        lab[i + l - 1] = a[i];
105    pLow += l - 1;
106    solve(l, pLow);
107    getRightPartF(l, pLow, pLow + 1, r);
108    solve(pLow + 1, r);
109
110    mergeSortX(l, pLow, pLow + 1, r);
111 }
112 int com(int u, int v)
113 {
114     node tu = g[u];
115     node tv = g[v];
116     return tu.A * tv.B < tv.A * tu.B;
117 }
118
119 int main()
120 {
121     // freopen("cash4.in", "r", stdin);
122     int N, S;
123     scanf("%d%d", &N, &S);
124     for (int i = 1; i <= N; ++i)
125     {
126         scanf("%lf%lf%lf", &g[i].A, &g[i].B, &g[i].rate);
127         g[i].y = 1.0 / (g[i].B + g[i].A * g[i].rate);
128         g[i].x = g[i].y * g[i].rate;
129         g[i].f = S;
130         lab[i] = i;
131     }
132     g[1].f = S;
133     sort(lab + 1, lab + N + 1, com);
134     solve(1, N);
135     double ans = 0;
136     for (int i = 1; i <= N; ++i)
137         ans = max(ans, g[i].f);
138     printf("%.3f\n", ans);
139     return 0;
140 }

```

Listing 2.4: cash.cpp

# Chapter 3

## 图论

### 3.1 tarjan

#### 3.1.1 2-sat

#### 3.1.2 割顶，点双联通分量

#### 3.1.3 桥，边双联通分量

### 3.2 平面图

farmland 那道题

### 3.3 网络流

#### 3.3.1 dinic

#### 3.3.2 费用流

#### 3.3.3 常见模型

### 3.4 弦图

### 3.5 最小树形图

## Chapter 4

# 数据结构

4.1 splay

4.2 lct

4.3 可持久化线段树

4.4 点分治

4.5 树链剖分

## Chapter 5

# 字符串

### 5.1 后缀数组

### 5.2 后缀自动机

## Chapter 6

# 其他算法

### 6.1 cdq 分治与读入优化

- 不要排结构体，因为排结构体到时候还要排回来。
- 线段树打时间戳不要 `memset()`;
- 在严格小的限制下，第二维排序的时候一定要双关键字排序
- 这题是三维空间中，三个坐标都不减的最长链

```
1 #include <iostream>
2 #include <cstring>
3 #include <cstdlib>
4 #include <cstdio>
5 #include <algorithm>
6 #define REP(i, n) for(int i = 0; i < (int) (n); ++i)
7 #define REPP(i, a, b) for(int i = (int) (a); i <= (int) (b); ++i)
8 #define REDD(i, a, b) for(int i = (int) (a); i >= (int) (b); --i)
9 #define MST(a, b) memset((a), (b), sizeof(a))
10 #define MAXN 111111
11 #include <vector>
12
13 using namespace std;
14 int zLim;
15
16 long long gTot[MAXN * 4];
17 int t, g[MAXN * 4], n, ti[MAXN * 4], now;
18 struct node
19 {
20     int x, y, z, f;
```

```
21     long long tot;
22 } a[MAXN];
23
24 int comx(node A, node B)
25 {
26     return (A.x < B.x) || ((A.x == B.x) && (A.y < B.y)) || ((A.x ==
27         B.x) && (A.y == B.y) && A.z < B.z);
28 }
29
30 int comy(node A, node B)
31 {
32     return A.y < B.y;
33 }
34
35 void change(int pos, int x, long long cnt)
36 {
37     pos += t;
38     if (ti[pos] != now) g[pos] = gTot[pos] = 0;
39     if (x < g[pos]) return;
40     if (x == g[pos]) gTot[pos] += cnt;
41     else gTot[pos] = cnt, g[pos] = x;
42     ti[pos] = now;
43
44     for(pos >>= 1; pos; pos >>= 1)
45     {
46         if (ti[pos << 1] != now) g[pos << 1] = gTot[pos << 1] = 0;
47         if (ti[pos << 1 ^ 1] != now) g[pos << 1 ^ 1] = gTot[pos << 1
48             ^ 1] = 0;
49         ti[pos] = now;
50         g[pos] = max(g[pos << 1], g[pos << 1 ^ 1]);
51         gTot[pos] = 0;
52         if (g[pos] == g[pos << 1]) gTot[pos] += gTot[pos << 1];
53         if (g[pos] == g[pos << 1 ^ 1]) gTot[pos] += gTot[pos << 1 ^ 1];
54     }
55
56 int ask(int l, int r, long long &cnt)
57 {
58     if (l > r) return 0;
59     int tmp = 0;
60     cnt = 0;
61     l += t - 1, r += t + 1;
62     for (; (l ^ r) != 1; l >>= 1, r >>= 1)
63     {
64         if (!(l & 1))
```

```

65         if (ti[l + 1] == now)
66     {
67         if (tmp == g[l + 1]) cnt += gTot[l + 1];
68         else if (tmp < g[l + 1])
69     {
70         tmp = g[l + 1];
71         cnt = gTot[l + 1];
72     }
73     }
74     if (r & 1)
75     {
76         if (ti[r - 1] == now)
77     {
78         if (tmp == g[r - 1]) cnt += gTot[r - 1];
79         else if (tmp < g[r - 1])
80     {
81         tmp = g[r - 1];
82         cnt = gTot[r - 1];
83     }
84     }
85     }
86     }
87     return tmp;
88 }
89
90 void solve(int l, int r)
91 {
92     if (l == r) return ;
93     int mid = (l + r) >> 1;
94     solve(mid + 1, r);
95
96     sort(a + mid + 1, a + r + 1, comy);
97     sort(a + l, a + mid + 1, comy);
98
99     // MST(g, 0);
100    //MST(gTot, 0);
101    ++now;
102    int pos = r + 1;
103    REDD(i, mid, l)
104    {
105        for (; pos > mid + 1 && a[pos - 1].y >= a[i].y; --pos)
106        {
107            change(a[pos - 1].z, a[pos - 1].f, a[pos - 1].tot);
108        }
109    }
110

```

```

111    long long tmpTot;
112    int tmp = ask(a[i].z, zLim, tmpTot) + 1;
113    if (a[i].f == tmp) a[i].tot += tmpTot;
114    else if (a[i].f < tmp)
115    {
116        a[i].f = tmp;
117        a[i].tot = tmpTot;
118    }
119    }
120
121    sort(a + l, a + r + 1, comx);
122    solve(l, mid);
123 }
124
125 int INT()
126 {
127     int res;
128     char ch;
129     while (ch = getchar(), !isdigit(ch));
130     for (res = ch - '0'; ch = getchar(), isdigit(ch);)
131         res = res * 10 + ch - '0';
132     return res;
133 }
134
135 int main()
136 {
137     int task;
138     freopen("in.txt", "r", stdin);
139     now = 0;
140     for (task = INT(); task; --task)
141     {
142         n = INT();
143         vector <int> dataZ;
144         REPP(i, 1, n)
145         {
146             a[i].x = INT();
147             a[i].y = INT();
148             a[i].z = INT();
149             a[i].f = 1;
150             a[i].tot = 1;
151             dataZ.push_back(a[i].z);
152         }
153         sort(dataZ.begin(), dataZ.end());
154         dataZ.resize(unique(dataZ.begin(), dataZ.end()) - dataZ.
155         begin());
156         REPP(i, 1, n)

```

156	{	165	{
157	a[i].z = (lower_bound(dataZ.begin(), dataZ.end(), a[i].	166	if (ans == a[i].f) cnt += a[i].tot;
	z) - dataZ.begin()) +1;	167	else if (ans < a[i].f) cnt = a[i].tot, ans = a[i].f;
158	}	168	}
159	zLim = dataZ.size();           for (t = 1; t <= zLim + 1; t	169	printf("%d %lld\n", ans, cnt);
	<= 1);	170	}
160	sort(a +1, a + n +1, comx);	171	return 0;
161	solve(1, n);	172	}
162	int ans = 0;		
163	long long cnt = 0;		
164	REPP(i, 1, n)		

Listing 6.1: hdu4742.cpp