石子分裂

算法 1 (sub 1)

$n=1$ 时考虑怎样分裂最优，显然是尽可能的等分。证明如下：

对于任意两堆石子，它们的大小分别为 $x,y(x\le y)$ ，代价为 x^2+y^2 。如果 $y-x\ge 2$ ，那么将它们改为 $x+1,y-1$ ，代价即为 $x^2+2x+1+y^2-2y+1=x^2+y^2-2(y-x-2)-2<x^2+y^2$ ，严格变小。

因此最终的每两堆石子大小相差不超过 1 。

主观难度：2

算法 2 (sub 2)

$m=1$ 时显然选最大的石子操作即可，证明略。

主观难度：2

算法 3 (sub 2-3)

根据算法 1 ，我们可以 $O(1)$ 计算出一堆石子操作若干次的代价。设 $cost(x,y)$ 表示把大小为 x 的石头操作 y 次的代价。

背包 DP ，设 $f_{i,j}$ 表示把前 i 堆石子操作 j 次的最小代价，转移枚举第 i 堆石子的操作次数，有

$$f_{i,j}=\min_{k=0}^{j}\{f_{i-1,j-k}+cost(a_i,k)\}$$

时间复杂度 $O(Tnm^2)$ 。

主观难度：3

算法 4 (sub 4)

算法 3 实际上是一个泛化物品背包的模型。每堆石子可以看做一个泛化物品，给它 i 次操作就会得到 $f(i)$ 的代价，两堆石子组成的整体仍然是一个泛化物品，事实上任意两个泛化物品都可以合并为一个新的泛化物品，我们要求的就是 n 个泛化物品的并。

对于单点修改可以用线段树维护泛化物品的并，复杂度 $O(Tm^2\log n+nm^2)$ 。

主观难度：4~5

算法 5 (sub 1-5)

算法 4 中的泛化物品就是一个离散函数，两个泛化物品合并就是做 $\min+$ 卷积。

不难发现初始时的离散函数是凹函数，因此合并实际上是做闵科夫斯基和，据此可以将算法 3 优化至 $O(Tnm)$ ，把算法 4 优化至 $O(Tm\log n+nm)$ 。

主观难度：5~6

算法 6 (sub 1-6)

考虑算法 5 中做闵科夫斯基和干了肾摸事情。两个物品合并就是把两组向量合起来排序，事实上把 n 个物品合并也就是把 n 组向量合起来排序。但事实上，我们只需要知道操作 m 次的结果，也就只需要知道前 m 个向量。

那么我们可以用堆维护每组向量，并依次贪心地取到前 m 个向量，事实上也可以理解为同时对 n 组向量进行归并排序。

于是复杂度被优化到 $O(T(n+m)\log n)$ 。

主观难度：6

肯德基

算法 1 (sub 1-2)

直接枚举因子并判断是否是平方数，复杂度 $O(Tn^2)$ 或 $O(T+n^2)$ 。

使用不同的枚举因子技巧还可以做到 $O(T+n^{1.5})$ 或 $O(T+n\log n)$ 。

主观难度：2

算法 2 (sub 1-3)

考虑更快地预处理每个数的最大平方因子。设 $f(x)$ 表示 x 的最大平方因子，容易注意到 f 实际上是积性函数。使用欧拉线性筛可以做到 $O(T+n)$ 。

主观难度：2~3

算法 3 (sub 1-6)

既然我们只关注积性函数 f 的前缀和，那么不难想到使用各种亚线性筛优化：

3.1 杜教筛 (sub 1-4)

常规做法复杂度 $O(Tn^{2/3})$ 。但是由于多组数据，我们可以在杜教筛的预处理部分预处理更多的值。

若我们预处理 f 的前 B 个值，复杂度事实上是 $O(B+T\sqrt{\frac{n^2}{B}})$ 。简单分析可以知道最优复杂度为 $O((Tn)^{2/3})$ 。

3.2 min25 筛 (sub 1-5)

复杂度玄学。

3.3 powerful number (sub 1-6)

注意到对于质数 p 有 $f(p)=1$ ，而同样满足 $I(p)=1$ 的恒等函数 I 的前缀和可以 $O(1)$ 求得，所以我们可以爆搜 powerful number 算 f 的前缀和。复杂度 $O(T\sqrt{n})$ 。

具体的，设积性函数 $g=f\times I^{-1}$ ，记 S_a 表示 a 的前缀和函数，我们要求的就是 $S_f(n)$ ：

$$
\begin{aligned}
S_f(n) &= \sum_{i=1}^{n} f(i) \\
&= \sum_{i=1}^{n} \sum_{d|i} g(d) I(\frac{i}{d}) \\
&= \sum_{d=1}^{n} g(d) \sum_{j=1}^{n/d} I(j) \\
&= \sum_{d=1}^{n} g(d) S_I(\lfloor \frac{n}{d} \rfloor) \\
&= \sum_{d=1}^{n} g(d) \lfloor \frac{n}{d} \rfloor
\end{aligned}
$$

$g(d) \neq 0$ 的必要条件是 d 是 powerful number ，即标准分解中每个质因子的质数都不小于 2 。

算法 4 (sub 1-5)

考虑对于每个平方因子 x 计算它对答案贡献了多少次，也就是满足 $y \leq n \land f(y) = x$ 的 y 有多少个。

若 $x|f(y)$ ，设 $z = \dfrac{y}{x}$ ，不难想到 $f(y) = x$ 当且仅当 z 没有平方因子，即 $\mu(z)^2 = 1$。

设 $\rho(m) = \sum_{i=1}^{m} [\mu(i)^2 = 1]$ ，那么答案即

$$
\begin{aligned}
&\sum_{x} x \sum_{x|y} [y \leq n \land \mu(\frac{y}{x})^2 = 1] \\
=&\sum_{x} x \sum_{z}^{n/x} [\mu(z)^2 = 1] \\
=&\sum_{x} x \rho(\lfloor \frac{n}{x} \rfloor) \\
=&\sum_{i=1}^{\sqrt{n}} i^2 \rho(\lfloor \frac{n}{i^2} \rfloor)
\end{aligned}
$$

事实上 $\rho(m) = \sum_{i=1}^{\sqrt{m}} \mu(i) \lfloor \dfrac{m}{i^2} \rfloor$ ，直接计算的复杂度为 $O(\sqrt{n} \log n)$ 。

算法 5 (sub 1-7)

对于算法 3.3 ，有趣的是 $g(d) \neq 0$ 当且仅当 d 是平方数，因此原式即

$$
\sum_{i=1}^{\sqrt{n}} g(i^2) \lfloor \frac{n}{i^2} \rfloor
$$

整除分块可以优化到 $O(\sqrt{n} + T\sqrt[3]{n})$ 。

主观难度：7

独立集计数

算法 1 (sub 1)

图是一个环的话，设 f_n, g_n 分别表示大小为 n 的环和链的独立集数量，转移考虑某一个点是否选择，有：

$$
\begin{aligned}
f_n &= g_{n-1} + g_{n-3} \\
g_n &= g_{n-1} + g_{n-2}
\end{aligned}
$$

$O(n)$ 递推 g 即可。

主观难度：2~3

算法 2 (sub 2)

直接枚举点集并按定义判断，复杂度根据不同实现 $O(2^n)$ ~ $O(2^n n^2)$ 。

主观难度：2

算法 3

考虑从边集的角度计算，对于一个点集 S ，设 $f(S)$ 表示该点集的导出子图的边集，我们要求的就是

$$\sum_{S\subset V}[f(S)=\varnothing]$$

根据子集容斥的原理 $[S=\varnothing]=\sum_{T\subset S}(-1)^{|T|}$ ，可以知道

$$\text{原式}=\sum_{S\subset V}\sum_{T\subset f(S)}(-1)^{|T|}=\sum_{T\subset E}(-1)^{|T|}\sum_{S\subset V}[T\subset f(S)]$$

设边集 T 的端点集合为 $g(T)$ ，注意到 $T\subset f(S)\Leftrightarrow g(T)\subset S$ 。于是

$$\text{原式}=\sum_{T\subset E}(-1)^{|T|}\sum_{S\subset V}[g(T)\subset S]=\sum_{T\subset E}(-1)^{|T|}2^{|V|-|g(T)|}$$

时间复杂度 $O(2^m m)$ 。

主观难度：5

算法 4 (sub 2-5)

上接算法 2 ，考虑更快地从点集的角度求答案。

折半搜索，把点集分为 V_1,V_2 两部分，先暴力求出两个部分的所有独立集，然后枚举一边的独立集 S 统计另一边有多少独立集 T 满足 $S\cup T$ 仍是独立集。

对于 $S\subset V_1$ 设 $f(S)\subset V_2$ 表示与点集 S 没有边相连的点集。那么答案即

$$\sum_{S}\sum_{T}[T\subset f(S)]$$

对于第二个求和式，预处理子集和，或者递推计算一个点集内有多少独立集即可。复杂度根据根据实现 $O(2^{n/2})$ ~ $O(2^{n/2}n)$ 。

主观难度：5

算法 5

上接算法 3 ，考虑更快地从边集的角度求答案。

注意到一棵树是可以 $O(n)$ 递推出独立集数量的，我们随意建一颗生成树，把边集分成树边 E_1 和非树边 E_2 两部分。

设 $f_1(S),f_2(S)$ 分别表示点集 S 的导出子图的边集与 E_1 和 E_2 的交。沿用算法 3 的推导方式：

$$
\begin{aligned}
&\sum_{S\subset V}[f_1(S)=\varnothing][f_2(S)=\varnothing]\\
=&\sum_{S\subset V}[f_1(S)=\varnothing]\sum_{T\subset f_2(S)}(-1)^{|T|}\\
=&\sum_{T\subset E_2}(-1)^{|T|}\sum_{S\subset V}[f_1(S)=\varnothing][T\subset f_2(S)]\\
=&\sum_{T\subset E_2}(-1)^{|T|}\sum_{S\subset V}[f_1(S)=\varnothing][g(T)\subset S]
\end{aligned}
$$

对于第二个求和式，可以树形 DP 求值。总复杂度 $O(2^{m-n}m)$ 。

主观难度：6

算法 6 (sub 1-6)

算法 2, 4 从点集的角度统计答案，算法 3, 5 从边集的角度统计答案，综合两者即可简单地得到更优的复杂度。

总复杂度 $O(\min\{2^{n/2},2^{m-n}m\})=O(2^{m/3}m)$ 。

主观难度：8