Counting the number of Sudoku's by importance sampling simulation

Ad Ridder
Vrije University, Amsterdam, Netherlands

May 31, 2013

Abstract

Stochastic simulation can be applied to estimate the number of feasible solutions in a combinatorial problem. This idea will be illustrated to count the number of possible Sudoku grids. It will be argued why this becomes a rare-event simulation, and how an importance sampling algorithm resolves this difficulty.

1 Introduction

The *Sudoku puzzle* was introduced in the Netherlands as late as 2005, but became quickly very popular. Nowadays most newspapers publish daily versions of various difficulties. The idea of the puzzle is extremely simple as can be seen from the following figures. Consider a 9×9 grid, divided into nine 3×3 blocks, where some of the boxes contain a digit in the range $1, 2, \ldots, 9$, for instance

6	2							1
	8			2				
4				3	1			8
2	7							
			9		7			
							9	3
5			7	8				6
				4			3	
3							5	7

As can be seen, there are no duplicates in any row, column or block. The problem is to complete the grid by filling the remaining boxes with 1–9 digits, while keeping the property of no duplicates in any row, column, or block. The solution is a *Sudoku grid*, in this case:

6	2	3	8	7	9	5	4	1
7	8	1	4	2	5	3	6	9
4	9	5	6	3	1	2	7	8
2	7	9	3	1	4	6	8	5
8	3	6	9	5	7	1	2	4
1	5	4	2	6	8	7	9	3
5	4	2	7	8	3	9	1	6
9	5	7	5	4	6	8	3	2
3	6	8	1	9	2	4	5	7

In this paper we are interested in the number of possible Sudoku grids. This is a combinatorial problem that might be solved by complete enumeration [6]. However, we shall show that a simple stochastic simulation approach gives an extreme good approximation, within a few percent of the exact number.

The simulation is based on importance sampling and this idea can be used also for counting problems where one does not know the exact number. Counting problems are important in certain areas of computer science [8], also from a theoretical point of view of algorithmic complexity. Many counting problems have no exact polynomial-time algorithm [7], and then one searches for polynomial-time approximation algorithms which are fast and give accurate approximations. A popular technique is to get estimates by some Markov chain Monte Carlo method [9, 12]. Recently, counting problems have been considered as rare-event problems [1, 2, 11], and then one might apply importance sampling techniques for getting fast and accurate estimates. Our paper is in this spirit.

2 Counting by simulation

Let Ω be a finite population of objects and $A \subset \Omega$ a set of interest, defined by some property of its elements. Denote by $|\Omega|$ and $|A|$ their sizes. We assume the following.

Assumption 2.1.

1. *The population size $|\Omega|$ is known (but large), the size $|A|$ of the set of interest is unknown.*

2. *It is easy to generate samples $\omega \in \Omega$ from the uniform probability distribution u on Ω.*

3. *It is easy to test whether a sample $\omega \in \Omega$ is an element of the target set A or not.*

Here 'easy' means that the associated algorithms run in polynomial time. We consider (Ω, u) as a finite probability space, and let $X : \Omega \to \Omega$ be the identity map with induced probability P. Thus for any set $B \subset \Omega$

$$\mathbb{P}(X \in B) = u(B) = \frac{|B|}{\Omega}.$$

Specifically for the set of interest we get

$$|A| = \mathbb{P}(X \in A)|\Omega| = u(A)|\Omega|.$$

Now we can see a simulation approach for estimating $|A|$: generate $X_1, \ldots, X_n$ as n i.i.d. copies of X, and use $\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{X_i \in A\}$ as an unbiased estimator of $\mathbb{P}(X \in A)$. More importantly,

$$Y(n) \doteq |\Omega|\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{X_i \in A\} \tag{1}$$

is an unbiased estimator of $|A|$, that is commonly known as the direct, or the crude Monte Carlo estimator. The performance of this estimator is measured through the sample size n required to obtain 5% relative error with 95% confidence [3], i.e.,

$$\mathbb{P}\big(|\, Y(n) - |A|\,| < 0.05|A|\big) \geq 0.95. \tag{2}$$

Assuming n large enough to approximate $Y(n) \overset{\text{d}}{\sim} N(|A|, \mathbb{V}ar[Y(n)])$ with

$$\mathbb{V}ar[Y(n)] = |\Omega|^2 \tfrac{1}{n} u(A)(1-u(A)) = |A|\,|\Omega|\tfrac{1}{n}(1-u(A)),$$

and letting $z_{1-\alpha/2} = 1.96 \approx 2$ to be the 95% two-sided critical value (or percentile) of the standard normal distribution, we get that (2) holds if and only if

$$2\sqrt{\mathbb{V}ar[Y(n)]} \leq 0.05|A| \;\Leftrightarrow\; n \geq \left(\frac{2}{0.05}\right)^2 \frac{|\Omega|}{|A|}(1-u(A)) = 1600\frac{1-u(A)}{u(A)}. \tag{3}$$

Equivalently, the performance criterion (2) holds if and only if the *relative error* $\text{RE}[Y(n)] \leq 2.5\%$, where a relative error of an estimator is defined to be the ratio of its standard deviation to its mean.

For our Sudoku problem we consider the population Ω to be all 9×9 grids for which each row is a permutation of the digits 1–9. We call such grids *9-permutation grids*, and it is easy to see that there are

$$|\Omega| = (9!)^9 \approx 1.091 \cdot 10^{50}$$

9-permutation grids, for example

3	1	2	8	6	5	4	9	7
6	3	7	5	9	2	4	1	8
1	8	3	4	6	5	9	2	7
4	9	5	3	7	6	2	8	1
2	3	9	7	4	8	6	1	5
2	9	7	3	5	1	6	4	8
1	4	2	8	7	6	9	3	5
3	6	9	8	4	2	1	5	7
5	6	4	3	7	9	1	2	8

Notice that also the items 2. and 3. of Assumption 2.1 are satisfied. However, the direct simulation algorithm fails to work: an experiment that generated 100 million 9-permutation grids, did not found any Sudoku grid among them. The reason is that the set A of Sudoku grids is a *rare event* in the 'world' of 9-permutation grids, i.e., the probability that a randomly generated 9-permutation grid turns out to be a Sudoku grid, is extremely small. We shall see later that this probability is $u(A) = |A|/|\Omega| \approx 6.114 \cdot 10^{-29}$. Hence, the required sample size (3) is

$$n \approx \frac{1600}{u(A)} \approx 2.5 \cdot 10^{31},$$

which would take about $8{\cdot}10^{18}$ years on a typical home PC (10^5 random 9-permutation grids were generated per second). Notice that the total number of calls of the random number generator would be about $2.5 \cdot 10^{31} \cdot 72 = 1.8 \cdot 10^{33}$ which is nowadays no problem [4].

2.1 Importance sampling

Suppose the simulation is executed by generating random 9-permutation grids according to a probability q on Ω. For each $\omega \in \Omega$, let $L(\omega) \doteq u(\omega)/q(\omega)$ be its associated *likelihood ratio*. Then it is easy to see that the corresponding (unbiased) importance sampling estimator of $|A|$ becomes

$$Y_q(n) \doteq |\Omega| \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{X_i \in A\} L(X_i). \tag{4}$$

What we do actually is estimating the *rare-event probability* $\mathbb{P}(X \in A)$ by importance sampling. Recently, there is much interest in techniques and methods concerning rare events and how to estimate their probabilities, both from a theoretical point of view, and from a practical vue point, see [3, 10] and the more popular [13].

The idea of importance sampling is to let the rare event to occur more often in such a way that the likelihood ratio does not blow up. The main issue, therefore, is to find the new probability (or *change of measure*) q such that the importance sampling estimator is efficient or optimal. Clearly, an estimator is *optimal* if its variance is zero. In that case a single sample suffices! However, in practice this is not realizable. More realistic is to achieve an efficient importance sampling estimator. Let $\hat{\gamma}_q(n) \doteq Y_q(n)/|\Omega|$ be the associated estimator of the probability $\mathbb{P}(X \in A)$, then we say that it is *efficient* if

$$\mathrm{RAT}[\hat{\gamma}_q(n)] \doteq \frac{\log \mathbb{E}[\hat{\gamma}_q(n)^2]}{\log \mathbb{E}[\hat{\gamma}_q(n)]} \approx 2. \tag{5}$$

Basically it means that the required sample size to obtain (3) grows polynomially, when the estimated probability decays exponentially fast to zero [5]. It is easy to show that the crude Monte Carlo estimator has RAT = 1, and that always RAT ≤ 2.

Comparison of the performances of the crude Monte Carlo and the importance sampling estimators will be given by considering their relative errors (RE, defined above) and their efficiency EFF defined by

$$\mathrm{EFF}[\hat{\gamma}_q(n)] \doteq 1/(\mathbb{V}ar[\hat{\gamma}_q(n)] \times \mathrm{CPU}[\hat{\gamma}_q(n)]),$$

where CPU stands for the computation time. Better performance is obtained by smaller RE, larger RAT, and higher EFF.

3 Importance sampling for generating Sudoku grids

We propose the following change of measure for our Sudoku problem.

1. Start with an empty grid.
2. Simulate the grid row-by-row from top to bottom.
3. Simulate a row box-by-box from left to right.
4. Suppose that box (i, j) (row i, column j) has to get a digit, and that it lies in the k-th 3×3-block (numbered left-to-right and top-to-bottom). Then, eliminate from $\{1, \ldots, 9\}$ all digits that have already been generated (i) in the boxes of row i (i.e., left of j), (ii) in the boxes of column j (i.e., above i), and (iii) in the boxes of the k-th 3×3-block (i.e., in the rows above i). Let R_{ij} be the remaining digits.
5. If R_{ij} is empty, generate in all remaining boxes random digits 1–9 similarly as under the uniform measure. The grid will not be a Sudoku grid.
6. If R_{ij} is non-empty, box (i, j) gets a digit generated uniformly from R_{ij}. Repeat from item 4. with the next box until all boxes are done.

Example 3.1.
Suppose that the grid is generated up to box $(i, j) = (3, 5)$ with

5	8	9	6	1	7	3	2	4
2	1	4	9	5	3	6	7	8
7	3	6	4					

Under the original uniform measure, box $(3, 5)$ would get a digit uniformly drawn from $\{1, 2, 5, 8, 9\}$. However, under the change of measure, box $(3, 5)$ gets a digit from $\{2, 8\}$ (uniformly).

The likelihood ratio $L(\omega) = u(\omega)/q(\omega)$ is a product of the likelihood ratios of the boxes:

$$L(\omega) = \prod_{\substack{(i,j)\\ i=1,\dots,9\\ j=1,\dots,9}} \ell_{ij},$$

where the box likelihood ratio is

$$\ell_{ij} = \frac{u_{ij}}{q_{ij}} = \begin{cases} \frac{1/(10-j)}{1/|R_{ij}|} & \text{if } R_{ij} \neq \emptyset \\ 1 & \text{otherwise.} \end{cases}$$

While executing the importance sampling algorithm, we calculate the likelihood ratio by updating the product after each newly generated digit. For instance in Example 3.1, the current likelihood ratio is (clearly, the box likelihood ratios of the first row are all equal to 1):

$$\underbrace{\left(\left[\tfrac{6}{9}\ \tfrac{5}{8}\ \tfrac{4}{7}\right]\left[\tfrac{4}{6}\ \tfrac{3}{5}\ \tfrac{2}{4}\right]\left[\tfrac{3}{3}\ \tfrac{2}{2}\ \tfrac{1}{1}\right]\right)}_{\text{row 2}}\ \underbrace{\left(\left[\tfrac{3}{9}\ \tfrac{2}{8}\ \tfrac{1}{7}\right]\left[\tfrac{3}{6}\right]\right)}_{\text{row 3}}.$$

Box $(3, 5)$ will get either 2 or 8, both with likelihood ratio $\ell_{35} = \frac{2}{5}$, which is going to be multiplied with the expression above.

In this way we generated 10^6 grids in about 15 seconds resulting in the following performance (we repeated this experiment 100 times and took the averages).

estimate	RE	RAT	EFF
$6.662 \cdot 10^{21}$	3.051%	1.895	$1.941 \cdot 10^{58}$

The average relative error of these hundred estimates to the actual value was 2.1%. The standard estimator (1) would have estimated performance after 10^6 simulated grids

estimate	RE	RAT	EFF
-	$1.266 \cdot 10^{13}$%	1.0	$1.041 \cdot 10^{33}$

4 Conclusion

We have shown how importance sampling simulation is applicable to approximate counting problems in combinatorial problems, and we gave an illustration for counting the number of possible 3×3 Sudoku grids. The performance of the estimator indicates that the importance sampling algorithm is efficient. However, when applied to larger Sudoku grids, this algorithm seems to be susceptible to the same difficulty that it does not generate quickly feasible Sudoku grids. Currently, we investigate importance sampling algorithms based on splitting the sample space and apply Gibbs sampling for generating samples.

References

[1] Bayati, M., Kim, J., and Saberi, A. 2010. A sequential algorithm for generating random graphs. *Algorithmica*, 58, pp. 860-910.

[2] Blanchet, J., and Rudoy, D. 2009. Rare event simulation and counting problems. Chapter 8 in *Rare event simulation using Monte Carlo methods*, eds. Rubino G. and Tuffin, B., Wiley, pp. 171-192

[3] Bucklew, J.A. 2004. *Introduction to Rare Event Simulation*. Springer, New York.

[4] L'Ecuyer, P. 2006. Random Number Generation, chapter 3 in *Elsevier Handbooks in Operations Research and Management Science: Simulation*, eds. S. G. Henderson and B. L. Nelson, Elsevier Science, Amsterdam, pp. 55-81.

[5] L'Ecuyer, P., Blanchet, J.H., Tuffin, B., and Glynn, P.W. 2010. Asymptotic robustness of estimators in rare-event simulation, *ACM Transactions on Modeling and Computer Simulation*, 20(1), article 6.

[6] Felgenhauer, B., and Javis, F. 2005. Enumerating possible Sudoku grids. Available at
`http://www.afjarvis.staff.shef.ac.uk/sudoku/sudoku.pdf`

[7] Jerrum, M. 2001. *Counting, sampling and integrating: algorithms and complexity*. Birkhäuser.

[8] Meer, K. 2000. Counting problems over the reals. *Theoretical Computer Science* 242, pp. 41-58.

[9] Mitzenmacher, M., and Upfal, E. 2005. *Probability and computing: randomized algorithms and probabilistic analysis*. Cambridge University Press.

[10] Rubino, G., and Tuffin, B. (Eds.). 2009. *Rare event simulation using Monte Carlo methods*. Wiley.

[11] Rubinstein, R.Y. 2007. How many needles are in the haystack, or how to solve fast #P-complete counting problems. *Methodology and Computing in Applied Probability* 11, pp. 5-49.

[12] Rubinstein, R.Y., Ridder, A., Vaisman, R. 2013. *Fast Sequential Monte Carlo Methods for Counting and Optimization*. Wiley.

[13] Taleb, N.N. 2007. *The black swan: the impact of the highly improbable*. Random House.