Available online at www.sciencedirect.com

ScienceDirect

Journal of Discrete Algorithms 5 (2007) 341–347

JOURNAL OF DISCRETE ALGORITHMS

www.elsevier.com/locate/jda

HyperQuick algorithm for discrete hypergeometric distribution

Aleš Berkopec

Faculty of Electrical Engineering, University of Ljubljana, Slovenia

Received 22 April 2005; accepted 24 January 2006

Available online 14 July 2006

Abstract

Based on the binomial identity

$$\sum_{k=0}^{x}\binom{M}{k}\binom{N-M}{n-k}=\sum_{m=M}^{N-n+x}\binom{m}{x}\binom{N-1-m}{N-m-n+x}$$

we present an algorithm for computing the cumulative distribution function of a random variable with discrete hypergeometric distribution. For any accuracy $\epsilon \geqslant 0$ the required number of computational cycles is less then $N-n$, where N is the size of the population and n is the size of the sample. In this article we prove the binomial identity above and give the formula for the number of computational cycles required to achieve the desired accuracy for an arbitrary set of parameters.

Keywords: Binomial identities; Hypergeometric distribution

1. Introduction

A hypergeometric distribution is used in samplings without replacement [1,2]. Consider the following problem: Given N balls, M of which are black and the rest are white, what is the probability $C(n, x, N, M)$ that out of n balls selected uniformly at random without replacement, at most x are black? Clearly the answer is

$$C(n,x,N,M)=\frac{\sum_{j=0}^{x}\binom{M}{j}\binom{N-M}{n-j}}{\binom{N}{n}}. \tag{1}$$

The computation of (1) is a time consuming process and involves evaluations of binomial coefficients with a high risk of numeric overflow. In this paper we present an algorithm that on one hand minimizes the risk of numeric overflow, and on the other hand computes (1) fast for values up to 10^8.

E-mail address: ales.berkopec@fe.uni-lj.si.

1570-8667/$ – see front matter
doi:10.1016/j.jda.2006.01.001

2. A binomial identity

The algorithm contains two computational loops from the implementation of the binomial identity involving $\widetilde{C}(n,x,N,M) = 1 - C(n,x,N,M)$, or

$$\underbrace{\frac{\sum_{k=0}^{x}\binom{M}{k}\binom{N-M}{n-k}}{\binom{N}{n}}}_{C(n,x,N,M)} + \underbrace{\frac{\sum_{m=x}^{M-1}\binom{m}{x}\binom{N-1-m}{N-m-n+x}}{\binom{N}{n}}}_{\widetilde{C}(n,x,N,M)} = 1.$$

Theorem 1. *Let x, n, M, N be integers such that $0 \leqslant x \leqslant n$ and $0 \leqslant M \leqslant N$. Then*

$$\sum_{k=0}^{x}\binom{M}{k}\binom{N-M}{n-k} = \sum_{m=M}^{N-n+x}\binom{m}{x}\binom{N-1-m}{N-m-n+x}. \tag{2}$$

Proof. Denote the left- and right-hand sides of (2) by $a(N,n)$ and $b(N,n)$, respectively. By Pascal's identity [3],

$$\binom{N-M}{n-k} + \binom{N-M}{n-k+1} = \binom{N-M+1}{n-k+1}.$$

Multiplying (3) by $\binom{M}{k}$ and summing on k from 0 to x yields

$$a(N+1,n+1) = a(N,n+1) + a(N,n). \tag{3}$$

Similarly, by Pascal's identity,

$$\binom{N-m-1}{N-m-n+x-1} + \binom{N-m-1}{N-m-n+x} = \binom{N-m}{N-m-n+x}.$$

Multiplying (4) by $\binom{m}{x}$ and summing on m from M to $N-n+x$ yields

$$b(N+1,n+1) = b(N,n+1) + b(N,n), \tag{4}$$

therefore $a(N,n)$ and $b(N,n)$ satisfy the same recurrence. Since

$$a(0,n) = b(0,n) = \begin{cases} 1, & n = 0, \\ 0, & n > 0 \end{cases} \tag{5}$$

and $a(N,0) = b(N,0) = 1$, it follows by induction on N that $a(N,n) = b(N,n)$, as claimed. □

3. An algorithm

The parameters n, N and x are constant. We set

$$p_m = p_{m,N-1}^{n-1}(x) = \frac{\binom{m}{x}\binom{N-1-m}{N-m-n+x}}{\binom{N-1}{n-1}}$$

and

$$J_m = \frac{p_{m+1}}{p_m} = \frac{1 - \frac{n-1-x}{N-1-m}}{1 - \frac{x}{m+1}}. \tag{6}$$

Obviously $J_m > 0 \Leftrightarrow (x < m) \wedge (n < N)$. The case $J_m = 0 \Leftrightarrow (x = m) \wedge (n = N)$ is trivial and is not considered here. With J_m and $M_0 = N - n + x$ we write

$$\widetilde{C}(n,x,N,M) = \frac{p_{M-1}(1 + \frac{1}{J_{M-2}} \cdot (1 + \frac{1}{J_{M-3}} \cdot (1 + \cdots \frac{1}{J_x})))}{p_{M_0}(1 + \frac{1}{J_{M_0-1}} \cdot (1 + \frac{1}{J_{M_0-2}} \cdot (1 + \cdots \frac{1}{J_x})))}. \tag{7}$$

Because of

$$p_{M_0} = p_{M-1} \cdot \prod_{i=M-1}^{M_0-1} J_i,$$

it follows that

$$\widetilde{C}(n, x, N, M) = \frac{\frac{1}{J_{M_0-1}} \cdot \frac{1}{J_{M_0-2}} \cdots \frac{1}{J_M} \cdot \frac{1}{J_{M-1}} \cdot s}{(1 + \frac{1}{J_{M_0-1}} \cdot (1 + \frac{1}{J_{M_0-2}} \cdot (1 + \cdots \frac{1}{J_M}(1 + \frac{1}{J_{M-1}} \cdot s)))}, \tag{8}$$

where

$$s = 1 + \frac{1}{J_{M-2}} \cdot \left(1 + \frac{1}{J_{M-3}} \cdots \frac{1}{J_{x+1}}\left(1 + \frac{1}{J_x}\right)\right).$$

Let $\tilde{c}_k$ be a finite sequence of approximations $\tilde{c}_k = \tilde{c}_k(n, x, N, M)$ for $k \in [0, M_0 - M]$:

$$\tilde{c}_k = \tilde{c}_k(n, x, N, M) = \frac{s \cdot \frac{1}{J_{M-1}} \cdot \frac{1}{J_M} \cdots \frac{1}{J_{M-1+k}}}{(((s \cdot \frac{1}{J_{M-1}} + 1) \cdot \frac{1}{J_M} + 1) \cdots + 1) \cdot \frac{1}{J_{M-1+k}} + 1}, \tag{9}$$

and thus $\widetilde{C}(n, x, N, M) = \tilde{c}_{M_0-M}$. By defining $c_k = 1 - \tilde{c}_k$, Eq. (1) can be written as

$$C(n, x, N, M) = c_{M_0-M}.$$

The algorithm is the implementation of Eq. (9) and $c_k = 1 - \tilde{c}_k$. It consists of the computation of s that requires a fixed $M - 1 - x$ number of steps and the variable part implemented with the while loop.

For $M \leqslant N$, $x \leqslant n$, $n \leqslant N$, $x, n, N, M \geqslant 0$ and function $J_m(n, x, N, m)$ HyperQuick algorithm for computing (1) is:

HyperQuick algorithm

```
 1: function InvJm(n,x,N,m):double;
 2:     InvJm:=(1-x/(m+1))/(1-(n-1-x)/(N-1-m))
 3: end function
 4: #
 5: begin
 6: # loop1: fixed number of steps
 7: s:=1.0;
 8: for k:=x to M-2 do
 9:     s:=s*InvJm(n,x,N,k)+1.0;
10: end for
11: # loop2: variable number of steps according to accuracy e
12: ak:=s;
13: bk:=s;
14: k:=M-2;
15: epsk:=2*e;
16: while (k<N-(n-x)-1) and (epsk>e) do
17:     ck:=ak/bk;
18:     k:=k+1;
19:     jjm:=InvJm(n,x,N,k);
20:     bk:=bk*jjm+1.0;
21:     ak:=ak*jjm;
22:     epsk:=(N-(n-x)-1-k)*(ck-ak/bk);
23: end while
24: result:=1-(ak/bk-epsk/2);
25: end
```

4. Convergence and error estimation

Theorem 2. *The successive approximations $\tilde{c}_k$ form a decreasing finite sequence.*

Proof. Let $\psi_k = ((s \cdot \frac{1}{J_{M-1}} + 1) \cdots + 1) \cdot \frac{1}{J_{M-1+k}} + 1$, therefore

$$\frac{\tilde{c}_{k+1}}{\tilde{c}_k} = \frac{1}{1 + \frac{J_{M+k}}{\psi_k}}.$$

Since $J_m > 0$ and $\psi_k > 0$, we find $\frac{\tilde{c}_{k+1}}{\tilde{c}_k} < 1$. □

Theorem 3. *The difference between the successive approximations $\tilde{c}_k$ is decreasing with increasing k*:

$$\frac{\tilde{c}_{k+1} - \tilde{c}_k}{\tilde{c}_k - \tilde{c}_{k-1}} < 1.$$

Proof.

$$\frac{\tilde{c}_{k+1} - \tilde{c}_k}{\tilde{c}_k - \tilde{c}_{k-1}} = \frac{J_{M+k}}{J_{M-1+k}} \cdot \frac{\psi_{k-1}}{\psi_k + J_{M+k}}.$$

It is possible to show that each of the factors is smaller than 1. For the right one:

$$\psi_k = \frac{\psi_{k-1}}{J_{M-1+k}} + 1 \Rightarrow \psi_{k-1} < \psi_k,$$

$$\psi_{k-1} < \psi_k \wedge J_{M-1+k} > 0 \Rightarrow \frac{\psi_{k-1}}{\psi_k + J_{M+k}} < 1.$$

We transform the left factor into a more suitable form:

$$\frac{J_{y+1}}{J_y} = \frac{1 - \frac{n-1-x}{N-2-y}}{1 - \frac{n-1-x}{N-1-y}}.$$

Since $\frac{n-1-x}{N-2-y} > \frac{n-1-x}{N-1-y}$, we get

$$\frac{J_{y+1}}{J_y} < 1,$$

and for both factors

$$\left(\frac{\psi_{k-1}}{\psi_k + J_{M+k}} < 1\right) \wedge \left(\frac{J_{M+k}}{J_{M-1+k}} < 1\right) \Rightarrow \frac{\tilde{c}_{k+1} - \tilde{c}_k}{\tilde{c}_k - \tilde{c}_{k-1}} < 1. \qquad \square$$

The successive approximations $\tilde{c}_k$ approach the value $\widetilde{C}(n, x, N, M)$ from the above: $\tilde{c}_k - \widetilde{C}(n, x, N, M) \geqslant 0$.

Theorem 4. *The value of $\widetilde{C}(n, x, N, M)$ is bounded with*

$$\widetilde{C}(n, x, N, M) \in [\tilde{c}_k - \epsilon_k, \tilde{c}_k],$$

where

$$\epsilon_k = (M_0 - M - k) \cdot |\tilde{c}_k - \tilde{c}_{k-1}|.$$

Proof. We put the approximate value in the middle of the interval

$$\tilde{c}_k - \frac{\epsilon_k}{2} \leqslant \widetilde{C}(n, x, N, M) \leqslant \tilde{c}_k + \frac{\epsilon_k}{2}. \tag{10}$$

From Theorems 2 and 3 it can be seen that ϵ_k decreases with increasing k, therefore

$$\left|\tilde{c}_k - \widetilde{C}(n,x,N,M)\right| < \left|\tilde{c}_{k-1} - \widetilde{C}(n,x,N,M)\right| \quad \forall k > 0.$$

Since

$$(J_m < J_{m-1}) \wedge (\lim_{m\to\infty} J_m = 1) \Rightarrow J_{m<\infty} > 1 \Rightarrow J_m > 1,$$

we set

$$\alpha_k = \prod_{i=M-1}^{M+k-2} \frac{1}{J_i} < \frac{k}{J_{M+k-2}}$$

and

$$\beta_k = \left(\left(\cdots\left(\frac{1}{J_{M-1}} + \frac{1}{s}\right)\cdot\frac{1}{J_M} + \frac{1}{s}\right)\cdot\frac{1}{J_{M+1}} + \cdots + \frac{1}{s}\right)\cdot\frac{1}{J_{M+k-2}} + \frac{1}{s} < \frac{k+1}{J_{M-1}},$$

where $s > \frac{M-x-1}{J_x}$,

$$\tilde{c}_k = \frac{\alpha_k}{\beta_k},$$

$$\tilde{c}_{k-1} = \frac{\alpha_k \cdot J_{M+k-1}}{(\beta_k - 1/s)\cdot J_{M+k-1}} = \frac{\alpha_k}{\beta_k - 1/s},$$

and finally

$$|\tilde{c}_k - \tilde{c}_{k-1}| = \frac{\alpha_k}{\beta_k}\left[1 - \frac{1}{1 - 1/(s\beta_k)}\right] < \frac{\alpha_k}{s\beta_k^2}.$$

The error $\epsilon_k = (M_0 - M - k)|\tilde{c}_k - \tilde{c}_{k-1}|$ can be estimated with

$$\epsilon_k < \left(\frac{M_0 - M}{k} - 1\right)\cdot\frac{J_N J_{M-1}^2}{M - x - 1}. \qquad \square \tag{11}$$

Theorem 5. *The number of the steps $K(\epsilon)$ required to achieve the accuracy $\epsilon \geqslant 0$ is less than $N - n$:*

$$K(\epsilon) < N - n.$$

Proof. The number of the steps $K(\epsilon)$ required to achieve the desired accuracy ϵ is the sum of the fixed part (computation of s) and the variable part (Eq. (11))

$$K(\epsilon) < \underbrace{M - x - 1}_{\text{fixed part}} + \underbrace{\frac{M_0 - M}{1 + \frac{M-x-1}{J_x J_{M-1}^2}\epsilon}}_{\text{from Eq. (11)}}. \tag{12}$$

Less than $K(\epsilon = 0)$ steps are needed to achieve any accuracy greater than zero. Taking into account $M_0 = N - n + x$, this means less than

$$K(\epsilon = 0) < M - x - 1 + M_0 - M = N - n - 1$$

steps, or $K(\epsilon) < N - n$. $\square$

For more information about our algorithm see Tables 1–3.

Acknowledgements

The author would like to thank Mr. Andrej Kores, Ljubljana, Slovenia, for the presentation of the problem that had lead to the development of HyperQuick algorithm. Dr. Tomaž Jarm and Dr. Borut Jurčič-Zlobec, Faculty of Electrical Engineering, University of Ljubljana, Slovenia, crosschecked the numerical results.

Table 1
The results and computational times. The computations were performed on PC Intel (R) Pentium (R) IV, CPU 3.00 GHz. The accuracy of HyperQuick algorithm is $\epsilon = 10^{-16}$

n	x	N	M	C	Time [s]
10	5	100	50	0.62966677311277	0.001
100	50	1000	500	0.54194604604641	0.001
1000	500	10000	5000	0.51329471498065	0.002
10000	5000	100000	50000	0.50420511457274	0.007
100000	50000	1000000	500000	0.50132980423988	0.052
1000000	500000	10000000	5000000	0.50042052198071	0.480
10000000	5000000	100000000	50000000	0.50013298075677	4.719

Table 2
The number of the successive approximations K required for the computation of $C(n, x, N, M)$ with the accuracy ϵ_k for $n = 2000$, $x = 1000$, $N = 20000$, $M = 10000$. The fixed part of K is $M - x - 1 = 8999$ (Eq. (12))

ϵ_k	K	C
10^{-1}	$8999 + 631$	0.5584
10^{-2}	$8999 + 777$	0.5143
10^{-3}	$8999 + 900$	0.5099
10^{-4}	$8999 + 1008$	0.5095
10^{-5}	$8999 + 1105$	0.5094
10^{-6}	$8999 + 1194$	0.5094
10^{-7}	$8999 + 1276$	0.5094

Table 3
Comparison of the computational times of HyperQuick on the Pentium (R) 4 CPU 3.20 GHz computer with the freely available packages. *R* package computes $C(n, x, N, M)$ in less than 10 milliseconds for every set of parameters above. HyperQuick computes $C(n, x, N, M)$ faster than other three packages, but only NCSS Probability Calculator is slower than HyperQuick by several orders of magnitude

Package	$C(n, x, N, M)$	t [s]
$n = 1000$, $x = 500$, $N = 10000$, $M = 5000$		
NCSS Probability Calculator	0.5132947150	7.1
Smith's statistical package	0.51329471	<0.001
StatCalc 1.1	0.513295	<0.001
HyperQuick	0.513294714980647	<0.001
R	0.513294714980647	<0.010
$n = 10000$, $x = 5000$, $N = 100000$, $M = 50000$		
NCSS Probability Calculator	–	>100
Smith's statistical package	0.50420511	<0.001
StatCalc 1.1	0.504205	<0.001
HyperQuick	0.504205114572739	<0.001
R	0.504205114572738	<0.010
$n = 1000000$, $x = 500000$, $N = 10000000$, $M = 5000000$		
NCSS Probability Calculator	–	$\gg$100
Smith's statistical package	0.50042047	1.0
StatCalc 1.1	0.500420	0.5
HyperQuick	0.500420521980705	0.3
R	0.500420521980698	<0.010
$n = 10000000$, $x = 5000000$, $N = 100000000$, $M = 50000000$		
NCSS Probability Calculator	–	$\gg$100
Smith's statistical package	0.50013207	10.2
StatCalc 1.1	0.50013	3.4
HyperQuick	0.500132980756772	2.7
R	0.500132980756751	<0.010

References

[1] M. Petkovšek, H. Wilf, D. Zeilberger, $A = B$, A K Peters Ltd., 1996.
[2] R.L. Graham, D.E. Knuth, O. Patashnik, Concrete Mathematics: A Foundation for Computer Science, Addison-Wesley, 1989.
[3] J. Riordan, Combinatorial Identities, John Wiley & Sons, 1968.