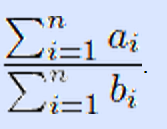
# 分数规划

分数规划是用来解决如下式子的最优问题的。



具体描述一般为n个物品分别有价值a[i],和体积b[i],现在需要你至少选择k个物品使得K= sum[a[i]]/sum[b[i]]（即单位体积性价比）最大。

对于此类题目，我们进行一下变形

K= sum[a[i]]/sum[b[i]]==》 0= sum[a[i]]- sum[b[i]]\*K

这样我们可以二分K，显然K越大，sum[a[i]]- sum[b[i]]\*K越小

然后计算r[i]=a[i]-b[i]\*K

选取r[i]最大的k个，看看sum[r[i]]是否大于0，如果大于0，说明至少对于当前K是一定可以满足的，如果小于0，说明就不能满足。如果等于0，说明这就是答案。

Poj 2976

题目大： 给定n个二元组(a,b)，去掉k个二元组，使得剩下的a元素之和与b元素之和的比率最大

 题目求的是 max(∑a[i] \* x[i] / (b[i] \* x[i]))   其中a,b都是一一对应的。 x[i]取0,1  并且 ∑x[i] = n - k;

 转：那么可以转化一下。  令K = ∑a[i] \* x[i] / (b[i] \* x[i])  则必然∑a[i] \* x[i] - ∑b[i] \* x[i] \* K= 0；（条件1）

并且任意的 ∑a[i] \* x[i] - ∑b[i] \* x[i] \* max(K) <= 0  (条件2，只有当∑a[i] \* x[i] / (b[i] \* x[i]) = max(K) 条件2中等号才成立)

     然后就可以枚举K , 对枚举的K， 求Q(r) = ∑a[i] \* x[i] - ∑b[i] \* x[i] \*Kr  的最大值，为什么要求最大值呢？  因为我们之前知道了条件2，所以当我们枚举到K为max(K)的值时，显然对于所有的情况Q(K)都会小于等于0，并且Q(K)的最大值一定是0.而我们求最大值的目的就是寻找Q(K)=0的可能性，这样就满足了条件1，最后就是枚举使得Q(K)恰好等于0时就找到了max(K)。而如果能Q(K)>0 说明该K值是偏小的,而Q(K)<0的话，很明显是K值偏大的，因为max(K)都是使Q(K)为0，说明不可能存在Q(K)=0了。

Poj3691

题意：给出一个图，求一个回路，使得回路上的点权之和/边权之和最大。

用SPFA[**算法**](http://lib.csdn.net/base/datastructure)，二分枚举K，判断是否存在负权回路，若存在，说明K偏小了，则增大K，若不存在，则减小K。

[poj2728-最小比率生成树](http://blog.csdn.net/zhang20072844/article/details/8176854)

题目意思：

有n个村庄，村庄在不同坐标和海拔，现在要对所有村庄供水，只要两个村庄之间有一条路即可，建造水管距离为坐标之间的欧几里德距离，费用为海拔之差，现在要求方案使得费用与距离的比值最小，很显然，这个题目是要求一棵最优比率生成树。

解法同上，每次都计算r[i][j]=a[i][j]-b[i][j]\*K的边跑kruscal或者prim，当最后答案跑出来为0就行了这题用了newton加速很明显。

Newton 迭代

虽然没搞懂原理（好像有点像泰勒n阶展开式）,但是这玩意貌似很好用的样子。

它可以用来代替二分，而且速度比二分快。

他和二分的时不同点在于，二分返回的是 ：在系数为K的情况下选k个有没有解。也就返回个0或者1

牛顿返回，在系数为K的情况下，我选取的最优解来更新K，

即K=sum[a[i]]/sum[b[i]],这样一旦我这次的最优解和上次相差小于eps就可以停止了， 注意此时K初始化为0。