浅析离散变换计数问题

江苏省扬州中学　胡渊鸣

May, 2012

Abstract

本文针对近年来出现的一些组合计数问题中的各类离散变换方法进行了一些归纳总结, 从最基本的离散变换思想入手, 最后上升到一定的抽象高度, 提出更加简便的解决方案.

由于本文侧重于思想方法的叙述, 很多在别的资料中容易找到的原理及证明本文将不再赘述.

[Key Words]离散变换, 反演, 组合计数, 符号化

Contents

1 离散变换简介

1.1 变换的思想

变换(Transform), 即是将一个对象通过某种规则变为另一个对象. 利用变换, 我们常常可以将某一难处理的对象A, 转化为另一层面上的易处理的对象A^*, 从而简化解题. 处理完A^*以后, 我们可以使用反变换, 得到需要的答案.

Table 1: 常见变换举例

变换名称	用途举例
*Laplace*变换	求解线性微分方程
*Fourier*变换	信号分析
洛伦茨(伽利略)变换	转换惯性参照系
二项式/*Stirling*/*Möbius*变换	组合数学

1.2 离散变换的运用

由于计算机不能处理连续的量, 实际计算中的变换往往是离散变换(Discrete Transform), OI中更是如此.

举例来说, 解线性方程组就是我们常用的离散变换. 很多时候, 对于一个无法求出的向量$\vec{x}$, 如果我们知道他的某个线性变换$\vec{y}$, $\vec{y} = \mathcal{A}\vec{x}$, 如果$|\mathcal{A}| \neq 0$, 那么我们便可以利用$\vec{x} = \mathcal{A}^{-1}\vec{y}$反解出$\vec{x}$

本文中提到的离散变换, 特指计数问题中的离散变换, 很多地方也叫做反演(Inversions)(本文中将不对这两个词进行区分). 其中, 近年来在各大比赛中出现的又主要是*Möbius*变换(*Möbius*反演), 二项式变换等. 由于这两类变换问题资料较少, 难度也较大, 本文就围绕这两种变换进行讨论.

2 传统离散变换方法

对于很多看似无从下手的计数问题, 我们往往可以先想想我们能够求出什么, 得到一个辅助计数对象的结果. 再用我们要求的计数对象去表示这些辅助计数对象, 得到一系列方程, 从而反解出目标计数对象.

2.1 例: CodeCraft 11 Number of Prime Strings

求由26个小写字母组成的周期和长度均为N的字符串的数目, 多组询问. 如"abcabc"周期为3, 不符合$N = 6$时的需求.

考虑到目标计数对象很难求出来, 我们不妨先想想我们能够求出什么. 不难发现我们很容易求出周期是$n(n\backslash N)$的约数, 长度为N的字符串的个数, 这样的字符串一定能够由任意长度为n的字符串重复$\frac{N}{n}$次得到, 于是这样的字符串的数目就是长度为n的所有字符串的数目, 即$G(n) = 26^n$

为了方便, 设$F(n)$为周期为n, 长度为N的字符串个数.

要求出$F(N)$, 我们必须找到$F(n)$与$G(n)$的关系. 不难发现, 对于长度为n的字符串, 如果我们枚举其周期, 就能得到$G(n) = \sum_{d\backslash n} F(d)$, 变形得到$F(n) = G(n) - \sum_{d\backslash n, d\neq n} F(d)$

显然$F(1) = 1$, $G(n)$也是很快可以求出的, 我们便可以利用上面的方程从小到大算出所有的$F(n)$, 问题也就解决了.

不过, 这样的算法效率似乎偏低, 如果我们采用每次求出$F(n)$后把$F(kn)$减去$F(n)$, 这样的复杂度达到了$O(N) - O(\sqrt{N}\lg N)$(有用的n必须是N的约数, 只有$O(\sqrt{N})$个), 并不理想.

如果我们对上面的方程$F(n) = G(n) - \sum_{d\backslash n, d\neq n} F(d)$进行进一步变形, 能得到更快的算法.

我们考虑不断把右边的F利用方程消去, 直到右边所有F全部消失, 只剩下G, 就能得到更简单的公式, 即$F(n) = \sum_{d\backslash n} \mu(\frac{n}{d})G(d) = \sum_{d\backslash n} \mu(\frac{n}{d})26^d$, 其中$\mu$ 是数论$M\ddot{o}bius$函数.

这样算法的复杂度就达到了$O(N) - O(\sqrt{N})$.

上面的例子虽然简单, 但是却提供了一个非常有用的公式, 即

$$G(n) = \sum_{d\backslash n} F(d) \Leftrightarrow F(n) = \sum_{d\backslash n} \mu(\frac{n}{d})G(d)$$

观察两个方法, 不难发现前面的方法实际上是利用加法原理分类讨论然后变形, 效率较低; 而后面的方法则是将前面得到的方程进行彻底的变形, 得到高效的解法. 实际上, 很多组合数学中的反演公式(离散变换), 如容斥原理, 都可以这样得到.

对于这种问题, 关键在于对题目中的计数对象F进行某些变形, 以致存在能够求出的合适的G, 我们不妨看一个例题.

2.2 例: NOI 2010 能量采集Energy

给定$N, M(N, M \leq 10^7)$, 求$\sum_{j=1}^{N}\sum_{k=1}^{M} \gcd(j,k)$

观察题目, 发现直接计算复杂度高达$O(N^2 \lg N)$, 必须换一个思路. 我们不妨设$F(n)$为$\gcd(a,b) = n, 1 \leq a \leq N, 1 \leq b \leq M$的数对$(a,b)$的个数, 那么答案就是$\sum kF(k)$

现在的问题是, 如何求出F呢? 考察一个类似的计数对象, 设$G(n)$为$n \setminus \gcd(a,b), 1 \le a \le N, 1 \le b \le M$的数对$(a,b)$的个数不难发现$G(n) = \lfloor \frac{N}{n} \rfloor \lfloor \frac{M}{n} \rfloor$.

为了通过G反解出F, 考虑G与F的关系. 不难发现, 如果我们枚举$\gcd(a,b)$, 就能得到$G(n) = \sum_{n \setminus d} F(d)$.

利用反演公式$F(n) = \sum_{n \setminus d} \mu(\frac{d}{n}) G(d)$

带入得$ans = \sum_d d \sum_{d \setminus k} \mu(\frac{k}{d}) \lfloor \frac{N}{k} \rfloor \lfloor \frac{M}{k} \rfloor = \sum_d d \sum_{1 \le k} \mu(k) \lfloor \frac{N}{kd} \rfloor \lfloor \frac{M}{kd} \rfloor$

由于出现了整除, 这个式子的计算可以采用经典的分段优化, 总复杂度可以达到$O(N)$.

2.3 离散变换的传统思路

关于运用离散变换计数, 一般思路如下:

首先选择一个恰当的计数对象(即一个从对象的特征到实数的函数), F, 然后选择一个恰当的变换G, 再考察G的意义, 得到G的表达式. 如果G是很容易求出的, 我们就可以利用G反演出F.

2.4 传统思路的弊端

观察上面总结出的思路, 不难发现, 每一步的选择, 都要求是“恰当的”, 这就对选手的直觉以及熟练程度提出了很高的要求. 如果选择了错误的计数对象, 或者选择了错误的变换, 解题将会前功尽弃.

2.5 表1 : 常见反演公式表

解题中能用到的反演公式一般就是以下这些, 读者可以自行体会. 由于该方法

有一些弊端, 可以不掌握这些公式而直接学习下面的内容. 最后一个公式较少见.

$$G(n)=\sum_{d\backslash n}F(d)\Leftrightarrow F(n)=\sum_{d\backslash n}\mu(\frac{n}{d})G(d)$$

$$G(n)=\sum_{n\backslash d}F(d)\Leftrightarrow F(n)=\sum_{n\backslash d}\mu(\frac{d}{n})G(d)$$

$$G(K)=\sum_{L\subseteq K}F(L)\Leftrightarrow F(L)=\sum_{K\subseteq L}(-1)^{|K|-|L|}G(K)$$

$$G(K)=\sum_{K\subseteq L}F(L)\Leftrightarrow F(L)=\sum_{L\subseteq K}(-1)^{|K|-|L|}G(K)$$

$$G(n)=\sum_{k\leq n}\binom{n}{k}F(k)\Leftrightarrow F(n)=\sum_{k\leq n}(-1)^{n-k}\binom{n}{k}G(k)$$

$$G(n)=\sum_{n\leq k}\binom{k}{n}F(k)\Leftrightarrow F(n)=\sum_{n\leq k}(-1)^{n-k}\binom{k}{n}G(k)$$

$$G(n)=\sum_{k\geq 1}F(n/k)\Leftrightarrow F(n)=\sum_{k\geq 1}\mu(k)G(n/k)$$

3 离散变换的抽象化

很多计数问题分析起来有难度, 我们不妨考虑将它们抽象化以达到简化问题的目的. 抽象化一方面的目是将很多组合计数问题就变成符号的游戏以便与思考, 另一方面可以帮助我们绕过深奥难懂的反演理论(限于拙劣的理解能力, 作者独自研究了几个月也才略知皮毛, 可见完全深入理解反演更是一项巨大的工程).

3.1 符号化初步

求由26个小写字母组成的周期和长度均为N的字符串的数目, 多组询问.

如果我们用S表示所有长度为N的字符串组成的集合, 定义$f:S\to Z$, 表示字符串s的周期, 那么我们要求的就是

$$\begin{aligned}
\sum_{s\in S}[f(s)=n] &= \sum_{s\in S}[f(s)\backslash n][\frac{n}{f(s)}=1] \\
&= \sum_{s\in S}[f(s)\backslash n]\sum_{d\backslash n}[d\backslash\frac{n}{f(s)}]\mu(d) \\
&= \sum_{s\in S}[f(s)\backslash n]\sum_{d\backslash n}[f(s)\backslash\frac{n}{d}]\mu(d) \\
&= \sum_{s\in S}\sum_{d\backslash n}[f(s)\backslash\frac{n}{d}]\mu(d) \\
&= \sum_{d\backslash n}\mu(d)\sum_{s\in S}[f(s)\backslash\frac{n}{d}] \\
&= \sum_{d\backslash n}\mu(\frac{n}{d})\sum_{s\in S}[f(s)\backslash d]
\end{aligned}$$

化简过程中, 我们利用了$[a=b]=[a\backslash b]\sum_{d\backslash\frac{a}{b}}\mu(d)]$这一变换核, 以及交换$\sum$的技巧.

观察我们得出的结果, $\sum_{s\in S}[f(s)\backslash d]$实际上就是周期是$d$的约数的字符串数目, 即$26^d$.

现在我们解决这类问题有了一个新的方法, 即先将要求的目标表示成和式的形式, 然后利用某种变换核将其中的谓词换掉, 然后通过交换和号起到改变计数对象的作用.

新方法和老方法本质相同. 使用这个方法的前提是读者要能够熟练运用$\sum$, $[statement]$等记号. 熟练掌握以后, 很多计数问题的转化就变得十分方便了.

上面这个问题给作者带来了一个重要思路, 即“符号化”. 经过一番研究, 外加贾志鹏神牛指点, 作者总结出一种解决离散变换的符号化的便捷方法.

定义计数对象的全集S, 并构造限制S中符合要求的对象的特征函数f, 这样我们就可以将我们的计数问题符号化为$\sum_{s\in S}[f(s)=X]$.

接下来是富有技巧性的一步, 选取恰当的变换核将$[f(s)=X]$换掉, 这样我们会得到一个含有两个$\sum$的表达式.

运用交换$\sum$的技巧, 我们会发现计数对象得到了某种程度上的变形. 如果我们选对了变换核, 新的计数对象将会更容易求出.

3.2 表2：常见变换核表

注意这也不是作者提出的最终解法, 读者稍微体会其中妙处后便可继续阅读后面的内容.

$$[n=1]=\sum_{d\backslash n}\mu(d)$$
$$[a=b]=[a\backslash b]\sum_{d\backslash \frac{a}{b}}\mu(d)$$
$$[a=b]=[b\backslash a]\sum_{d\backslash \frac{b}{a}}\mu(d)$$
$$[K=\varnothing]=\sum_{L\subseteq K}(-1)^{|L|}$$
$$[S=T]=[S\subseteq T]\sum_{S\subseteq L\subseteq T}(-1)^{|L|-|S|}$$
$$[S=T]=[T\subseteq S]\sum_{T\subseteq L\subseteq S}(-1)^{|L|-|T|}$$
$$[a=b]=\sum_{a\le k\le b}(-1)^{k-a}\binom{k}{a}\binom{b}{k}$$
$$[a=b]=\sum_{b\le k\le a}(-1)^{k-b}\binom{k}{b}\binom{a}{k}$$

使用变换核进行计数对象的转换, 可以帮助我们绕过反演这一深奥难懂的步骤. 我们不妨再看一个简单而能说明问题的题目.

3.3 例: 互质数对计数

求互质的数对(a,b)*的数目, 其中*$1\le a\le n, 1\le b\le m$

我们按部就班, 先设计数对象的$S=[1..n]\times[1..m]$, 于是我们要求的, 就是

$$\sum_{(a,b)\in S}[\gcd(a,b)=1]$$

利用变换核$[n=1]=\sum_{d\backslash n}\mu(d)$代换上式中的$[\gcd(a,b)=1]$, 得到

$$\sum_{(a,b)\in S}\sum_{d\backslash \gcd(a,b)}\mu(d)$$

交换两个$\sum$的顺序, 得到

$$\sum_{d}\mu(d)\sum_{(a,b)\in S}[d\backslash a][d\backslash b]$$

这时, 我们得到了一个新的计数对象, 即$\sum_{(a,b)\in S}[d\backslash a][d\backslash b]$. 考虑其实际意义, 不难得

到它实际上就是$\lfloor\frac{n}{d}\rfloor\lfloor\frac{m}{d}\rfloor$ 于是我们就得到了一个高效的式子

$$\sum_d \mu(d)\lfloor\frac{n}{d}\rfloor\lfloor\frac{m}{d}\rfloor$$

这个结果是相当令人满意的, 通过$O(n)$的预处理, 我们可以在$O(\sqrt{n})$的时间之内完成每次询问.

通过这个例子, 我们不难发现, 所谓的变换核, 帮助我们完成的就是谓词的转换. 一开始我们的谓词$[\gcd(a,b)=1]$过于严格, 以至于难以高效计数, 而利用$[n=1]=\sum_{d\backslash n}\mu(d)$之后, 我们得到的谓词$[d\backslash \gcd(a,b)]$是可以高效计数计数的.

这也提示我们, 选择变换核的时候, 我们就要考虑好自己希望得到什么样的谓词, 这样才能选对变换核.

关于选择变换核的技巧, 我们不妨再看一个题目.

3.4 例: JOI 一道计数好题

给定N, 求$[1..N]$的幂集合的子集(除去空子集)中, 子集中元素交集大小为M的子集个数.

这道题中, 我们取$S=\mathcal{P}(\mathcal{P}([1..N]))$, 即我们能取的方案实际上是$[1..N]$的幂集合的幂集合. 定义特征函数$f:\mathcal{P}(\mathcal{P}([1..N]))\to\mathcal{P}([1..N])$, 表示每种方案的交集.

我们要求的是

$$\sum_{s\in S}[|f(s)|=M]$$

. 为了简便计算, 我们定义$K=[1..M]$, 即一个势为M的$[1..N]$子集. 于是我们要求的就变成

$$\binom{N}{M}\sum_{s\in S}[f(s)=K]$$

不难发现我们随意取$[1..N]$的一个势为M的子集, 结果是不会变的. 这里出现了$[f(s)=K]$, 我们应该用哪个变换核呢? 不妨直接考虑, 这个谓词变成什么样才能方便我们计数? 经过一些尝试, 我们发现, 如果能够把谓词变成$[K\subseteq f(s)]$, 新的计数对象$\sum_{s\in S}[K\subseteq f(s)]$是很简单的, 考虑$|K|$个元素是必须出现在子集的所有元素中的, 而去掉这$|K|$个元素后, 剩下的$2^{N-|K|}$种剩余中则可以任取, 但不能什么都不取, 即$\sum_{s\in S}[K\subseteq f(s)]=2^{2^{N-|K|}}-1$.

先将能把$[f(s)=K]$变成$[K\subseteq f(s)]$的变换核

$$[S=T]=[T\subseteq S]\sum_{T\subseteq L\subseteq S}(-1)^{|L|-|T|}$$

带入原式并交换和号, 得到

$$\binom{N}{M}\sum_{L\supseteq K}(-1)^{|L|-|K|}\sum_{s\in S}[L\subseteq f(s)]$$

于是, 答案就为

$$\binom{N}{M}\sum_{K\subseteq L}(-1)^{|L|-|K|}(2^{2^{N-|L|}}-1)$$

集合个数是指数级的, 但是集合的势是线性的. 我们枚举第二个$\sum$中集合L的势k, 得到

$$\binom{N}{M}\sum_{M\le k}\binom{N-M}{k-M}(-1)^{k-M}(2^{2^{N-k}}-1)$$

至此, 我们就完成了这题的公式推导.

另外, 本题也可以使用二项式反演推导, 读者会发现, 二项式反演实际上是这种方法在计数对象只和集合的势有关而与集合包含的具体内容无关时的特例.

3.5 最终方法: 使用直接变换公式

上面的方法似乎还是较为麻烦, 特别是带入变换核后的交换$\sum$的过程, 对于不熟悉$\sum$的性质的选手来说还是很容易出错的. 不难发现, 带入变换核并变形只是一个中间步骤, 我们完全可以直接使用变形后的公式.

如, Number of Prime Strings一题中, 如果我们将计数对象符号化后, 直接使用公式

$$\sum_{s\in S}[f(s)=N]=\sum_{d\backslash N}\mu(\frac{N}{d})\sum_{s\in S}[f(s)\backslash d]$$

就能一步得出结论!

经过探索, 我们终于找到了最为简便的方法.

Step 1.(符号化) 定义计数对象的全集S, 并构造限制S中符合要求的对象的特征函数f, 这样我们就可以将我们的计数问题符号化为$\sum_{s\in S}[f(s)=X]$.

Step 2.(谓词转换) 找一个更加方便计数的谓词代替$[f(s)=X]$,

Step 3.(代入公式, 得到结果) 找到原谓词和新谓词的变换公式, 代入公式, 便可得到结果.

由此可见新方法比前面提到的传统方法思路更加清晰, 又比利用变换核带入的方法少了大片推导过程, 在时间紧迫的竞赛中, 该方法可谓是解题利器.

文末有一份详细的表格, 列出了解题中常用的几个公式. 理解以后对这些公式进行记忆并熟练运用, 就可以将很多反演计数问题快速解决.

4 总结

*Möbius*反演一直是组合数学中的一块硬骨头, 加上资料少, 内容多, 很多选手难以很好地掌握这块内容.

作者对这方面内容进行了较长期的思考, 将书本上的内容与实际解题结合, 将各种方法进行了整理, 理清了方法之间的关系. 作者在总结概念的基础上进行了扩展的思考, 提出了更为简便的解题方法, 是本文的亮点.

在学习相关内容的过程中, 作者遇到了很多困难, 有的问题甚至想了一个月还难以解决, 但这更加坚定了作者攻克难题的决心, 并决定整理出一份与OI紧紧相扣的关于离散变换计数的资料. 在不断的思考与总结中, 作者发现了各种离散变换的简单推导方法.

本文中运用的符号化技巧, 不仅适用于离散变换, 对于其它的公式推导也有很大的帮助. 这得益于*Concrete Mathematics*一书. 感兴趣的读者不妨去研究一下这本书中的介绍$\sum$的各种变形方法的部分.

限于篇幅, 作者不可能把复杂的*Möbius*反演理论全部呈现在读者面前, 不过本文中提到的方法, 足以解决作者遇到的全部涉及*Möbius*反演的问题. 读者如果有兴趣, 可以去参考*Introductory Combinatorics*一书的*Möbius Inversion*一节.

本文的另一个亮点是较好地呈现了作者探索便捷方法的过程, 从很多资料上都有的传统方法, 到利用变换核代换, 再到最终的直接代入公式, 作者经历了不下一年的探索, 走了很多弯路, 也从中学到了很多新的知识. 文末的几个公式, 是作者一年多探索过程中得到的精华部分. 既然几个公式就能说明全部内容, 为什么还要写这么长的文章? 对此, 作者的观点是

无论是书本上具体的知识, 还是解决问题的有效方法, 都是停留在“会”的层面上, 这对于当今时代对创新的高要求是远远不够的. 如果本文只是列举了几个公式, 这样是毫无意义的. 只有把作者艰辛探索的过程呈现的读者面前, 才能帮助读者深入理解作者的新方法, 同时也能展示新时代信息学奥赛选手不屈不挠, 永不放弃的顽强精神和勇于批判, 自我创新的思考能力.

另外, 本文中涉及的计数问题, 往往是在模某个数(一般是如1000000007的质数)的意义下的, 这是为了避免高精度计算, 同时也要求选手具备一定的数论知识, 如欧拉定理, 扩展Euclid算法, 中国剩余定理, 线性筛法及其扩展运用等. 由于这方面的内容不是本文讨论的重点, 读者可以自行查阅其他资料, 如参考文献[1], 以及本文附件中的代码.

本文中的三个表中的公式, 都是按顺序对应的, 即按照“反演公式-变换核-直接变换公式”对应. 读者可以从中体会到这三个方法的联系. 不仅如此, 每个表中的公

式内部也有一定的联系, 读者深入体会后联系便会逐渐显现出来.

5 表3：常见直接变换公式表

$$\sum_{s\in S}[f(s)=1]=\sum_{d}\mu(d)\sum_{s\in S}[d\backslash f(s)]$$
$$\sum_{s\in S}[f(s)=N]=\sum_{N\backslash d}\mu(\frac{d}{N})\sum_{s\in S}[d\backslash f(s)]$$
$$\sum_{s\in S}[f(s)=N]=\sum_{d\backslash N}\mu(\frac{N}{d})\sum_{s\in S}[f(s)\backslash d]$$
$$\sum_{s\in S}[f(s)=\varnothing]=\sum_{L}(-1)^{|L|}\sum_{s\in S}[L\subseteq f(s)]$$
$$\sum_{s\in S}[f(s)=T]=\sum_{L\subseteq T}(-1)^{|T|-|L|}\sum_{s\in S}[f(s)\subseteq L]$$
$$\sum_{s\in S}[f(s)=T]=\sum_{T\subseteq L}(-1)^{|T|-|L|}\sum_{s\in S}[L\subseteq f(s)]$$
$$\sum_{s\in S}[f(s)=n]=\sum_{k\leq n}(-1)^{n-k}\sum_{s\in S}\binom{k}{f(s)}$$
$$\sum_{s\in S}[f(s)=n]=\sum_{n\leq k}(-1)^{n-k}\sum_{s\in S}\binom{f(s)}{k}$$

References

[1] 贾志鹏,《线性筛法和积性函数》

[2] 贾志鹏,《Topcoder SRM 450-499 Solution》

[3] *Ronald L. Graham, Donald E. Knuth, Oren Patashnik*,《Concrete Mathematics》

[4] *Richard A. Brualdi*,《Introductory Combinatorics》