走进概率的世界

——信息学竞赛中概率问题求解初探

安徽省合肥一中 梅诗珂

摘要

信息学中许多算法的设计都与概率有关。信息学竞赛中求概率或期望的问题也占有相当的分量，并且具有较大的难度。本文应用组合数的性质、误差分析、补集转化和函数分段等方法和技巧，求解了四个例题，从而总结了概率问题的一般特点与对应策略。

关键字

概率，随机变量，连续，离散，概率密度，积分

目录

正文

1 基础知识

1.1 样本空间、事件和概率

样本空间 S 是一个集合，它的元素称为**基本事件**。样本空间的一个子集被称为**事件**，根据定义，所有基本事件互斥。

概率：如果有一种事件到实数的映射 **P{}**，满足：

1） 对任何事件 A，P{A}≥0

2） P{S}=1

3） 对两个互斥事件，P{A∪B}=P{A}+P{B}

则可称 P{A}为事件 A 的概率。上述三条称为概率公理。

1.2 随机变量

如果对样本空间 S 中的任意事件 e，都有**唯一的实数** X(e)与之对应，则称 X=X(e)为样本空间 S 上的**随机变量**，其中**离散型随机变量**与**连续型随机变量**较常见。

1.2.1 离散型随机变量及其概率分布

取值范围为有限或无限可数个实数的随机变量称为**离散型随机变量**。设离散型随机变量 X 取值 x_i 时的概率为 p_k (k=1,2,…),则称 X 的所有取值以及对应概率为 **X 的概率分布**，记做 $P\{X=x_k\} = p_k$ (k=1,2,…)。常见的离散型随机变量的概率分布有两点分布，二项分布，几何分布，超几何分布，泊松分布。

1.2.2 连续型随机变量及其概率分布

如果 X 是在实数域或区间上取连续值的随机变量，设 X 的**概率分布函数**为 F(x)=P{X≤x}，若存在非负可积函数 f(x)，使对任意的 x，有 $F(x)=\int_{-\infty}^{x} f(t)dt$，则称 X 为**连续型随机变量**，称 f(x)为 X 的**概率密度函数**。要注意，概率密度不是概率。常见的连续型随机变

量分布有均匀分布，正态分布，指数分布。

1.2.2.1 连续型随机向量及其概率分布

如果 X1,X2,…,XN 都是连续型随机变量，则称(X1,X2,…,XN)为**N 维随机向量**，其概率分布函数为 F(x1,x2,…,xN)=P{X1≤x1,X2≤x2,…,XN≤xN}。若存在非负可积函数 f(x1,x2,…,xN)使得 $F(x_1,x_2,...,x_N)=\int_D f(x_1,x_2,...,x_N)dx_1dx_2...dx_N$，其中等式右端表示 N 重积分，就称 f(x1,x2,…,xN)是 N 为随机向量(X1,X2,…,XN)的**联合概率密度函数**。如果有 X1,X2,…,XN 互相独立，并且分别有概率密度函数 f1(x1),f2(x2),…,fN(xN)，那么 $f(x_1,x_2,...,x_N)=f_1(x_1)f_2(x_2)...f_N(x_N)$ 就是一个合法的联合概率密度函数。

1.3 数学期望

1.3.1 离散型随机变量的数学期望

设离散型随机变量 X 的分布律为 P{X=xk} = pk（k=1,2,…），若 $\sum_{k=1}^{\infty}|x_kp_k|$ 存在，则称 $\sum_{k=1}^{\infty}x_kp_k$ 为 X 的**数学期望**，简称**期望**，记为 E(X)。

1.3.2 连续型随机变量的数学期望

设连续型随机变量 X 的概率密度函数为 f(x)，若广义积分 $\int_{-\infty}^{+\infty}|xf(x)|dx$ 收敛，则称 $\int_{-\infty}^{+\infty}xf(x)dx$ 为连续型随机变量 X 的数学期望，记为 E(X)。

1.4 积分

积分：设函数 f 在闭区间[a,b]上有定义，记区间[a,b]的一个分割 π 为(x0=a,x1,x2,…,xn=b)(x0<x1<x2<…<xn)，记||π||=Max(xi-x(i-1))（1≤i≤n），我们任取 ξi∈[x(i-1),xi](1≤i≤n)为区间[x(i-1),xi]的代表，如果存在一个数 A，使得对任意小的 ε >0，都存在 δ >0，只要分割 π 满足||π||<δ，都有 $|\sum_{i=1}^{n}f(\xi_i)(x_i-x_{i-1})-A|<\varepsilon$ 而不管 ξi 的取值，我们就称函数 f 在[a,b]上**可积**，称 A 为函数 f 在[a,b]上的**定积分**，记为 $\int_a^b f(x)dx$。一个实用的结论是：任意连续函数在任意闭区间上都是可积的。

Newton-Leibniz 公式：设连续函数 f 在[a,b]上有定义，则 $\int_a^b f(x)dx=F(b)-F(a)$，其中 F(x)是 f(x)的任一个原函数。

下面，我们将分别对离散型随机变量与连续型随机变量两种情况举两个例子进行分析。

2 关于离散型随机变量的问题

关于离散型随机变量的问题，在信息学竞赛中出现过多次了，比如 NOI2005 的《聪聪与可可》，NOI2006 的《神奇口袋》，NOI2008 的《赛程安排》。这类问题对算法设计要求往往不高，只要证明出了关键的定理，问题就被解决了。解决这类问题，就一定要对离散型随机变量的常见分布的性质比较熟练。

下面，我们来看一道关于最大胜率的例题。

2.1 例一：LastMarble[1]

题目描述：

有 red 个红球，blue 个蓝球在一个袋子中。两个玩家轮流从袋子中取球，每个人每次可以取 1，2 或 3 个球，但在他把球拿出袋子之前，他并不知道所取球的颜色。每次球被取出袋子后，它们的颜色被公布给所有人。取走最后一个红球的人输。现在已知有人在游戏开始前取走了 removed 个球，并且谁也不知道球的颜色。在两个玩家都采取最优策略时，先手的胜率是多少？

约束条件： 1≤red,blue≤100 ， 0≤removed≤red-1。

分析：

当 removed=0 的时候，这个问题是很普通的动态规划问题。我们只需设 F(r,b)代表现在剩 r 个红球，b 个蓝球，面对当前局面的玩家所能得到的最大胜率。那么：

$$F(0,b)=1 \quad (\ b\geqslant 0\ )$$

$$F(r,b)=\max_{1\le m\le Min(r+b,3)}\Big(\sum_{0\le p\le r,0\le q\le b,p+q=m}\frac{C_r^pC_b^q}{C_{r+b}^m}(1-F(r-p,b-q))\Big) \quad (1)$$

其中 $\frac{C_r^pC_b^q}{C_{r+b}^m}$ 是取到 p 个红球，q 个蓝球的概率。F(red,blue)就是我们要的答案。为了解决 removed>0 的情况，我们需要

定理一：在 red 个红球，blue 个蓝球中先取 a 个球，再取 b 个球，剩余不同颜色的球数的概率分布与先取 b 个球，再取 a 个球所对应的剩余不同颜色的球数的概率分布是相同的。

[1] TopCoder SRM 349 div one 1000

证明：这个定理看上去很显然，因为先取 a 个球再取 b 个球应当是与直接取(a+b)个球等价的，现在我们用代数方法证明。首先证明 $C_n^a C_{n-a}^b = C_n^{a+b} C_{a+b}^a (a+b \le n)$。因为

$$C_n^a C_{n-a}^b = \frac{n!(n-a)!}{a!(n-a)!b!(n-a-b)!} = \frac{n!}{(a+b)!(n-a-b)!}\frac{(a+b)!}{a!b!} = C_n^{a+b} C_{a+b}^a (a+b \le n)。$$

假设在先取 a 个球，再取 b 个球后，红球被取了 r 个。显然每次取球是互相独立的。所以这种情况的概率是

$$\sum_{a1=0}^{Min(a,r)} \left(\frac{C_{red}^{a1} C_{blue}^{a-a1}}{C_{red+blue}^{a}} \frac{C_{red-a1}^{r-a1} C_{blue-(a-a1)}^{b-(r-a1)}}{C_{red+blue-a}^{b}} \right)$$

$$= \sum_{a1=0}^{Min(a,r)} \left(\frac{C_{red}^{r} C_{blue}^{a+b-r}}{C_{red+blue}^{a+b}} \frac{C_{r}^{a1} C_{a+b-r}^{a-a1}}{C_{a+b}^{a}} \right) \qquad \text{（应用上面的结论）}$$

$$= \frac{C_{red}^{r} C_{blue}^{a+b-r}}{C_{red+blue}^{a+b}}$$

其中 a1 表示第一次取到的红球数，在第二个等式的右边，因为 a1≤a 和 a1≤r 总有一个成立，所以等式恒成立。这个等式告诉我们从 red 个红球与 blue 个蓝球中先取 a 个球，再取 b 个求，与直接取(a+b)个球造成的剩余不同颜色的球数的概率分布是完全相同的。所以取球的顺序与最终的结果没有关系。■

我们设 F(r,b)表示当前有 r 个红球，b 个蓝球的，被事先取走了 removed 个球（不知道它们的颜色），面对这个局面的玩家所能得到的最大胜率。当玩家取走 m 个球时，根据定理一，**新的局面与玩家先取走 m 个球，再让 removed 个球被取走所得到的局面完全一样！**但是有一种边界情况要考虑:由于不能保证 removed≤r，可能在一些情况下，取走 m 个球后，玩家已经输了，而不能进行下面的游戏。要解决这种特殊情况，只需对 F 的定义和动态规划方程略加修改：让 F(r,b)表示当前有 r 个红球，b 个蓝球，被取走了 removed 个球**但仍然至少还有 1 个红球**的情况下，当前玩家的最大胜率。

我们用 Pro(r,b,k)表示有 r 个红球，b 个蓝球，取走 k 个球而红球数仍大于 0 的概率，那么 Pro(r,b,k)= $\sum_{a<r,0\le k-a\le b} \frac{C_r^a C_b^{k-a}}{C_{r+b}^k}$。我们用递推式求解，显然

Pro(0,b,0) = 0;

Pro(r,b,0) = 1; (r>0)

$$\Pr o(r,b,k) = \frac{r}{r+b}\Pr o(r-1,b,k-1) + \frac{b}{r+b}\Pr o(r,b-1,k-1)$$

再考虑(1)，因为 r 个红球，b 个蓝球取走 removed 个球仍有至少 1 个红球的情况，包含了有(r-p)个红球，(b-q)个蓝球，取走 removed 个球后仍有至少 1 个红球的情况，因此在前

者满足的基础上后者满足的概率是 $\frac{\Pr o(r-p,b-q,removed)}{\Pr o(r,b,removed)}$。由此我们得出最终方程。

$$F(r,b)=0 \qquad (r+b = removed)$$

$$F(r,b)=\underset{1\le m\le Min(r+b,3)}{Max}\left(\sum_{p+q=m}\frac{\Pr o(r-p,b-q,removed)}{\Pr o(r,b,removed)}\frac{C_r^pC_b^q}{C_{r+b}^m}(1-F(r-p,b-q))\right)$$

$$(r+b>removed) \qquad (2)$$

F[red,blue]就是我们所求的答案。

观察（2）式，我们发现它与（1）式唯一不同之处只是多乘了 $\frac{\Pr o(r-p,b-q,removed)}{\Pr o(r,b,removed)}$，两个式子的形式惊人的相似，既在意料之外又在情理之中。而如果证明不出定理（1），那么解题方法很可能会复杂很多，这也就说明了数学证明对解决概率问题的重要性。下面一个例子，我们将体会到观察的重要性。

2.2 例二：Randomness[2]

题目描述：

有一个随机数生成器能随机返回 1 到 R 的正整数，现在有 N 个事件，要求第 i 个事件的发生概率是 $\frac{a_i}{b_i}$，用该随机数生成器设计一种事件触发装置，使随机数生成器的期望使用次数 Exp 尽量少，求出 Exp(精确到 1e-7)。

约束条件： 2≤R≤1000 ， 1≤N≤1000 ， 1≤ai≤bi<=1000(1≤i≤N) ， 且 $\sum_{i=1}^{N}\frac{a_i}{b_i}=1$，所有数都是正整数。

分析：

为了观察题目规律，我们先看一个例子：

当 R=100 ， N=3， $\frac{a1}{b1}=\frac{a2}{b2}=\frac{a3}{b3}=\frac{1}{3}$ 时，最优策略可以是：当生成的数 x≤99 时执行

[2] UVA 11429 Problemsetter: Derek Kisman

事件 x mod 3，否则再使用一次随机数生成器，事件的选择与上一次一样。这样，每个事件发生的概率是 $\sum_{i=1}^{\infty}\frac{33}{100^i}=\frac{\frac{33}{100}-0}{1-\frac{1}{100}}=\frac{1}{3}$；

$$\text{Exp}=\sum_{i=0}^{\infty}\frac{1}{100^i}=\frac{1-0}{1-\frac{1}{100}}=\frac{100}{99};$$

我们看到第一次使用生成器(定义为第一层生成器)时，有一些直接映射到事件，其余的每一个返回值分别对应一次新的生成器的使用，定义为第二层生成器。归纳地，定义第 i+1(2≤i)层生成器为由第 i 层生成器返回值引起的新的生成器的使用。

很明显，对事件 i 而言，假设第 k 层生成器中有 D(i,k)个返回值对应着事件 i 的发生，一定有

$$\frac{ai}{bi}=\sum_{k=1}^{\infty}\frac{D(i,k)}{R^k}$$

这里 D(i,k)<R(k=1,2,…)，因为如果某个 D(i,k)>=R，我们把 D(i,k)减去 R，D(i,k-1)加 1，则等式仍成立，而 Exp 减小了。同时可以看出，这就是 $\frac{ai}{bi}$ 的 $\frac{1}{R}$ 进制表示，所以 D(i,k)的取值是唯一的。

为了计算 Exp，我们定义 H(i+1)为第 i 层生成器产生的导致新的生成器使用的返回值个数，定义 H(1)=1，那么 $\frac{H(i)}{R^i}$ 就是第 i 层生成器对 Exp 的贡献，我们有

$$Exp=\sum_{i=1}^{\infty}\frac{H(i)}{R^i}$$

至于求 H(i),因为第 i 层生成器中所有导致新生成器使用的返回值的个数，就是第 i 层生成器被使用的个数，减去其中所有导致事件的返回值个数，也就是

$$H(i+1)=H(i)R-\sum_{k=1}^{N}D(i,k),$$

我们不可能求出所有的 H(i)，也没有必要，因为对所有的 m>=2，我们有

$$\sum_{k=m}^{\infty}\frac{H(k)}{R^k}<\sum_{k=m}^{\infty}\frac{NR}{R^k}=\frac{N}{R^{m-2}(R-1)}<=\frac{N}{R^{m-2}}$$

这个式子告诉我们：**只要 m 足够大，我们可以让 Exp 的误差小到任意程度**，而且这个误差减小的速度是很快的。根据题中要求的精度，只要算到第 50 层就足够了。

回顾本题的分析过程，我们发现开头对例子的观察是关键，有了对该样例的观察我们才得到了第 i 层随机生成器的概念，接下来的一切就水到渠成了。这告诉我们解决概率问题多从简单情况，特殊情况观察是个不错的选择。这是因为概率问题涉及的状态多，比较抽象，只有多从形象的例子观察规律，才能较好的把握问题的模型，才谈得上解决问题。

3 关于连续型随机变量的问题

关于连续型随机变量的问题，信息学竞赛中出现得还不多，可能是由于题目的解决往往要用到积分。但这类题目中积分的技巧一般并不是难点，最多只是用到 Newton-Leibniz 公式而已，所以今后比赛中很有可能出现更多的关于连续型随机变量的问题，解决这类问题的一些思想在其他类型题目中也是有用的。下面我们将结合两道例题谈谈解决关于连续型随机变量的问题的常见思路。

3.1 例三：RNG[3]

题目描述：

有 N 个随机数生成器，第 i 个等概率地返回[0,Ri]中的一个实数(1≤i≤N)。问所有生成数的和小于等于 b 的概率是多少(答案精确到 1e-9)？

约束条件： 1≤N≤10 ， 1≤R≤10(1≤i≤N) ， 0≤b≤100. Ri,b 都是整数。

分析：

我们首先对题意进行简单地分析。第 i 个随机数生成器返回的值对应着一个随机变量，不妨设为 Xi，那么 Xi 在区间[0,Ri]上均匀分布。同时这 N 个随机变量是互相独立的。这些随机变量的和也是一个随机变量，不妨设为 S。我们要求的就是 S 小于等于 b 的概率，记为 P(X≤b)。

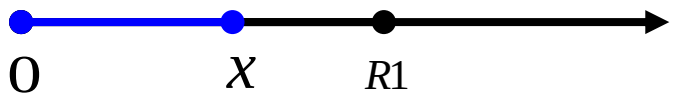
我们将从几何与代数的角度，用两种方法解决问题。

3.1.1 方法一

回忆基础知识部分，N 个随机变量对应着 N 维空间，比较复杂，我们首先从 N 较小的情况观察。

当 N=1 时

我们把 X1 的取值范围[0,R1]看成数轴上的线段，S=X1 ≤x 看成半直线。P(S ≤ x)就是它们的公共部分长度与 R1 的比值。



当 N=2 时

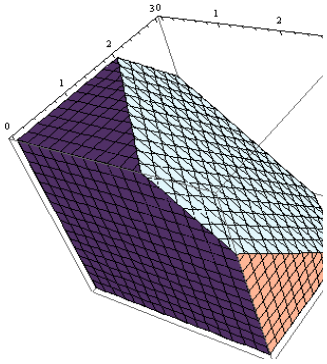
把(X1,X2)的取值范围看成平面直角坐标系的一个矩形；S =X1+X2 ≤x 可以看成半平面；P(S ≤x)就是它们的公共部分的面积与矩形的面积(R1×R2)的比值。

当 N=3 时

(X1,X2,X3)的取值范围可看成空间直角坐标系的长方体；S =X1+X2+X3 ≤x 可以看

[3] SPOJ 2002 Random Number Generator

成半空间；P(S ≤x)就是它们的公共部分的体积，与长方体的体积(R1×R2×R3)的比值。下图是一个例子。



一般的，对任意的 N，(X1,X2,…,XN)的取值范围就是 N 维空间中的一个区域，S ≤ x 是一个半 N 维空间，它们公共部分的 N 维体积与(R1×R2×…×RN)的比值就是 P(S ≤ x)。由于(R1×R2×…×RN)是一个常数，问题就转化为求公共部分的 N 维体积。不妨记 V([a1,b1],[a2,b2],…,[aN,bN],x)为在 Xi 取值范围为[a1,b1]时，它们的和小于 x 的区域的 N 维空间体积。

每个变量的取值范围都是有限的，当 N 较大时，区域有很多边，使情况非常复杂。这时我们想到一个常用技巧：补集转化。

[0,Ri] = [0,+∞) - (Ri ,+∞)

那么 V([0,R1],[0,R2],…,[0,RN],x)就可以用 Xi 取值范围[0,+∞)或[Ri,+∞)时的小于等于 x 的区域 N 维体积表示了。具体如何表示并不是重点，有兴趣的同学可以参看附录，这里不作展开。

现在的问题是当 Xi 的取值范围为[0,+∞)或[Ri,+∞)时的小于等于 x 的区域 N 维体积怎么求，当 Xi 的取值范围为[Ri,+∞)时，用 X’i=Xi-Ri 替换 Xi,同时把 x 减去 Ri，问题等价，而 X’i 的取值范围恒为[0,+∞)。

那么问题归结为求 V([0,+∞),[0,+∞),…, [0,+∞),x),经过简单的观察与证明，可以发现其值为 $\frac{x^N}{N!}$，具体的证明参见附录。

至此问题全部解决。回顾解题过程，我们从观察 N=1,2,3 的情况入手，把问题归结为求公共部分的 N 维体积，接着通过补集转化和化简把问题归为求 V([0,+∞),[0,+∞),…, [0,+∞),x)。通过不断的转化，我们对问题的了解越来越深入。

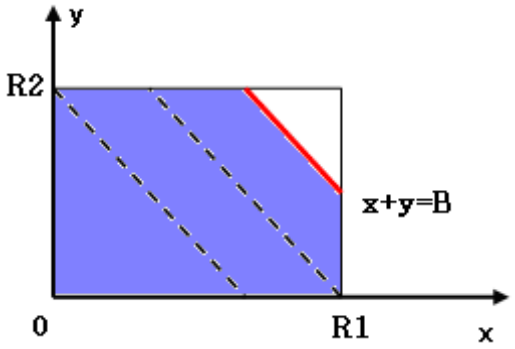
3.1.2 方法二

我们仍然从 N 较小的情况观察。

当 N=1 时，我们考虑 x 在[0,+∞) 变化的情况，P(S≤x)可以表示成函数

$$P(S \le x) = \begin{cases} x/R1 & (\ 0 \le x \le R1\ ) \\ 1 & (\ x > R1\ ) \end{cases}$$

当 N=2 时



如图所示，P(S≤x)就是矩形在直线 x1+x2=x 下的部分(阴影部分)的面积除矩形总面积。为了计算直线 x1+x2=x 下的部分，我们用直线 x+y=R1, x+y=R2 把矩形分成 3 部分，在每部分内直线 x1+x2=x 被矩形所截的线段长随着 x 线性变化。我们对每部分求出 P(S≤x)的表达式

$$P(S\le x)=\begin{cases}\dfrac{x^2}{2R1R2} & (0\leqslant x\leqslant R2)\\ \dfrac{2x-R2}{2R1} & (R2<x\leqslant R1)\\ \dfrac{-x^2+(2R1+2R2)x-(R1^2+R2^2)}{2R1R2} & (R1<x\leqslant R1+R2)\\ 1 & (R1+R2<x)\end{cases}$$

回顾 N=1, N=2 的 P(S≤x)的求解过程，我们发现可以划分若干区间，每个区间的 P(S≤x)可以表示为一个关于 x 的多项式，把 x 带入其所在区间对应的多项式中，得到的就是 P(S≤x)。现在的问题是：如何划分区间？

由于 R1, R2, …, RN 都是整数，我们自然提出了猜想：P(S≤x)在相邻整数区间可以表示成关于 x 的多项式。这个猜想是正确的。下面我们来证明。

证明：

设 $f_i(x)$ 表示前 i 个随机变量的和 Si=X1+X2+…+Xi(也是一个连续型随机变量)的概率密度函数，我们首先证明，$f_i(x)$ 在相邻整数区间能用多项式表示。

用归纳法，当 i=1 时，所求就是 X1 的概率密度函数，由于 X1 在[0, R1]上均匀分布，根据概率密度函数的定义，易得：

$$f_1(x)=\begin{cases}1/R1 & (0\leqslant x\leqslant R1)\\ 0 & (x>R1)\end{cases}$$

显然它们是多项式。

设当 i=2, 3, …, N-1 时猜想都成立，当 i=N 时，因为 X1, X2, …, Xi 是互相独立的，所以前(i-1)个随机变量的和 S_{i-1} 与 Xi 相独立，而 Xi 的概率密度函数就是一个在[0, Ri]上取

$\frac{1}{R_i}$ ，其他位置取 0 的分段函数(跟 $f_1(x)$ 类似),根据联合概率密度函数的定义，有：

$$f_i(x)=\int_{Max(0,x-Ri)}^{x}\frac{1}{Ri}\,f_{i-1}(t)dt$$

任取一个整数区间[L-1，L](L=1,2,…)，考虑 Max(0,x-Ri)=x-Ri 的情况，我们发现可以把积分区间这样划分：[x-Ri,L-Ri],[L-Ri,L-Ri+1],…,[L-2,L-1],[L-1,x]，其中除了第一个与最后一个区间外，所有区间的两端都为常数，它们的积分值自然是常数，对于第一个与最后一个区间，我们用 Newton-Leibniz 公式，由于多项式的原函数仍然是多项式，所以上式右段可以表示成两个多项式和加上若干常数，显然仍是多项式。
如果 Max(0,x-Ri)=0，那么变化只是第一个区间变为[0,1]，上面的结论仍然成立。由于[L-1,L]是任取的，所以 $f_i(x)$ 在 i=N 时在相邻整数区间可以表示成关于 x 的多项式。

根据概率密度函数的定义，有

$$P(S\le x)=\int_0^x f_N(t)dt=\int_0^{\lfloor x\rfloor} f_N(t)dt+\int_{\lfloor x\rfloor}^{x} f_N(t)dt$$

其中右端第 1 项是常数,第 2 项应用 Newton-Leibniz 公式仍得到多项式,所以 $P(S\le x)$ 在相邻整数区间可以表示为关于 x 的多项式。

至此问题全部解决。我们同样从 N 较小的情况入手，提出了 $P(S\le x)$ 在相邻整数区间可以用多项式表示的猜想，并加以证明，证明同时也就得到了算法。在方法二中，观察与大胆猜想是解题的核心。

3.1.3 比较两种方法

在第一中方法中，我们得到的公式相对简单，方法二中我们为了得到 $P(S\le x)$ 需要不断积分，相比复杂一些。同时方法一中的证明相对简单，而方法二证明用到了联合概率密度，相对复杂，抽象。

但方法二有它的优点：可推广性强。这是因为每个随机变量都是均匀分布这个条件对方法二中的证明来说可以减弱：只要每个随机变量的概率密度函数是多项式就行。而方法一中之所以能把概率看成 N 维体积的比值，其根本原因就是每个随机变量都是均匀分布的。

举个简单的例子：如果要求的是 N 个随机数的平方和小于等于 x 的概率，那么方法一将无能为力，而方法二只要简单套用即可。

两种方法都从观察简单情况着手，但第二种方法对题目数学本质了解更透彻，这使方法二成为解决连续型随机变量问题的一般性方法。

下面我们来看一个复杂的例子。

3.2 例四：Random Shooting[4]

题目描述：

一个 N 个顶点的凸多边形位于顶点分别为 C（0，0），D（100，0），E（100，100），F（0，100）的正方形中。在正方形中随机选两个点，问它们至少满足下列条件之一的概率是多少？

条件 1：两个点中有至少一个点在凸多边形内。

条件 2：两个点的连线段与凸多边形相交。

约束条件： 3≤N≤8 ， 凸多边形任意 3 顶点不共线，顶点坐标都是 1 到 99 的正整数。

分析：

我们首先要简化问题。我们设凸多边形的面积为 S，正方形的面积 Total=100*100=10000。那么两点都在凸多边形内的概率

$$P1=(\frac{S}{Total})^2$$

正好有一个点在凸多边形内的概率

$$P2=2(\frac{S}{Total})(1-\frac{S}{Total})$$

再设 $P3=\sum_{i=1}^{N}P(\text{两点连线与凸多边形的第i条边相交})$，

考虑所有与第 i(1≤i≤N)条边相交的两点的集合，它们可以分为两部分：

部分 1:其中一点在多边形外，一点在多边形内。由于多边形是凸的，所以这种情况下，连线段只会与之有唯一的交点（不考虑与多边形的顶点相交的情况，下同），因此对不同的边，部分 1 是不相交的。所以所有边的部分 1 的概率和就是 P2。

部分 2：两点都在多边形外。由于多边形是凸的，这两点连线一定与其中两条边相交，所以对每个属于部分 2 的两点，正好被计算了 2 次。所以所有边的部分 2 的概率和就是 $2P(\text{两点都在凸多边形外且与凸多边形相交})$。

于是我们有 $P3=2P(\text{两点都在凸多边形外且与凸多边形相交})+P2$。

我们要求的是 $P1+P2+\frac{P3-P2}{2}$。

这样问题就转化为求 P3,也就转化为求两点与每条边相交的概率，大大简化了问题。

设当前要计算与边 AB 相交的概率 P，我们让 By≥Ay(如果 Ay≥By,则交换 A,B 点)，这样有向线段 AB 与 x 轴的夹角就在[0,π]中。为什么要这样做呢？因为当两点与 AB 相交时，一定一点在 AB 的左侧，另一点在 AB 的右侧，这样我们只要讨论 AB 左侧的点即可。

[4] SGU 333 Random Shooting

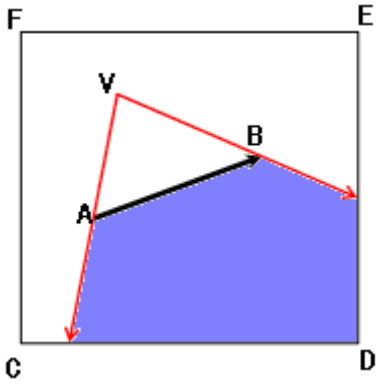


图 1

观察上图，我们发现对固定的点 V(x0, y0)，集合 E={(x, y)|U(x, y)与 V 的连线与 AB 相交}就是图中的阴影部分的点。因为我们可以想象 V 是一个点光源，AB 是一块挡板，那么 V 照不到的地方就是 E，也就是 V 向 A，B 分别作射线，其与正方形的交点，与正方形的顶点还有 A，B 围成的区域。我们设 V(x0, y0)点所得到的阴影部分面积为 $S_V(x0, y0)$，根据连续随机变量概率分布的定义，我们有

$$P = \int_0^{100}\int_0^{100} \frac{S_V(x,y)}{Total} dydx \qquad (1)$$

但是 V 可以取无限多个位置，求出每个点(x, y)的 $S_V(x,y)$ 是不可能的，我们需要进一步观察。

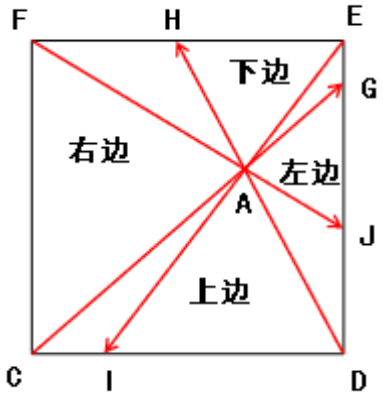


图 2

我们考虑有向线段 AB 中的 A 点，从 C, D, E, F 分别向 A 作射线，与正方形分别交于 G, H, I, J 点，这些点将正方形分成了 8 个区域，就得到了图 2。我们很容易就发现相同区域内所有点向 A 作射线与正方形都交于同一边，比如区域 FAC 中的点向 A 作射线都与边 ED 相交，区域 GAJ 内点向 A 作射线都与边 FC 相交，HAE 中的点向 A 作射线都与边 CD 相交。我们称 CD 为下边，DE 为右边，EF 为上边，FC 为左边，那么同一区域内的点一定与四条边的唯一一条对应。

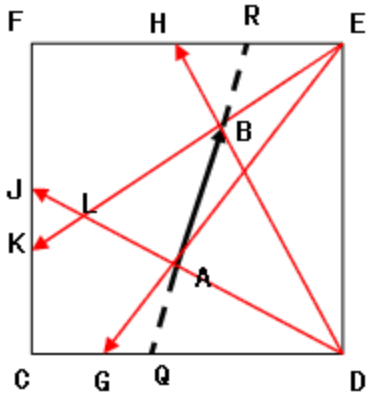


图 3

现在回头考虑 AB。观察图 3，我们把 AB 延长，与正方形交于 QR，根据上面的结论，我们只要考虑 QR 左侧(即四边形 CQRF)即可，从 D,E 分别向 A,B 作射线，他们与正方形的交点把四边形 CQRF 分成了若干区域。相同的区域内点向 A,B 作射线与正方形相交的边是相同的。区域可以按其内的点向 A,B 作射线与正方形相交的边的不同分类。由于区域都在 AB 左侧，那么对任意的点 V，VB 定在 VA 的逆时针方向，所以区域最多只有 10 类：A 左 B 左，A 左 B 下，A 左 B 右，A 左 B 上，A 下 B 下，A 下 B 右，A 下 B 上，A 右 B 右，A 右 B 上，A 上 B 上。

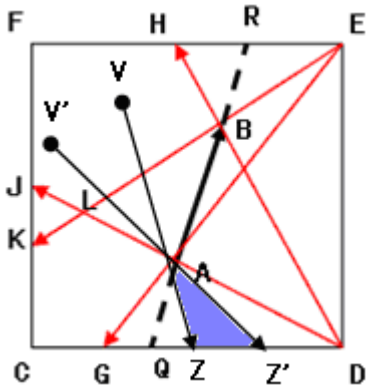


图 4

考虑某一个区域，设其中 VA 与正方形对应边的交点为 Z，当 V 移动到V'的位置时，设新的交点为Z'。如图 4 所示，在 A 这一侧产生的面积的差值就是三角形 ZAZ'的面积（可能是负的），对 VB 也有相同的结论。而 VB 在 VA 的逆时针方向，它们不会相交，也就不会互相干扰。这就告诉我们，对同一个区域中的点 V(x,y)，阴影部分面积$S_V(x,y)$等于某个常数减去两个三角形的面积。我们称这个常数为基本面积，那么就有

$$S_V(x,y) = \text{基本面积} - f(x,y) - g(x,y)$$

其中 f(x,y)与 g(x,y)分别表示点(x,y)在 A 侧与 B 侧的阴影面积与基本面积的出入，它们只与区域有关。

我们看图 4 中的左上角的区域，它属于 A 下 B 右区域，基本面积可以看成多边形 ACDEB（一个凹多边形）的面积，

$$f(x,y)=\frac{1}{2}Ay(Ax-Ay\frac{x-Ax}{y-Ay})=\frac{1}{2}(AxAy-Ay^2\frac{x-Ax}{y-Ay}),$$

$$g(x,y)=\frac{1}{2}(100-Bx)((100-By)-(100-Bx)\frac{y-By}{x-Bx})=\frac{1}{2}((100-Bx)(100-By)-(100-Bx)^2\frac{y-By}{x-Bx}$$

对每个区域我们可以求出基本面积，f 和 g。而 f,g 其实就是某个三角形的面积，因此每个区域$S_V(x,y)$的形式都可以表示为

$$S_V(x,y)=c0+c1\frac{x-x0}{y-y0}+c2\frac{y-y1}{x-x1} \qquad (2),$$

其中 c0,c1,c2,x0,y0,x1,y1 都是常数。把(2)带入（1），得

$$P=\frac{1}{Total}\sum_{区域Q}(c0\iint_{(x,y)\in Q}dxdy+c1\iint_{(x,y)\in Q}\frac{x-x0}{y-y0}dxdy+c2\iint_{(x,y)\in Q}\frac{y-y1}{x-x1}dydx) \qquad (3)$$

为了计算$\iint_{(x,y)\in Q}\frac{y-y1}{x-x1}dydx$，我们把每一个区域用竖直线分割成若干小块，每块都是包含两条竖直边的四边形，不妨设这个四边形的四个顶点为（st，botst），（st，topst），（en，boten），（en，topen），其中 st<en,botst≤topst,boten≤topen。则对该四边形的边(st,botst)，(en,boten)，我们可用 y=a1x+r1 表示；对边（st,topst），(en,topen)可用 y=a2x+r2 表示，其中 a1,a2,r1,r2 为常数。于是

$$\iint_{(x,y)\in Q}\frac{y-y1}{x-x1}dydx=\int_{st}^{en}\int_{a1x+r1}^{a2x+r2}\frac{y-y1}{x-x1}dydx$$

$$=\int_{st}^{en}\frac{ax^2+bx+c}{x-x1}dx$$

$$=\int_{st}^{en}a'x+b'+\frac{c'}{x-x1}dx$$

$$=\frac{a'(en^2-st^2)}{2}+b'(en-st)+c'(\ln(|\,en-x1\,|)-\ln(|\,st-x1\,|))$$

其中 a,b,c,a’,b’,c’都是常数，该区域被分成的四边形的积分值的和就是$\iint_{(x,y)\in Q}\frac{y-y1}{x-x1}dydx$。为了计算$\iint_{(x,y)\in Q}\frac{x-x0}{y-y0}dxdy)$只需把区域用水平线分割成若干包含两条水平边的四边形，用相同方法求解。

至此本问题已获解决。

纵观整个解题过程，正是不断地通过转化，使问题的复杂程度降到可接受的范围，而耐心细致的观察分析则是解决问题的关键，积分技巧的运用并不复杂，也不是难点。这启示我们：解决连续随机变量的概率分布问题，数学功底是基础，观察与分析是关键。

4 总结

本文分别对与离散型随机变量和连续型随机变量有关的问题举了两个例子。第一个例子中，我们灵活应用组合数的性质解决了最大胜率问题；第二个例子中，我们通过定义随机数生成器的层数与误差分析，得到了最优的随机触发装置；第三个例子中，分别应用补集转化和函数分段求解两种方法分别，解决了多个随机变量和的分布问题，并进行了比较；第四个例子中，我们通过对凸多边形的简化和分情况讨论，最终把问题归结为求解一个积分式。四道题的解法各有特色，这正是概率问题复杂多变的表现。但从分析过程中也不难总结出一些通用的原则。四道题无一例外地要求对数学定义与相关知识准确地把握；从简单情况进行分析着手，强调简化问题。这些原则都是由概率问题数学性强、抽象复杂的特点决定的。化繁为简、化抽象为形象，是分析求解概率问题的重要思路。

感谢

感谢刘汝佳老师在论文选题上的指导！
感谢寿鹤鸣同学对论文修改提出的宝贵意见！
感谢集训队的金斌同学对解题提供的帮助！
感谢集训队的汤可因同学的帮助！
感谢唐文斌、胡伟栋教练的指导！

参考文献

1.《算法导论》 [美]*Thomas H.Cormen* 等著
2.《概率论》 何书元著
3.《算法艺术与信息学竞赛》 刘汝佳、黄亮著
4.《数学分析教程》 常庚哲、史济怀著
5. 国家集训队 2005 至 2008 年论文

附录

附录 1 V([0,R1],[0,R2],…,[0,RN],x)的表示。

$$V([0,R1],[0,R2],\cdots[0,RN],x)$$
$$=V([0,+\infty)-(R1,+\infty),[0,+\infty)-(R2,+\infty),\cdots,\ [0,+\infty)-(R2,+\infty),x)$$

设 $D_i \in \{0, R_i\}$ (1≤i≤N)，设集合 $Q \subseteq \{1,2,...,N\}$ 为所有满足 $D_i = R_i$ 下标集合，那么

$$V([0,R1],[0,R2],[0,RN],x)=\sum_{i=0}^{N}(-1)^i \sum_{|Q|=i} V((D_1,+\infty),(D_2,+\infty),...(D_{N,}+\infty),x)$$

该式与容斥原理的公式相近。

附录 2　对 V([0,+∞)，[0,+∞)，…，[0,+∞),x)的值为 $\frac{x^N}{N!}$ 的证明。

证明：

用归纳法，当 N=1 时，V((0,+∞),x) = x 显然符合结论。

设当 N=2,3,…,k-1 时都有结论成立，那么 N=k 时，V([0,+∞)，[0,+∞),…，[0,+∞),x)就是一个 k 维锥体的 k 维体积，椎体的底面面积是 $\frac{x^{k-1}}{(k-1)!}$。同时我们知道体积就是截面面积的积分值，而对与锥体的顶点距离为 h 的截面而言，其截面面积为 $\frac{x^{k-1}}{(k-1)!}(\frac{h}{x})^{k-1}$，所以

$$V([0,+\infty),[0,+\infty),\cdots,[0,+\infty),x) = \int_0^x \frac{x^{k-1}}{(k-1)!}(\frac{h}{x})^{k-1}dh = \frac{1}{(k-1)!}(\frac{x^k}{k}-0) = \frac{x^k}{k!} \blacksquare$$

附录 3 论文原题

LastMarble 题目描述：

Problem Statement

A bag is filled with a given quantity of red and blue marbles. In each turn, a player reaches into the bag and removes 1, 2, or 3 marbles. The player looks to see the color of the marbles, and announces how many of each color marble were removed. The last player to remove a red marble from the bag loses. You are given ints **red** and **blue**, the number of red and blue marbles initially in the bag. Before play begins, a friend removes several of the marbles from the bag, at random, without showing either of you, given in int **removed**.

Assuming you go first, and each player makes the optimal choice of number of marbles to remove, calculate the probability that you win the game.

Definition

Class: LastMarble
Method: winningChance
Parameters: int, int, int
Returns: double
Method signature: double winningChance(int red, int blue, int removed)
(be sure your method is public)

Notes

-Return value must be within 1e-9 absolute or relative error of the actual result.

Constraints

-**red** will be between 1 and 100, inclusive.-**blue** will be between 1 and 100, inclusive.-**removed** will be between 0 and **red** - 1, inclusive.

Randomness 题目描述：

Problem C
Randomness
Input: Standard Input

Output: Standard Output

Efficient generation of "true" random numbers is an interesting problem in Computer Science. The reasonably fast random number generators (RNGs) that are provided in code libraries are usually merely "pseudorandom". They have a finite cycle length and their low-order bits can be predictable to some extent.

Even if the RNGs themselves could be perfect, their common usage is not. For instance, if you have a function that returns a true random integer between 1 and 100 inclusive, and you want to choose between three events, each with probability 1/3, you cannot do it with just one RNG call.

However, if we allow for making more RNG calls if necessary, it is possible to achieve a perfectly uniform distribution over the three events. In fact, any perfect RNG can be used to choose among ANY set of events and associated probabilities with perfect accuracy.

Assume that you have access to a perfect RNG that returns numbers between 1 and R inclusive, and the desired probabilities of N disjoint events. You must determine an algorithm that uses the RNG to choose among the events with exactly the correct probability, while minimizing the expected (average) number of RNG calls.

Input

Input will consist of at most 32 cases, each consisting of two lines. The first line will contain two integers, R and N, satisfying $2 \leqslant R \leqslant 1000$ and $2 \leqslant N \leqslant 1000$. The second line will contain N pairs of integers a_i and b_i, with $1 <= a_i < b_i <= 1000$. The desired probability of event i is a_i / b_i. The sum of the N probabilities will be 1.

Input will be terminated by a line containing two zeros.

Output

For each case, output, rounded to six fractional digits, the minimum expected number of random number generator calls required to decide among the events.

RNG 题目描述：

2002. Random Number Generator

Problem code: RNG

LoadingTime got a RNG (*Random Number Generator*) from his classmate several weeks ago. And he spent a lot of time study it. He found that RNG can generate a real number in range [-S,S] by executing following steps. First RNG generates n integer $X_1..X_n$, the sum of which is equal to S. Then for each X_i, it generates a real number in range

$[-X_i, X_i]$ randomly. The output (a real number) of RNG will be the sum of the **N** generated real numbers. LoadingTime noticed that the distribution of the output was very interesting, and he wanted to know: for given **N** and **X**, what's the probability that the generated number is in range [**A**,**B**]. Could you help him?

Input

The first line contains an integer T representing the number of test cases.

For each test case, the first line contains three integers **N**, **A**, **B**($1 \leqslant N \leqslant 10$, $-100 \leqslant A \leqslant B \leqslant 100$) In the second line of the test case, you are given $X_1 \ldots X_n$($1 \leqslant X_i \leqslant 10$).

Output

For each test case, print a line contains a real number representing the probablity as the problem required. It must be printed with exactly nine decimal places.

Example

Input:
```
5
1 -100 100
10
1 10 90
10
1 -20 5
10
2 -20 5
5 5
5 -5 10
1 2 3 4 5
```

Output:
```
1.000000000
0.000000000
0.750000000
0.875000000
0.864720052
```

Random Shooting 题目描述：

333. Random Shooting

Time limit per test: 1 second(s)

Memory limit: 65536 kilobytes
input: standard
output: standard

You're playing a new game at the shooting range. First, a target is placed on a square board. The target itself is a convex polygon strictly inside the board.

Then, you're allowed to shoot twice. In case at least one of your shots gets inside the target, you win. Moreover, if the segment connecting the points of your shots intersects the target, you still win. If none of the above holds, you lose.

Assuming your aiming is very bad (i.e., the points of your shots are independently uniformly distributed all over the board), compute the probability of you winning.

Input

The first line of input contains an integer N, $3 \leqslant N \leqslant 8$, — the number of vertices of the target. The next N lines contain two integers each, x_i and y_i, $1 \leqslant x_i, y_i \leqslant 99$, decribing the coordinates of the vertices of the target (a convex polygon) in the counter-clockwise direction. No three vertices of the target lie on the same line. The coordinate system is chosen so that the corners of the board have coordinates (0, 0), (0, 100), (100, 0) and (100, 100).

Output

Output the required probability. Your solution will be accepted if it is within 10^{-7} of the correct one.

Example(s)

sample input	sample output
4 25 25 75 25 75 75 25 75	0.7328341830