Approximate Counting by Dynamic Programming

Martin Dyer[*]
School of Computing
University of Leeds
Leeds LS2 9JT, UK.
dyer@comp.leeds.ac.uk

ABSTRACT

We give efficient algorithms to sample uniformly, and count approximately, the solutions to a zero-one knapsack problem. The algorithm is based on using dynamic programming to provide a *deterministic* relative approximation. Then "dart throwing" techniques are used to give arbitrary approximation ratios. We also indicate how further improvements can be obtained using randomized rounding. We extend the approach to several related problems: the m-constraint zero-one knapsack, the general integer knapsack (including its m-constraint version) and contingency tables with constantly many rows.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms; G.3 [**Mathematics of Computing**]: Probability.

General Terms

Algorithms, Theory.

1. INTRODUCTION

In this paper we describe efficient algorithms to sample uniformly, and count approximately, solutions to the zero-one knapsack problem and some related problems. For definitions and background on polynomial time sampling and approximate counting see, for example, the monograph of Jerrum [13].

Specifically we address both the single and multiple constraint versions of zero-one knapsack, the general integer knapsack with arbitrary upper bounds (both single and multiple constraint), and contingency tables with a constant number of rows. In each case the algorithms are based on a dynamic programming computation which provides a *deterministic* approximation ratio of polynomial size. Then simple "dart throwing" techniques can be used to boost this to arbitrarily good approximation ratios.

Previous approaches to the problems we discuss here have been based almost exclusively on the Markov chain Monte Carlo (MCMC) approach. One exception is the algorithm of Cryan and Dyer [2] for contingency tables with constantly many rows. This combines dynamic programming with volume approximation, but the approximate volume computation does itself involve MCMC methods.

The *zero-one knapsack problem* has been approached by MCMC. The best result known, due to Morris and Sinclair [16, 17], gives sampling in time $O(n^{9/2+\epsilon})$, for any $\epsilon > 0$, for a problem with n variables. In section 2.1 we give an $O(n^3)$ time sampling algorithm and a fully polynomial randomized approximation scheme (*fpras*), with relative error ε, running in time $O(n^3 + \varepsilon^{-2}n^2)$, i.e. essentially the same time bound. In section 2.2 we show how this can be improved further to $O(n^{5/2}\sqrt{\log(n\varepsilon^{-1})} + \varepsilon^{-2}n^2)$ using randomized rounding. Similar improvements seem possible in the other problems we consider, but we will not explore them here.

The (multiple constraint) *multidimensional knapsack* has also been considered previously [8, 15, 17]. Here the best result known, again due to Morris and Sinclair [15, 17], gives sampling with time bound $n^{2^{O(m)}}$. We improve this substantially in section 2.3 to give sampling, and an *fpras*, with a time bound of $O(n^{2m+1})$.

The *general integer knapsack* problem has perhaps been less studied from the viewpoint of approximate counting. The analysis of [8] was extended to this case, but the time bound for the Markov chain given there is $2^{O^*(\sqrt{n})}$ when there are n variables. It is likely that the methods of [17] apply to this problem, but it seems that this has not yet been done. In section 2.4 we show that the single constraint problem has an $O(n^5)$ sampling algorithm and an *fpras* with similar running time. Again, these results generalise easily to give running time $O(n^{2m+3})$ for m constraints.

Sampling and counting *contingency tables* have been studied intensively. See [1, 2, 3, 4, 6, 7, 9, 10, 11, 14, 15, 18], for example. The practical relevance of this problem was discussed by Diaconis and Efron [5]. It is still not known how to sample general contingency tables uniformly in polynomial time, but algorithms are known for some special cases. In particular, the case where the number of rows is considered to be a *constant* has been examined recently. See, for example, [2, 3, 9, 11]. The previous best result known for this problem was the algorithm of [2], which gives sampling in time $n^{O(m^2)}$ when there are m rows and n columns. In

[*]Supported by the EPSRC grant "Sharper Analysis of Randomized Algorithms: A Computational Approach" and by the EC IST Thematic Network RAND-APX.

section 3 we describe a substantial improvement, resulting in $O(n^{4m+1})$ sampling time, and give an *fpras* with similar time bound.

2. COUNTING KNAPSACK SOLUTIONS

We consider sampling and approximate counting for several variants of the knapsack problem.

The following standard notation is used throughout the paper. The set of integers is denoted by $\mathbb{Z}$, the *non-negative* integers by $\mathbb{N}$, and the positive integers by $\mathbb{Z}_+$. The reals are denoted by $\mathbb{R}$. For any $i, j \in \mathbb{N}$ with $i \le j$, we will denote by $[i, j]$ the set of integers $\{i, \dots, j\}$, and we will denote by $[j]$ the set $[1, j]$ for any $j \in \mathbb{Z}_+$.

2.1 The zero-one knapsack

Let $B_n = \{0,1\}^n$, and let S denote the set

$$S = \big\{x \in B_n : \sum_{j=1}^n a_j x_j \le b\big\},$$

where $0 \le a_1 \le a_2 \le \cdots \le a_n \le b$ are integers.[1]

Let k be such that $a_j \le b/n$ for $j \le k$ and either $k = n$ or $a_{k+1} > b/n$. Let $C = \{0,1\}^k \times \{0\}^{n-k}$. If $x \in C$ then $\sum_{j=1}^n a_j x_j \le \sum_{j=1}^k a_j \le kb/n \le b$, so $x \in S$. Thus $C \subseteq S$.

Let $\alpha_j = \lfloor n^2 a_j / b \rfloor$ and $\delta_j = n^2 a_j/b - \alpha_j$, so $0 \le \delta_j < 1$. Let S' be the solution set of

$$\sum_{j=1}^n \alpha_j x_j \le n^2, \quad \text{with } x \in B_n.$$

Now $|S'|$ can be determined in $O(n^3)$ time, using dynamic programming. Write $F(r,s) = |\{x \in B_r : \sum_{j=1}^r \alpha_j x_j \le s\}|$. In $O(n^3)$ time, the dynamic programming tabulates $F(r,s)$ $(1 \le r \le n,\ 0 \le s \le n^2)$, using the recursion

$$\begin{aligned} F(1,s) &= \begin{cases} 1 & \text{if } s < \alpha_1 \\ 2 & \text{otherwise} \end{cases} \\ F(r,s) &= F(r-1,s) + F(r-1, s-\alpha_r) \quad (r \ge 2). \end{aligned}$$

Then we have $|S'| = F(n, n^2)$.

If $x \in S$, $\sum_{j=1}^n \alpha_j x_j \le (n^2/b) \sum_{j=1}^n a_j x_j \le (n^2/b)b = n^2$, so $x \in S'$. Thus $S \subseteq S'$ and $|S| \le |S'|$. If $S' \ne S$, suppose $x \in S' \setminus S$. Then clearly there exists an integer $p = p(x)$ such that $x_p = 1$ and $p \notin [k]$. Otherwise $x \in C \subseteq S \subseteq S'$, a contradiction. If there is more than one such integer, take $p(x)$ to be the smallest. Note that we have $\alpha_p \ge n$.

Define a map $f : S' \to B_n$, as follows. If $x \in S$ then $f(x) = x$. Otherwise $x \in S' \setminus S$, and $p(x)$ is well defined. Define $f(x) = y$, where $y_j = x_j$ for $j \ne p(x)$, and $y_p = 0$. For any $x \in S' \setminus S$, with $y = f(x)$, we have

$$\begin{aligned} \sum_{j=1}^n a_j y_j &= \frac{b}{n^2} \sum_{j=1}^n (\alpha_j + \delta_j) y_j \\ &= \frac{b}{n^2} \Big(\sum_{j=1}^n \alpha_j y_j + \sum_{j=1}^n \delta_j y_j \Big) \\ &= \frac{b}{n^2} \Big(\sum_{j=1}^n \alpha_j x_j - \alpha_p + \sum_{j=1}^n \delta_j y_j \Big) \\ &\le \frac{b}{n^2} (n^2 - n + n) \\ &= b, \end{aligned}$$

so $f(x) \in S$. Hence $f(S') = S$. But, for $y \in S$, we have $|f^{-1}(y)| \le (n+1)$, since any element of $f^{-1}(y)$ may change a single coordinate of y or none. Thus

$$|S'| = |f^{-1}(S)| \le (n+1)|S|.$$

Hence $1 \le |S'|/|S| \le (n+1)$ so $|S'|/\sqrt{n+1}$ approximates $|S|$ deterministically within a factor $\sqrt{n+1}$ and can be computed in $O(n^3)$ time. Since 0-1 knapsack is self-reducible, existence of an *fpras* for the problem now follows indirectly from a general result of Sinclair and Jerrum [19]. However, we will now describe a simpler and more efficient "dart-throwing" method to construct an *fpras* directly.

The $F(r,s)$ table can be used to determine a uniform point in S' in $O(n)$ time, by tracing back probabilistically from $F(n,n^2)$, as follows. With probability $F(n-1,n^2)/F(n,n^2)$ set $x_n = 0$, else set $x_n = 1$ with the remaining probability $F(n-1, n^2 - \alpha_n)/F(n,n^2)$. If $x_n = 0$, recursively determine $x_{n-1}, x_{n-2}, \dots, x_2, x_1$ by tracing back from $F(n-1,n^2)$ and, if $x_n = 1$, trace back similarly from $F(n-1, n^2 - \alpha_n)$. The resulting point of S' has probability at least $1/(n+1)$ of lying in S. If so, it is uniformly distributed in S, and we accept it. Otherwise we repeat the whole process independently. After $n+1$ repetitions we have a sample with probability at least $1 - e^{-1}$. Hence a sample of ν uniform points in S can be determined in $O(n^3 + n^2\nu)$ time[2] with probability at least $1 - e^{-\Omega(n)}$.

To have an *fpras* for $|S|$, we need only estimate the probability $\rho = |S|/|S'| \ge 1/(n+1)$, since $|S'| = F(n,n^2)$. With ν points in S', the sampling error is $O(1/\sqrt{\nu n})$. We require this to be smaller than $\varepsilon\rho = \Omega(\varepsilon/n)$. Hence we need $\nu = O(\varepsilon^{-2} n)$. The complexity of the *fpras* is then $O(n^3 + \varepsilon^{-2} n^2)$.

2.2 Randomized rounding

We will show how to reduce the running time of the *fpras* for the zero-one knapsack problem by (almost) a $\sqrt{n}$ factor, using randomized rounding. Let ε be as defined in section 1 and used in section 2.1, $K = \sqrt{2n \ln(n/\varepsilon)}$, $\hat{\alpha}_j = \lfloor 2nKa_j/b \rfloor$ and $\delta_j = 2nKa_j/b - \hat{\alpha}_j$. Now let W_j be independent random variables such that

$$\Pr(W_j = 1) = 1 - \Pr(W_j = 0) = \delta_j \qquad (j \in [n]),$$

and let $\alpha_j = \hat{\alpha}_j + W_j$. Hence $2nKa_j/b = \alpha_j + \delta_j - W_j$, and $\mathbf{E}[\alpha_j] = 2nKa_j/b$. Let S be defined as above and let $S' = \{x \in B_n : \sum_{j=1}^n \alpha_j x_j \le (2n+1)K\}$. Then, for any given values of the random variables W_j, $|S'|$ can be

[1]A single (rational) linear inequality in zero-one variables can always be put in this form.

[2]Here and elsewhere we count arithmetic operations, rather than operations on bits.

determined by dynamic programming as before. The improvement in the running time will come from performing this dynamic programming calculation with a smaller right hand side. Note that the α_j ($j \in [n]$) are random variables, and hence S' is a random set, but S is fixed. For any fixed $x \in B_n$, let χ_x be the indicator random variable of the event $\{x \in S'\}$, and $\bar{\chi}_x = 1 - \chi_x$. Let $\Delta_x = \sum_{j=1}^n (W_j - \delta_j)x_j$.

Lemma 1. *For fixed $x \in B_n$ and $\gamma \in \mathbb{R}$, $\gamma > 0$, we have* $\Pr(\Delta_x > \gamma K) \le (\varepsilon/n)^{(2\gamma)^2}$, $\Pr(\Delta_x < -\gamma K) \le (\varepsilon/n)^{(2\gamma)^2}$.

Proof. Since $W_j x_j \in \{0,1\}$ and $\mathbf{E}[W_j x_j] = \delta_j x_j$, by an inequality of Hoeffding [12], we have

$$\Pr(\Delta_x > \gamma K) \le e^{-2\gamma^2 K^2/n} = (\varepsilon/n)^{(2\gamma)^2}.$$

The other inequality is proved identically. □

We will first show that

Lemma 2. $\Pr(\,|S' \cap S| < (1 - \varepsilon^3/n^3)|S|\,) \le \varepsilon/n$.

Proof. Given $x \in S$, we first bound $\mathbf{E}[\bar{\chi}_x] = \Pr(x \notin S')$.

$$\begin{aligned} \sum_{j=1}^n \alpha_j x_j &= \sum_{j=1}^n (2nKa_j/b - \delta_j + W_j)x_j \\ &\le 2nK + \Delta_x \\ &\le (2n+1)K, \end{aligned}$$

provided $\Delta_x \le K$. So, for $x \in S$,

$$\mathbf{E}[\bar{\chi}_x] \;=\; \Pr(x \notin S') \;\le\; \Pr(\Delta_x > K) \;\le\; (\varepsilon/n)^4,$$

by Lemma 1. Thus

$$\mathbf{E}[\,|S \setminus S'|\,] \;=\; \sum_{x \in S} \mathbf{E}[\bar{\chi}_x] \;\le\; \varepsilon^4 |S|/n^4.$$

Hence, using the Markov inequality,

$$\begin{aligned} \Pr(|S' \cap S| < (1 - \varepsilon^3/n^3)|S|) &= \Pr(|S \setminus S'| > \varepsilon^3|S|/n^3) \\ &\le \frac{\varepsilon^4|S|/n^4}{\varepsilon^3|S|/n^3} \\ &= \frac{\varepsilon}{n}. \end{aligned}$$

□

Define $f : B_n \to B_n$ as follows. If $x \in S$, set $f(x) = x$. If $x \notin S$, let $k = \arg\max_j a_j x_j$, so $a_k x_k > b/n$. Now define $y = f(x)$ by setting $y_k = 0$ as before. For each $x \in B_n$, define

$$d_x = \min\{d : f^d(x) \in S\},$$

and let

$$S_d = \{x \in B_n : d_x = d\} \qquad (0 \le d \le n).$$

Observe that d_x is the shortest edge-distance from x to S in the hypercube B_n. Note also that the sets S_d are defined without any reference to random variables. Clearly $S_0 = S$, and $S_1 = f^{-1}(S) \setminus S$. Using an argument similar to that used to bound $|f^{-1}(S)|$ in section 2.1, it follows that

$$|S_d| \le \tbinom{n}{d}|S| \qquad (d \ge 0).$$

Also, $x \in S_d$ implies

$$\sum_{j=1}^n a_j x_j > (1 + (d-1)/n)b \qquad (d > 0).$$

Lemma 3. $\Pr(|S'| > (n+2)|S|) \le \varepsilon/n$.

Proof. Note that $x \in S'$ if and only if

$$\begin{aligned} (2n+1)K &\ge \sum_{j=1}^n \alpha_j x_j \\ &= \sum_{j=1}^n (2nKa_j/b + W_j - \delta_j)x_j \\ &= \frac{2nK}{b}\sum_{j=1}^n a_j x_j + \Delta_x. \end{aligned}$$

Since $x \in S_d$ implies $\sum_{j=1}^n a_j x_j > (1+(d-1)/n)b$, it follows that $x \in S' \cap S_d$ implies

$$\Delta_x \le -(2d-3)K \qquad (d > 1).$$

So, if $x \in S_d$ ($d > 1$),

$$\Pr(x \in S') \;\le\; \Pr(\Delta_x \le -(2d-3)K).$$

Thus, if $x \in S_d$ ($d > 1$), we have

$$\mathbf{E}[\chi_x] \;=\; \Pr(x \in S') \;\le\; (\varepsilon/n)^{(4d-6)^2},$$

using Lemma 1 again. Hence, for $d > 1$,

$$\begin{aligned} \mathbf{E}[\,|S' \cap S_d|\,] &= \sum_{x \in S_d} \mathbf{E}[\chi_x] \\ &\le (\varepsilon/n)^{(4d-6)^2}\tbinom{n}{d}|S| \\ &\le (\varepsilon^2/n)^d |S|/d!. \end{aligned}$$

Let $S'_2 = S' \cap \bigcup_{d=2}^n S_d$. Therefore,

$$\mathbf{E}[\,|S'_2|\,] \;\le\; |S|\sum_{d=2}^n \frac{(\varepsilon^2/n)^d}{d!} \;\le\; (\varepsilon^2/n)^2|S|.$$

Now, again using the Markov inequality,

$$\Pr(|S'_2| > \varepsilon^3|S|/n) \;\le\; \frac{(\varepsilon^2/n)^2|S|}{\varepsilon^3|S|/n} \;=\; \varepsilon/n.$$

Thus, with probability at least $(1 - \varepsilon/n)$,

$$\begin{aligned} |S'| &\le |S_0| + |S_1| + |S'_2| \\ &\le |S| + n|S| + \varepsilon^3|S|/n \\ &< (n+2)|S|. \end{aligned}$$

□

Thus we have $|S'| < (n+2)|S|$ and $|S' \cap S| \ge (1 - \varepsilon^3/n^3)|S|$ with probability at least $(1 - 2\varepsilon/n)$. Hence, with the same probability, for all $n > 2$,

$$\rho = \frac{|S \cap S'|}{|S'|} \ge \frac{(1 - \varepsilon^3/n^3)|S|}{(n+2)|S|} \ge \frac{1}{2n}.$$

Now, by sampling from S', we can determine $\hat{\rho}$ satisfying $\rho\sqrt{1-\varepsilon} \le \hat{\rho} \le \rho/\sqrt{1-\varepsilon}$ in $O(n^2/\varepsilon^2)$ time. (We need $O(n/\varepsilon^2)$ samples, each requiring $O(n)$ time.)

Now let $\Psi = \hat{\rho}|S'|/\sqrt{1-\varepsilon}$ be our estimate of $|S|$. We have

$$\begin{aligned} \Psi &\le \frac{\rho|S'|}{1-\varepsilon} = \frac{|S' \cap S|}{1-\varepsilon} \le \frac{|S|}{1-\varepsilon}, \\ \text{and}\quad \Psi &\ge \rho|S'| = |S' \cap S| > (1-\varepsilon)|S|. \end{aligned}$$

If $\Pr\left(\rho\sqrt{1-\varepsilon} \le \hat{\rho} \le \rho/\sqrt{1-\varepsilon}\right) = 1 - \eta$, the total failure probability is at most $(\eta + 2\varepsilon/n)$, the latter term arising from the errors associated with Lemmas 2 and 3. So we have an *fpras*, with overall running time $O(n^{5/2}\sqrt{\log(n/\varepsilon)} + n^2/\varepsilon^2)$.

2.3 The multidimensional knapsack

The multidimensional (zero-one) knapsack problem,

$$S = \bigcap_{i=1}^{m} S_i, \quad \text{where} \quad S_i = \{x \in B_n : \sum_{j=1}^{n} a_{ij} x_j \le b_i\}, \qquad (1)$$

with $a_{ij} \ge 0$ ($i \in [m]$, $j \in [n]$)[3], can be solved by the same technique if the number of constraints m is a constant. Let $S' = \bigcap_{i=1}^{m} S'_i$, where $S'_i = \{x \in B_n : \sum_{j=1}^{n} \alpha_{ij} x_j \le n^2\}$ with $\alpha_{ij} = \lfloor n^2 a_{ij}/b_i \rfloor$. We can show $S \subseteq S'$ exactly as before.

Let $K_i = \{j : a_{ij} \le b_i/n\}$. For $x \in S' \setminus S$, let $I(x) = \{i : x \in S'_i \setminus S_i\}$. As before, for every $i \in I(x)$, there exists $p_i(x) \notin K_i$ such that $x_{p_i} = 1$. Construct $f(x) = y$ by $y_{p_i(x)} = 0$ for $i \in I(x)$ and $y_j = x_j$ otherwise. Then it can be shown as before that $f(x) \in S$. The inverse mapping changes some set of coordinates P with $0 \le |P| \le m$, so

$$|f^{-1}(y)| \le 1 + n + \binom{n}{2} + \cdots + \binom{n}{m} \le n^m \qquad (m, n \ge 2),$$

and therefore we have $|S'| \le n^m |S|$. The dynamic programming computation to determine $|S'|$ takes $O(n^{2m+1})$ time. Using the same ideas as in section 2.1, we can obtain a uniform sample of size ν from S in time $O(n^{2m+1} + n^{m+1}\nu)$, and an *fpras* for approximate counting which takes $O(n^{2m+1} + \varepsilon^{-2} n^{m+1})$ time.

2.4 The general integer knapsack

Let $U_r = \{\ 0 \le x_j \le u_j,\ j \in [r]\}$, where the u_j are given integers. We want to estimate $|S|$, where

$$S = \{x : \sum_{j=1}^{n} a_j x_j \le b,\ x \in U_n\},$$

with $a_1, \ldots, a_n, b > 0$ given integers. Note that we can assume $u_j \le \lfloor b/a_j \rfloor$. Let

$$\begin{aligned} h_j(x_j) &= \lfloor 2n^2 a_j x_j / b \rfloor \quad (0 \le x_j \le u_j,\ j \in [n]), \\ \text{and} \quad S' &= \{x : \sum_{j=1}^{n} h_j(x_j) \le 2n^2,\ x \in U_n\}. \end{aligned}$$

Now let $C = \{x : a_j x_j \le b/n,\ j \in [n]\}$. It follows easily that $C \subseteq S \subseteq S'$. Thus, if $x \in S' \setminus S$, there exists $p = p(x)$ such that $a_p x_p > b/n$. Note that $h_p(x_p) \ge 2n$. Define $f : S' \to U_n$, by $f(x) = x$ if $x \in S$ and $f(x) = y$ otherwise, where $y_j = x_j$ for $j \ne p(x)$ and $y_p = \lfloor x_p/2 \rfloor$. Now, if $y = f(x)$,

$$\begin{aligned} \sum_{j=1}^{n} a_j y_j &= \frac{b}{2n^2} \sum_{j=1}^{n} 2n^2 a_j y_j / b \\ &= \frac{b}{2n^2} \Big(\sum_{j \ne p} 2n^2 a_j x_j / b + 2n^2 a_p \lfloor x_p/2 \rfloor / b \Big) \\ &\le \frac{b}{2n^2} \Big(\sum_{j \ne p} (h_j(x_j) + 1) + n^2 a_p x_p / b \Big) \\ &\le \frac{b}{2n^2} \Big(\sum_{j \ne p} h_j(x_j) + n - 1 + \tfrac{1}{2}(h_p(x_p) + 1) \Big) \\ &\le \frac{b}{2n^2} \Big(\sum_{j \ne p} h_j(x_j) + n - 1 + h_p(x_p) - n + \tfrac{1}{2} \Big) \\ &\le \frac{b}{2n^2} \Big(2n^2 - \tfrac{1}{2} \Big) < b. \end{aligned}$$

Thus $f(S') = S$. But $|f^{-1}(y)| \le 2n + 1$, since $y \in f^{-1}(y)$ and, for any $1 \le p \le n$, there are at most two possible values of x_p.

We calculate $F(r, s) = |\{x \in U_r : \sum_{j=1}^{r} h_j(x_j) \le s\}|$ by dynamic programming, with $|S'| = F(n, 2n^2)$. Let

$$\kappa_j = \frac{b}{2n^2 a_j}, \quad \tau_j = \lfloor u_j / \kappa_j \rfloor \le 2n^2.$$

Let $\Delta_j(t) = |\{x_j : h_j(x_j) = t\}|$. Then

$$\begin{aligned} \Delta_j(t) &= \lceil (t+1)\kappa_j \rceil - \lceil t \kappa_j \rceil \quad (0 \le t < \tau_j), \\ \Delta_j(\tau_j) &= u_j + 1 - \lceil \tau_j \kappa_j \rceil. \end{aligned}$$

Now the recurrence is

$$\begin{aligned} F(r, s) &= \sum_{t=0}^{\tau_r} \Delta_r(t) F(r-1, s-t), \\ F(1, s) &= 1 + \min\big[\lfloor s\kappa_1 \rfloor, u_1\big]. \end{aligned}$$

The table $F(r, s)$ ($1 \le r \le n,\ 0 \le s \le 2n^2$) can be determined in $O(n^5)$ time. The probabilistic traceback takes $O(n^3)$ time, so we can generate a sample of size ν from S in $O(n^5 + n^4\nu)$ time. Again we need $O(\varepsilon^{-2} n)$ samples from S' for an *fpras*, giving $O(n^5 + \varepsilon^{-2} n^4)$ time. The generalisation to the m-constraint version is similar to the zero-one case, and leads to $O(n^{2m+3} + \nu n^{m+3})$ time for a sample of size ν, and $O(n^{2m+3} + \varepsilon^{-2} n^{m+3})$ time for an *fpras*.

3. CONTINGENCY TABLES

We denote the i,jth element of a matrix $x \in \mathbb{N}^{m \times n}$ by x_{ij}, and its jth column by x_j ($i \in [m], j \in [n]$).

A *contingency table* with row sums $r = (r_1, \ldots r_m)$ and column sums $c = (c_1, \ldots c_n)$ is any $x \in \mathbb{N}^{m \times n}$ such that $\sum_{j=1}^{n} x_{ij} = r_i$ ($i \in [m]$) and $\sum_{i=1}^{m} x_{ij} = c_j$ ($j \in [n]$). Note that, if $N = \sum_{i=1}^{m} r_i$ then also $N = \sum_{j=1}^{n} c_j$. For given row and column sums, we wish to estimate the number of distinct tables. We consider the case where m is considered to be a constant. We may assume $m \le n$, otherwise we can transpose the table. We will assume, without loss of generality, that $c_n = \max_{j=1}^{n} c_j$. We also assume $c_n \ge m^5$. Otherwise, it is not difficult to see that we may count *exactly* by dynamic programming in $O(n^{m+1})$ time.

[3]The problem is unlikely to have an *fpras* without some such assumption, even if $b_i > 0$ for all $i \in [m]$. See the appendix.

For $x \in \mathbb{N}^{m\times n}$, let $x^* \in \mathbb{N}^{m\times(n-1)}$ denote x with its nth column deleted. Let

$$X_j = \big\{x_j \in \mathbb{N}^m \ : \ \sum_{i=1}^m x_{ij} = c_j\big\} \qquad (j \in [n-1]),$$

and

$$X = \big\{x^* \ : \ x_j \in X_j,\ j \in [n-1]\big\}.$$

Letting $r = (r_1, \ldots, r_m)$, the set S of contingency tables with totals r_i, c_j can be written

$$S = \big\{x^* \in X \ : \ \sum_{j=1}^{n-1} x_j \le r\big\}.$$

Let $h_j : X_j \to \mathbb{N}^m$ be defined by

$$[h_j(x_j)]_i = \lfloor 2n^2 x_{ij}/r_i \rfloor \quad (i \in [m]),$$

and let

$$S' = \big\{x^* \in X \ : \ \sum_{j=1}^{n-1} h_j(x_j) \le 2n^2\mathbf{1}\big\},$$

where $\mathbf{1}$ is the m-vector of 1's. Clearly $S \subseteq S'$.

For $t \in T = [0, 2n^2]^m$, we can calculate

$$F(k,t) = |\{(x_1, \ldots, x_k) \in \textstyle\prod_{j=1}^k X_j : \sum_{j=1}^k h_j(x_j) \le t\}|$$

by dynamic programming. Then $|S'| = F(n-1, 2n^2\mathbf{1})$.

Let $\xi_i(t_i) = \left\lceil \dfrac{r_i t_i}{2n^2} \right\rceil$ and, for any $j \in [n-1]$, define

$$\begin{aligned}\Delta_j(t) &= |\{x_j \in X_j : h_j(x_j) = t\}| \\ &= |\{x_j \in X_j : \xi_i(t_i) \le x_{ij} < \xi_i(t_i+1),\ i \in [m]\}|.\end{aligned}$$

Then, if $s \in T$, the recurrence is

$$\begin{aligned}F(k,s) &= \sum_{t\in T} \Delta_k(t) F(k-1, s-t) \quad (k>1), \\ F(1,s) &= \Delta_1(s).\end{aligned}$$

Thus the table $F(k,s)$ $(k \in [n-1],\ s \in T)$ can be determined in $O(n^{4m+1}D)$ time, where D is the time needed to determine $\Delta_j(t)$. We consider this next.

LEMMA 4. *$\Delta_j(t)$ can be determined in $O(m2^m)$ arithmetic operations.*

PROOF. Note that each $\Delta_j(t)$ is of the form

$$M = |\{\zeta \in \mathbb{Z}^m : \textstyle\sum_{i=1}^m \zeta_i = \xi,\ 0 \le \zeta_i \le u_i\ (i \in [m])\}|. \quad (2)$$

For $\sigma \in \{0,1\}^m$, let

$$e(\sigma) = \sum_{i=1}^m \sigma_i, \quad z(\sigma) = \xi - \sum_{i=1}^m \sigma_i(u_i+1),$$

and

$$Z(\sigma) = \big\{\zeta : \ \textstyle\sum_{i=1}^m \zeta_i = z(\sigma),\ \zeta_i \ge 0\ (i \in [m])\big\}.$$

Note that

$$|Z(\sigma)| = \binom{z(\sigma)+m-1}{m-1}.$$

Now, using the principle of inclusion-exclusion, we have

$$\begin{aligned}M &= \sum_{\sigma\in\{0,1\}^m} (-1)^{e(\sigma)}|Z(\sigma)| \\ &= \sum_{\sigma\in\{0,1\}^m} (-1)^{e(\sigma)}\binom{z(\sigma)+m-1}{m-1}.\end{aligned}$$

Each term in the sum can be calculated in $O(m)$ arithmetic operations, and there are 2^m terms. $\square$

Thus all $F(k,s)$ can be calculated in $O(n^{4m+1})$ time. Hence we can determine $|S'|$. The probabilistic traceback takes $O(n^{2m+1})$ time, given that we can select uniformly from sets of the form (2) in constant time (for fixed m). We will consider this point later.

We now construct the mapping f from S' to S. This is not as straightforward as the construction of the mappings in section 2. If $x^* \in S$, then we set $f(x^*) = x^*$. Otherwise let $I(x^*) = \{i : \sum_{j=1}^{n-1} x^*_{ij} > r_i\}$. For $i \in I(x^*)$, let $p(i)$ be such that $x^*_{ip} = \max_{j=1}^{n-1} x^*_{ij} > r_i/n$. Now, for any $x^* \in S'$,

$$\begin{aligned}\sum_{j=1}^{n-1} x^*_{ij} &\le \frac{r_i}{2n^2}\sum_{j=1}^{n-1}([h_j(x_j)]_i + 1) \\ &< \frac{r_i}{2n^2}\big(2n^2+n\big) \\ &= \Big(1+\frac{1}{2n}\Big)r_i.\end{aligned}$$

Now let $J = \{j : j = p(i) \text{ for some } i \in I(x^*)\}$ and $\ell = |J|+1$. Note that $\ell \le m$, since $x_j \in X_j$, $j \in [n-1]$, implies that $[m] \setminus I(x^*) \ne \emptyset$. Let $\hat{y}_{ij} = 0$ $(i \in [m], j \in J)$, $\hat{y}_{ij} = x^*_{ij}$ otherwise. Then $\rho_i = r_i - \sum_{j=1}^{n-1}\hat{y}_{ij} \ge 0$.

If $i \in I(x^*)$, $\rho_i \ge x^*_{ip} - \frac{1}{2n}r_i \ge \frac{1}{2n}r_i$. Thus, if $i \in I(x^*)$, $j \in J$, $x^*_{ij} \le x^*_{ip} \le \rho_i + \frac{1}{2n}r_i \le 2\rho_i$. If $i \notin I(x^*)$, $j \in J$, then clearly $x^*_{ij} \le \rho_i$. Also, $x^*_j \in X_j$ implies $x^*_{ij} \le c_j$. Thus we have $x^*_{ij} \le 2\min(\rho_i, c_j)$ for any $i \in [m]$, $j \in J$.

We use the columns in $J \cup \{n\}$ to "complete" $\hat{y}$ to a contingency table y. We can do this by setting these columns to any $m \times \ell$ contingency table with row sums ρ_i, $i \in [m]$, and column sums c_j, $j \in J \cup \{n\}$. The method we use is similar to that in [3]. Let $N' = \sum_{i=1}^m \rho_i = \sum_{j\in J\cup\{n\}} c_j$, and let us assume without loss that the rows are re-numbered if necessary so that $\rho_m = \max_{i=1}^m \rho_i$. Note that $\rho_m \ge N'/m$ and $c_n \ge N'/\ell$.

Let $a_{ij} = \lfloor \min(\rho_i, c_j)/m^3 \rfloor$, $Q_{ij} = \lfloor x^*_{ij}/(a_{ij}+1) \rfloor \le 2m^3$, $R_{ij} = x^*_{ij} \bmod (a_{ij}+1)$, so that $x^*_{ij} = (a_{ij}+1)Q_{ij} + R_{ij}$, for $i \in [m-1], j \in J$. Now, for all $i \in [m-1], j \in J$, let

$$Q'_{ij} = \left\lfloor \frac{\rho_i c_j}{N'(a_{ij}+1)} \right\rfloor, \quad z_{ij} = (a_{ij}+1)Q'_{ij} + R_{ij}.$$

We must show that the z_{ij} can be augmented to give a contingency table. This will be so if, and only if,

$$\sum_{j\in J} z_{ij} \le \rho_i \quad (i \in [m-1]), \qquad (3)$$

$$\sum_{i\in[m-1]} z_{ij} \le c_j \quad (j \in J), \qquad (4)$$

$$\sum_{i\in[m-1]}\sum_{j\in J} z_{ij} \ge N' - \rho_m - c_n. \qquad (5)$$

This formulation is well known. See, for example, [10]. Now, for (3), we have

$$\begin{aligned}\sum_{j\in J} z_{ij} &\le \sum_{j\in J}\Big(\frac{\rho_i c_j}{N'} + a_{ij}\Big)\\ &\le \sum_{j\in J}\Big(\frac{\rho_i c_j}{N'} + \frac{\rho_i}{m^3}\Big)\\ &< \rho_i\left(1 - \frac{1}{\ell} + \frac{1}{m^2}\right)\\ &< \rho_i.\end{aligned}$$

Similarly, for (4),

$$\begin{aligned}\sum_{i=1}^{m-1} z_{ij} &\le \sum_{i=1}^{m-1}\Big(\frac{\rho_i c_j}{N'} + a_{ij}\Big)\\ &\le \sum_{i=1}^{m-1}\Big(\frac{\rho_i c_j}{N'} + \frac{c_j}{m^3}\Big)\\ &< c_j\left(1 - \frac{1}{m} + \frac{1}{m^2}\right)\\ &< c_j.\end{aligned}$$

Finally, for (5),

$$\begin{aligned}\sum_{i=1}^{m-1}\sum_{j\in J} z_{ij} &> \sum_{i=1}^{m-1}\sum_{j\in J}\Big(\frac{\rho_i c_j}{N'} - (a_{ij}+1)\Big)\\ &\ge \sum_{i=1}^{m-1}\sum_{j\in J}\Big(\frac{\rho_i c_j}{N'} - \frac{c_j}{m^3} - 1\Big)\\ &\ge \frac{(N'-\rho_m)(N'-c_n)}{N'} - \frac{N'-c_n}{m^2} - m^2\\ &\ge N' - \rho_m - c_n + \frac{N'}{m^2} - \frac{N'}{m^2} + \frac{N'}{m^3} - m^2\\ &\ge N' - \rho_m - c_n,\end{aligned}$$

provided $N' \ge m^5$. But this is implied by our assumption that $c_n \ge m^5$.

We can now define the mapping f. Let $f(x^*) = y^*$, where $y_j^* = x_j^*$, $j \notin J$, and $y_j^* = z_j$, $j \in J$. For the inverse mapping, the set J contains at most $(m-1)$ of the $(n-1)$ columns, and there are at most n^{m-1} ways of selecting it. Given the set J, the ρ_i ($i \in [m]$) can be calculated, and we can determine the largest ρ_i, which we assume to be ρ_m. Now the a_{ij} can be determined, and hence the R_{ij} from

$$R_{ij} = y_{ij}^* \bmod (a_{ij}+1) \quad (i \in [m-1], j \in J).$$

Finally we must select the Q_{ij}. Since

$$Q_{ij} \in [0, 2m^3] \quad (i \in [m-1], j \in J).$$

there are at most $(2m^3+1)^{(m-1)^2}$ ways of selecting them all. For fixed m this is constant. Thus $|f^{-1}(y^*)| = O(n^{m-1})$.

We can generate as before by tracing back. However, at each stage we now need to generate a random point in a set M of the form (2). Let us defer this issue temporarily, and suppose we can do this in constant time for fixed m. It then follows, using the same ideas as before, that a sample of ν tables can be computed in time $O(n^{4m+1} + \nu\, n^{3m})$, and that there is an *fpras* with running time $O(n^{4m+1} + \varepsilon^{-2} n^{3m})$.

Let us now return to the question of generation in the traceback. Fortunately, we can use the method above to bootstrap itself. Note that (essentially) a set of the form (2) is the set of solutions to a $2 \times m$ contingency table, with row sums ξ, $\sum_{i=1}^m u_i - \xi$ and column sums $u_1, u_2, \ldots, u_m$. Thus the method above can be used to generate a point in $O(m^9)$ time, provided that we can trace back when there are only two rows. Thus we need to generate a uniform point in

$$\zeta_1 + \zeta_2 = \xi, \quad 0 \le \zeta_1 \le u_1,\ 0 \le \zeta_2 \le u_2.$$

But this is straightforward. We choose

$$\zeta_1 \in [\max(0, \xi - u_2), \min(u_1, \xi)]$$

uniformly at random, and then set $\zeta_2 = \xi - \zeta_1$.

Finally, observe that this method for generation could be used to count approximately in the dynamic programming phase. Then we use Lemma 2 only for tables with two rows. With this approach, the implied constant in the time bound for the dynamic programming part of the algorithm depends only polynomially on m. We omit the details, since the appearance of m in the exponent of n makes this an issue of secondary importance.

4. REFERENCES

[1] F. Chung, R. Graham and S. Yau. On sampling with Markov chains. *Random Structures and Algorithms*, **9**, 1996, pp. 55–77.

[2] M. Cryan and M. Dyer. A polynomial-time algorithm to approximately count contingency tables when the number of rows is constant. In *Proceedings of the 34th Annual Symposium on Theory of Computing*, 2002, pp. 240–249.

[3] M. Cryan, M. Dyer, L. Goldberg, M. Jerrum and R. Martin. Rapidly mixing Markov chains for sampling contingency tables with a constant number of rows. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, 2002, pp. 711–720.

[4] J. De Loera and B. Sturmfels. Algebraic unimodular counting. Preprint, University of California at Davis, 2001.

[5] P. Diaconis and B. Efron. Testing for independence in a two-way table: new interpretations of the chi-square statistic (with discussion). *Annals of Statistics*, **13**, 1995, pp. 845–913.

[6] P. Diaconis and A. Gangolli, Rectangular arrays with fixed margins. In *Discrete Probability and Algorithms* (D. Aldous, P. Varaiya, J. Spencer and J. Steele, Eds.), IMA Volumes on Mathematics and its Applications, **72**, Springer, New York, 1995, pp. 15–41.

[7] P. Diaconis and L. Saloff-Coste. Random walk on contingency tables with fixed row and column sums. Technical Report, Department of Mathematics, Harvard University, 1995.

[8] M. Dyer, A. Frieze, R. Kannan, A. Kapoor, L. Perkovic and U. Vazirani. A mildly exponential algorithm for estimating the number of knapsack solutions. *Combinatorics, Probability and Computing*, **2**, 1993, pp. 271–284.

[9] M. Dyer and C. Greenhill. Polynomial-time counting and sampling of two-rowed contingency tables. *Theoretical Computer Science*, **246**, 2000, pp. 265–278.

[10] M. Dyer, R. Kannan and J. Mount. Sampling contingency tables. *Random Structures and Algorithms*, **10**, 1997, pp. 487–506.

[11] D. Hernek. Random generation of $2 \times n$ contingency tables. *Random Structures and Algorithms*, **13**, 1998, pp. 71–79.

[12] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, **58**, 1963, pp .13–30.

[13] M. Jerrum, *Counting, sampling and integrating: algorithms and complexity*, Birkhäuser, Basel, 2003.

[14] B. Morris, Improved bounds for sampling contingency tables. In *3rd International Workshop on Randomization and Approximation Techniques in Computer Science*, Lecture Notes in Computer Science **1671**, 1999, pp. 121–129.

[15] B. Morris, *Random walks in convex sets*. PhD thesis, Department of Statistics, University of California, Berkeley, 2000.

[16] B. Morris and A. Sinclair, Random walks on truncated cubes and sampling 0-1 knapsack solutions. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*, 1999, pp. 230–240.

[17] B. Morris and A. Sinclair, Random walks on truncated cubes and sampling 0-1 knapsack solutions, July 2002, submitted. (An extended and improved version of [16].)

[18] J. Mount, *Application of convex sampling to optimization and contingency table generation*. PhD thesis, Technical report CMU-CS-95-152, Computer Science Department, Carnegie Mellon University, 1995.

[19] A. Sinclair and M. Jerrum, Approximate counting, uniform generation, and rapidly mixing Markov chains. *Information and Computation*, **82**, 1989, pp. 93–133.

APPENDIX

We consider the multidimensional knapsack problem (1) of section 2.3 when the coefficients can be arbitrary integers.

LEMMA 5. *If the $a_{ij} \in \mathbb{Z}$ can be arbitrary then, unless RP=NP, there is no* fpras *for counting solutions to (1) for any $m \geq 2$, even if $b_i \geq K$ $(i \in [m])$ for any $K \in \mathbb{Z}_+$.*

PROOF. Let $K > 0$ be an arbitrary integer. First note that we can reduce general m to the case $m = 2$ by adding $(m-2)$ inequalities of the form $\sum_{j=1}^n a_{ij}x_j \leq \sum_{j=1}^n a_{ij} + K$ $(i \in [3, m])$ for arbitrary $a_{ij} \in \mathbb{N}$. Clearly these inequalities are all redundant.

For $m = 2$, suppose to the contrary that an *fpras* exists. Let $\sum_{j=1}^n a_j x_j = b$, where $a_j \in \mathbb{Z}_+$ $(j \in [n])$, be an arbitrary instance of the NP-Complete zero-one knapsack *decision* problem.

Consider the instance

$$\begin{aligned} \sum_{j=1}^n (K+1)a_j x_j - (K+1)bx_{n+1} &\leq K, \\ -\sum_{j=1}^n (K+1)a_j x_j + (K+1)bx_{n+1} &\leq K, \end{aligned}$$

of the 2-constraint zero-one knapsack *approximate counting* problem. This is obviously equivalent to

$$\left| \sum_{j=1}^n a_j x_j - bx_{n+1} \right| \leq \frac{K}{K+1}.$$

If $x_{n+1} = 0$, then clearly $x_j = 0$ $(j \in [n])$ is the unique solution. If $x_{n+1} = 1$, then any solution is a solution of $\sum_{j=1}^n a_j x_j = b$. Thus, if we can distinguish between one solution and two or more solutions, we can decide whether the knapsack instance has a solution. But using the *fpras* with any $\varepsilon < \frac{1}{2}$ (say) we can clearly decide this in BPP, which would then imply RP=NP. See [13, p. 94]. □