Chapter 1

搜索

1.1 折半搜索

FZU 2178 礼物分配

给 $n(n \leq 30)$ 个物品和这些物品分别对 A 与 B 的价值，然后需要将这些物品给 A 和 B，A 和 B 分别拥有的数量 $numA$ 和 $numB$ 要满足：$numA + numB = n$ 且 $\mid numA - numB \mid \leq 1$。求两人获得的价值和之差绝对值最小值：$min(\mid sumA - sumB \mid)$。

不妨假设第一人取了 $former = \frac{n+1}{2}$ 件物品，第二个人取了 $later = \frac{n}{2}$ 件物品。因为 n 最大为 30，如果直接枚举的话肯定会超时的。考虑第一个人在前 $former$ 个中取 i 个可以获得的价值的所有情况：这个可以状压 dp 递推出来，然后存在一个数组中。接着状压枚举第二个人在后 $later$ 件物品中取的情况，如果第二个人在后 $later$ 件物品中取的数量固定了，那么第一个人在前 $former$ 件物品中取的数量也固定了。只要二分查找这种情况的最优解即可。

时间复杂度：$O(\frac{n+1}{2} * 2^{\frac{n+1}{2}} * \log K)$

```
const int MAX_N = 35;
const int inf = 0x3f3f3f3f;

int T, n;
int v[MAX_N], w[MAX_N];
int store[16][10000], num[16];
// C[15][8] 大概 6400 左右

void solve() {
    memset(num, 0, sizeof (num));
    int former = (n + 1) / 2, later = n / 2;
    for (int s = 0; s < (1 << former); ++s) {
        int ret1 = 0, ret2 = 0, tmp = 0;
        for (int i = 0; i < former; ++i) {
            if (s & (1 << i)) ret1 += v[i], tmp++;
            else ret2 += w[i];
        }
        store[tmp][num[tmp]++] = ret1 - ret2;
    }
    for (int i = 0; i <= former; ++i) { sort(store[i], store[i] + num[i]); }
    int ans = inf;
    for (int s = 0; s < (1 << later); ++s) {
        int ret1 = 0, ret2 = 0, tmp = 0;
        for (int i = 0; i < later; ++i) {
            if (s & (1 << i)) ret2 += w[i + former];
            else ret1 += v[i + former], tmp++;
        }
        int left = former - tmp, key = ret2 - ret1;
        int pos = lower_bound(store[left], store[left] + num[left], key) - store[left];
        if (pos >= num[left]) ans = min(ans, abs(store[left][num[left] - 1] - key));
        else {
            ans = min(ans, abs(store[left][pos] - key));
            if (pos > 0) ans = min(ans, abs(store[left][pos - 1] - key));
            if (pos < num[left] - 1) ans = min(ans, abs(store[left][pos + 1] - key));
        }
    }
    printf(”%d\n”, ans);
}

int main() {
    scanf(”%d”, &T);
    while (T--) {
        scanf(”%d”, &n);
        for (int i = 0; i < n; ++i) { scanf(”%d”, &v[i]); }
        for (int i = 0; i < n; ++i) { scanf(”%d”, &w[i]); }
        solve();
    }
    return 0;
}
```