

斜率优化模板详解

```
//优化形如 $dp[i] = \max(b[j] * a[i] + c[j])$ 的 $dp$  不要求 $b[j]$ 单调  $\min$ 时取反即可
//动态维护一个下凸壳, 复杂度 $O(n \log n)$ 
//适用于 $\text{int}$  无精度误差

#include <bits/stdc++.h>

typedef long long ll;
const ll INF = LLONG_MAX;
inline ll divide(ll a, ll b){return a / b - ((a ^ b) < 0 && a % b);} //严格向下取整, 适用于负数

class L{
public:
    static bool isquery;
    //mutable的作用在于使multiset的const_iterator也能修改它所指向的元素, 如果不加则不能修改
    mutable ll k, b, p;

    //p是该条线段的右端点
    L (ll k = 0, ll b = 0, ll p = 0):k(k), b(b), p(p){}
    bool operator < (const L &l) const{return isquery ? p < l.p : k < l.k;}
    ll gety(ll x) const{return k * x + b;}
};
bool L::isquery = false;

typedef std::multiset <L>::iterator iter;

//l1和l2是两条斜率相邻的直线  $l1.k < l2.k$ 
//该函数判断l1是否完全在l2上方, 如果不是, 顺便计算出l1的右端点
bool over(std::multiset <L> &set, iter l1, iter l2){
    //没有比插入直线斜率更大的直线, 右端点为+INF 返回false不再继续删除
    if (l2 == set.end()) return l1 -> p = INF, false;

    //斜率相同时比较截距, 保留截距较大的直线
    if (l1 -> k == l2 -> k) l1 -> p = l1 -> b > l2 -> b ? INF : -INF;
    else l1 -> p = divide(l2 -> b - l1 -> b, l1 -> k - l2 -> k);

    //向下取整的原因见图1 A点向上取整时 l2的值仍然比l1大, 向下取整时 l2的值则已经小于l1了
    return l1 -> p >= l2 -> p;
}

void insert(std::multiset <L> &set, ll k, ll b){
    //v,w是插入的直线在set中的iterator u是斜率恰大于v的一条直线
    auto u = set.insert(L (k, b)), v = u ++, w = v;

    //当u完全被v覆盖时, 删除u
    while (over(set, v, u)) u = set.erase(u);
}
```

```
//如果v完全在凸壳下面，那么删除v这里能用over判断的原因如图2 A1确实在A2左边。
//这里要再写一遍over(set, w, set.erase(v))的原因是if中计算了over(set, -- w,
v)这个“假”交点，
//需要用原来正确的交点覆盖回来。
if (v != set.begin() && over(set, -- w, v)) over(set, w, set.erase(v));

//这里实现和斜率大的情况稍有些不同，是先判断(-- w) -> p >= v -> p这个条件，
//再计算下一个w与v的交点（第一个w在上面的if中已经算过了）
while ((v = w) != set.begin() && (-- w) -> p >= v -> p) over(set, w,
set.erase(v));
}

ll query(std::multiset<L> &set, ll x){
    L::isquery = true;
    auto u = *(set.lower_bound(L(0, 0, x)));
    L::isquery = false;
    return u.gety(x);
}

int main(){
    return 0;
}
```



From:
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:intrepidword:zhongzihao:slopt_opt

Last update: 2021/04/01 23:16