积分赛第二场题解

Voleking

May 13, 2017

Contents

A – Protecting the Flowers (Greedy)

A – 题意

. . .

- 保护祖国的花朵：约翰的奶牛每分钟吃掉 D_i 朵花，把它赶走需要 T_i 分钟（来回加倍）。问最小损失花朵数量。

贪心策略

尽量先赶走吃得多并且走得慢的牛，如何衡量 “多”“慢”？需要两头牛 (a, b) 作比较：
a.T * b.D < b.T * a.D

Code

```
scanf("%d", &N);
for (int i = 0; i < N; ++i) {
    scanf("%d%d", &p[i].second.first, &p[i].second.second);
    p[i].first = (double)p[i].second.first / p[i].second.second;
    sum += p[i].second.second;
}
sort(p, p + N);
for (int i = 0; i < N; ++i) {
    sum -= p[i].second.second;
    ans += sum * p[i].second.first * 2;
}
printf("%lld\n", ans);
```

B – The Meeting Place Cannot Be Changed (Ternary Search)

B – 题意

. . .

- 找啊找啊找朋友：南北方向上的路有 n 个朋友，每个人有一个坐标 X_i 和速度 V_i，求所有朋友最小会面时间。

三分搜索

设坐标 X 为最优面基地点，可以想象 X 1 –> 1e9，所有人到达时间先减小后增大。EPS!!!

Code

```
double C(double x) {
    double res = 0;
    for (int i = 0; i < N; i++)
        res = max(abs(X[i] - x) / V[i], res);
```

```
    return res;
}

int main(int argc, char const *argv[])
{
    scanf("%d", &N);
    double lb = 1e9, ub = 1;
    for (int i = 0; i < N; i++) {
        scanf("%lf", X + i);
        lb = min(X[i], lb); ub = max(X[i], ub);
    }
    for (int i = 0; i < N; i++)
        scanf("%lf", V + i);
    while (lb + EPS < ub) {
        double ll = (2. * lb + ub) / 3.;
        double rr = (lb + 2. * ub) / 3.;
        if (C(ll) <= C(rr)) ub = rr;
        else lb = ll;
    }
    printf("%.10f\n", C(lb));
    return 0;
}
```

C – Chloe and pleasant prizes (DFS)

C – 题意

...

- 给一棵树，求两棵不相交子树的最大和。

DFS

第一遍 DFS，找到每个结点最大和次大子树的值 m[v]，mm[v]；第二遍 DFS，求出答案。

Code

```
ll cnt(int v, int p) {
    ll res = A[v]; m[v] = INF, mm[v] = INF;
    for (int u : G[v])
        if (u != p) {
            ll tmp = cnt(u, v);
            res += tmp;
            tmp = max(tmp, m[u]);
```

```
            if (tmp > m[v]) {
                mm[v] = m[v]; m[v] = tmp;
            } else if (tmp > mm[v]) mm[v] = tmp;
        }
    return res;
}
ll dfs(int v, int p) {
    ll ans = INF;
    for (int u : G[v])
        if (u != p) ans = max(ans, dfs(u, v));
    if (m[v] != INF && mm[v] != INF) ans = max(ans, m[v] + mm[v]);
    return ans;
}

int main(int argc, char const *argv[])
{
    scanf("%d", &N);
    rep(i, 0, N) scanf("I64d", A + i);
    rep(i, 1, N) {
        scanf("%d%d", &u, &v); --u, --v;
        G[u].push_back(v);
        G[v].push_back(u);
    }
    cnt(0, -1);
    ll ans = dfs(0, -1);
    if (ans == INF) printf("Impossible\n");
    else cout << ans << endl;
    return 0;
}
```

D – Ability To Convert (DP)

D – 题意

. . .

- 给出 n 进制和一个对应的数，问最小换成多大的十进制数。(10 直接用 10 代替)

贪心

从后往前取，每次取最大的不超过 n 的数。

DP

a[i][j] 表示从 i 开始连续 j 个数是多少，-1 不存在。
dp[i] 表示前 i 个数最小能换成多大的十进制数。

Code

```
cin >> n >> s;
memset(a, -1, sizeof a);
for (int i = 0; i < 60; i++) dp[i] = INF;
for (int i = 0; i < s.size(); i++) {
    a[i][1] = s[i] - '0';
    if (i) for (int j = 2; j < 10; j++)
            if (a[i - 1][j - 1] > 0) a[i][j] = a[i - 1][j - 1] * 10 + a[i][1];
    for (int j = 1; j < 10; j++)
        if (a[i][j] != -1 && a[i][j] < n) {
            if (j == i + 1) dp[i] = a[i][j];
            else if (dp[i - j] <= INF / n) dp[i] = min(dp[i], dp[i - j] * n + a[i][j]);
        }
}
cout << dp[s.size() - 1] << endl;
```

E – Pythagorean Triples (Math)

E – 题意

...

- 勾股数，知一求二

数学，构造

$$(n+1)^2 - n^2 = 2n + 1$$

$$(n^2+1)^2 - (n^2-1)^2 = 2 * (2 * n^2) = (2n)^2$$

Code

```
cin >> a;
if (a % 2 == 1) {
    ll n = a / 2;
    b = 2LL * n * (n + 1);
```

```
    c = 2LL * n * (n + 1) + 1;
} else {
    ll n = a / 2;
    b = n * n - 1;
    c = n * n + 1;
}
if (b && c)
    cout << b << " " << c << endl;
else cout << -1 << endl;
```

F – Teacher Bo (Pigeonhole Principle)

F – 题意

...

- n 个点中找两个点对，使得曼哈顿距离相等。

抽屉原理

坐标的范围为 10^5，曼哈顿距离为 $2 * 10^5$，暴力枚举。

Code

```
cin >> N >> M;
for (int i = 0; i < N; ++i) {
    scanf("%d%d", &X[i], &Y[i]);
}
if (N * (N - 1) > 4 * M - 2) {
    printf("YES\n");
    continue;
}
memset(flag, 0, sizeof flag);
bool ok = false;
for (int i = 0; i < N && !ok; ++i)
    for (int j = i + 1; j < N && !ok; ++j) {
        int dis = abs(X[j] - X[i]) + abs(Y[j] - Y[i]);
        ++flag[dis];
        if (flag[dis] >= 2) {
            ok = true;
            break;
        }
    }
if (ok)
```

```
    printf("YES\n");
else
    printf("NO\n");
```

G – Problem Buyer (Greedy & STL)

G – 题意

...

- 有 n 个区间，求至少选多少个，使得给定的 m 个数能对应到任选的区间中，每个数只能对应一个区间。

xjb

先考虑只有一个数的情况，假设有 a 个区间不能与之匹配，则至少要选择 a + 1 个区间。再考虑两个数的情况，假设分别有 a，b 个区间不能与这两个数匹配，则至少要选则 max(a,b) + 1 个。如果有两个数不能匹配区间都一致时（或两个数相等），显然有 a=b，则要选 a + 2 个。

Code

```
scanf("%d", &t);
for (int _ = 0; _ < t; _++) {
    scanf("%d%d", &n, &m);
    for (int i = 0; i < n; i++)
        scanf("%d%d", &seg[i].first, &seg[i].second);
    for (int i = 0; i < m; i++)
        scanf("%d", C + i);
    sort(seg, seg + n);
    sort(C, C + m);
    int ans = 0;
    priority_queue<int, vector<int>, greater<int> > que;
    for (int i = 0, it = 0; i < m; i++) {
        while (it < n && seg[it].first <= C[i]) que.push(seg[it++].second);
        while (!que.empty() && que.top() < C[i]) que.pop();
        if (que.size() == 0) {ans = -1; break;}
        ans = max(ans, n - (int)que.size());
        que.pop();
    }
    if (~ans) printf("Case #%d: %d\n", _ + 1, ans + 1);
    else printf("Case #%d: IMPOSSIBLE!\n", _ + 1);
}
```

Thx