Estimating the range of a function in an online setting.

John A. Mount ⋆

jmount@esgear.com 552 Melrose Ave.
San Francisco, Ca 94127-2220

Abstract

Consider an unknown function $L(\cdot) : \{1, \cdots, d\} \to \{1, \cdots, r\}$ with range $R = \{L(i) | i = 1, \cdots, d\}$. Given $d, r, \epsilon, \delta > 0$ we show how to compute an estimate $\tilde{p}$ such that with probability at least $1 - \delta$ we have $||R|/r - \tilde{p}| \leq \epsilon \tilde{p}$. This is an estimate with a fixed *relative* error, which is stronger than finding an estimate with a fixed absolute error. This calculation can be performed efficiently in one pass through the domain of L (allowing the the method to be used in online situations) using only $O\left(\log r \left(\log \log r + \log 1/\delta\right) / \epsilon^2\right)$ words of storage. The method is based on pairwise-independent pseudo-random variables.

Key words: Keywords: analysis of algorithms, online algorithms, pairwise independence, pseudo-random generation, randomized algorithms, sampling.

1 Problem and Notation

Assume we have access to an unknown function $L(\cdot) : \{1, \cdots, d\} \to \{1, \cdots, r\}$ and memory of size polynomial in $\log(r), \log(1/\delta), 1/\epsilon$. Let R be the range of $L(\cdot)$: $\{L(x) | x = 1, \cdots, d\}$. Our question is: can we estimate $p = |R|/r$ to within a small *relative* error? We assume that $L(\cdot)$ is available only in an *online* fashion. That is we are given $d, r, \delta, \epsilon > 0$ and have access to an oracle that returns the tuple $(i, L(i))$ on the ith access (results after dth access undefined). However, for convenience we will argue as if we had direct read-once access to $L(\cdot)$.

Our interest in this problem arose after solving a problem in combinatorial chemistry [5] using a dynamic programming approach [1]. We developed

⋆ Work performed while at CombiChem, Inc.

an algorithm that could find all solutions to a problem subject to certain additional side-constraints. We then had access to all solutions of the original problem by enumerating all possible settings for the side-constraints. We wanted to know how many solutions the original (unaltered) problem admitted. We considered an online formulation for the following reasons. The same solution could occur for many different settings of the side-constraints. The solution method was expensive. More solutions were found than we could store, yet solutions were very rare.

The difficult case to estimate is when simultaneously R is too large to store and $|R|/r$ is so small that we can not explicitly store the description of a sample of $\{1, \cdots, r\}$ large enough to have a good chance of hitting R. We exhibit a simple probabilistic algorithm that can determine an estimate $\tilde{p}$ such that $||R|/r - \tilde{p}| \leq \epsilon\tilde{p}$ in only one pass through the domain of L and using only $O\left(\log r \left(\log\log r + \log 1/\delta\right)/\epsilon^2\right)$ words of storage. The parameter δ is the odds that the algorithm fails.

Our solution is based on the observation: if we could draw a sample $S \subset \{1, \cdots, r\}$ uniformly at random such that $|S| = k/\epsilon^2 p$ (for some constant k), then with high probability we have *both* that $|S \cap R|/|S| \approx |R|/r = p$ and that $|S \cap R| \in O\left(1/\epsilon^2\right)$ independent of p. This observation follows from Stockmeyer's $\log\log r$ approximate counting scheme [7] or Sipser's Coding Lemma [6]. We produce a pairwise-independent sample, $S_{a,b}$, that requires only a constant number of $O\left(\log r\right)$ sized words of storage to specify. We show that with probability $\geq \frac{3}{4}$ we have both that the sample has nearly the correct density (p) and that $|S_{a,b} \cap R|$ is small ($O\left(1/\epsilon^2\right)$ words). To simplify our proofs we work with $|S_{a,b} \cap R|/\mathrm{E}[|S_{a,b}|]$ (an estimate of the density of our sample) instead of the true sample density $|S_{a,b} \cap R|/|S_{a,b}|$. We exhibit a procedure, called **scan**, that produces an independent family of such samples so that with overwhelming odds a good estimate can be found. The procedure **scan** itself requires a rough estimate of p to design its samples. A procedure called **est** that finds such an estimate by attempting a geometric sequence of values against **scan**.

The relationship between random generation and counting is already very well understood [2], our formulation emphasizes limited space in an online setting and computing within a relative error ϵ (treated as a parameter rather than a constant) instead of an absolute error. Standard methods require one of $|R|$, $r/|R|$ or $\max_{x,y \in R} |\{i|L(i) = x\}|/|\{i|L(i) = y\}|$ to be bounded by a polynomial (we do not).

For integers x, r ($r > 0$) we define $< x >_r$ to be the unique integer y such that $0 \leq y \leq r - 1$ and r divides into $x - y$ evenly. For technical reasons we assume that r is a prime number.

2 Method

We define a procedure called **scan**$(d, r, u, L(\cdot), \gamma, \epsilon)$. As above, d is the cardinality of the domain, r is the cardinality of the co-domain (and prime), and u is a bound ($0 < u \leq 1$) such that $u \geq p = |R|/r$. The function $L(\cdot)$ takes an integer from $\{1, \cdots, d\}$ and returns an integer from $\{1, \cdots, r\}$. We define γ, ϵ as two control parameters ($0 < \gamma \leq 1$, $0 < \epsilon \leq 1$). The procedure **scan** returns either ("estimate",$\tilde{p}$) where $\tilde{p}$ is such that $|p - \tilde{p}| \leq \epsilon\tilde{p}$ or returns ("bound",$\tilde{p}$) which means that we have a probabilistic proof that $p \leq \tilde{p} \leq u/2$ or returns ("failure",1). We prove that **scan** returns a correct "estimate" or "bound" with probability at least $1-\gamma$. This procedure simulates drawing a pseudo-random sample $S_{a,b}$ that is usually of size m (defined below). The procedure then attempts to compute $|S_{a,b} \cap R|$, which can be done as long as $|S_{a,b} \cap R| \leq l$ (where l is our space-bound). If $|S_{a,b} \cap R| > l$ the sample is considered a failure (we account for the bias this introduces). The sample $S_{a,b}$ is the set of all i such that $< a \times i + b >_r < m$. The size of the sample varies ($|S_{a,b}| = m$ if $a \neq 0$ otherwise it is 0 or r, depending on b), but we have $\mathrm{E}[|S_{a,b}|] = m$. We show that with significant probability the computation can be completed within the space-bound and we have $|S_{a,b} \cap R|/\mathrm{E}[|S_{a,b}|] \approx p$. Using a median finding trick we show, with probability at least $1 - \gamma$, a moderate number of repetitions, which can be performed in parallel, are sufficient to guarantee that **scan** does not return "failure" *and* and the returned "bound" or returned "estimate" is correct. A second procedure **est** produces the necessary bounds for **scan** and completes the argument. For notational clarity (e.g. avoiding some subscripts) **scan** and **est** are described as making multiple passes through the domain of $L(\cdot)$. The one-pass or online/oracle implementation should be obvious and is our primary interest.

We define the procedure **scan**:

```
procedure scan(d, r, u, L(·), γ, ε)
   set tol = εu/2(1+ε) (the working tolerance)
   set m = ⌈64(1+ε)²/(ε²u)⌉ (the target sample size)
   set l = ⌈(u+tol) m⌉ + 2 (how many of the samples we can count)
   set t = ⌈12 log (1/γ)⌉ + 1 (the number of trials)
   if max(l, m, 1/u) ≥ min(d, r) then {
         return explicit count: ("estimate",|R|/r)
         (if this requires more than l words of storage return ("failure",1))
   }
   for h = 1, ···, t do {
         pick a_h ∈ {0, ···, r−1} uniformly at random
         pick b_h ∈ {0, ···, r−1} uniformly at random
         set V_h = {}, z_h = 0, i = 1
         while i ≤ d and z_h ≠ ∞ do {
               if < a_h × L(i) + b_h >_r < m and L(i) ∉ V_h then {
                     if |V_h| ≤ l then {
                           set V_h = V_h ∪ {L(i)}
                     } else {
                           set z_h = ∞
                     }
               }
               set i = i + 1
         }
         if z_h ≠ ∞ then set z_h = |V_h|/m
   }
   set p̃ = any median of z_1, z_2, ···, z_t
   if p̃ is ∞ then return ("failure",1)
   if p̃ ≥ u/2(1+ε) then return ("estimate",p̃)
   return ("bound",p̃ + tol)
endprocedure
```

We have, for clarity, left open how the sets V_h are maintained. We can assume that some efficient, $O(\log |V_h|)$ time per access, method is used.

Theorem 1 *If $u \geq p$ then in procedure* **scan** *for each h:*

$$Pr[z_h = \infty \text{ or } |z_h - p| \geq \epsilon u/2(1+\epsilon)] < \frac{1}{4}\ .$$

Proof: Using the assumption that $u \geq p$, there is nothing to show unless $\max(l, m, 1/u) < \min(d, r)$. For each $h = 1, \cdots, t$ and $j \in R$, define a random variable (random in the choice of a_h, b_h):

$$\zeta_{h,j} = \begin{cases} 1 & < a_h \times j + b_h >_r < m \\ 0 & \text{otherwise} \end{cases}\ .$$

Observe that the $\zeta_{h,j}$ are identically distributed, and that $\Pr[\zeta_{h,j} = 1] = m/r$ and that $\zeta_{h,j}, \zeta_{h,k}$ (for $j \neq k$) are pairwise-independent (see [3]). Define

$$\zeta_h = \frac{1}{m} \sum_{j \in R} \zeta_{h,j} \ .$$

We have $l \geq (u + \epsilon u/2(1+\epsilon))m$ so it is sufficient to show $\Pr\left[|\zeta_h - p| \geq \frac{\epsilon}{4(1+\epsilon)} u\right] < \frac{1}{4}$. By the Chebychev inequality [3,4] we have

$$\Pr\left[|\zeta_h - p| > \frac{\epsilon}{4(1+\epsilon)} u\right] \leq \mathrm{E}\left[(\zeta_h - p)^2\right] / \left(\frac{\epsilon u}{4(1+\epsilon)}\right)^2 .$$

$$\begin{aligned}
\mathrm{E}\left[(\zeta_h - p)^2\right] &= \mathrm{E}\left[\left(\frac{1}{m}\sum_{j\in R}\left(\zeta_{h,j} - \frac{m}{r}\right)\right)^2\right] \\
&= \mathrm{E}\left[\frac{1}{m^2}\sum_{j\in R}\sum_{k\in R}(\zeta_{h,j} - m/r)(\zeta_{h,k} - m/r)\right] \\
\text{(by pairwise-independence)} &= \mathrm{E}\left[\frac{1}{m^2}\sum_{j\in R}(\zeta_{h,j} - m/r)^2\right] \\
\text{(by identical distribution)} &= \frac{1}{m^2} pr \mathrm{E}\left[(\zeta_{h,1} - m/r)^2\right] \\
&= \frac{1}{m^2} pr \left(\frac{m}{r}\right)\left(1 - \frac{m}{r}\right) \\
&< p/m
\end{aligned}$$

Because $u \geq p$ and $m \geq 64(1+\epsilon)^2/\epsilon^2 u$ we have:

$$\Pr\left[|\zeta_h - p| \geq \frac{\epsilon}{4(1+\epsilon)} u\right] < \frac{p}{m} / \left(\frac{\epsilon u}{4(1+\epsilon)}\right)^2 \leq 1/4 \ .$$

□

Theorem 2 *If $u \geq p$ then procedure* **scan** *returns ("estimate",$\tilde{p}$) such that $|\tilde{p} - p| \leq \epsilon \tilde{p}$ or ("bound",$\tilde{p}$) such that $p \leq \tilde{p} < u/2$ with probability at least $1 - \gamma$.*

Proof: Again, assuming $u \geq p$, there is nothing to show unless $\max(l, m, 1/u) < \min(d, r)$. For each $h = 1, \cdots, t$, define

$$s_h = \begin{cases} 1 & z_h = \infty \text{ or } |z_h - p| \geq \epsilon u/2(1+\epsilon) \\ 0 & \text{otherwise} \end{cases} .$$

Let g be the index such that $z_g = \tilde{p}$ is the median of $z_1, \cdots, z_t$ picked by algorithm **scan**. The ordering of the z_h's is a refinement of the ordering of the s_h's, so s_g is itself a median of $s_1, \cdots, s_t$. The s_h $(h = 1, \cdots, t)$ are $\left\lceil 12 \log \frac{1}{\gamma} \right\rceil + 1$ independent random variables with $\mathrm{E}[s_h] < 1/4$ (by Theorem 1); using Jerrum, Valianta and V. Vazirani's Lemma 6.1 [2] we see that with probability at least $1-\gamma$ we have that $s_g = 0$. When this is the case we have $|\tilde{p}-p| < \epsilon u/2(1+\epsilon)$. If $\tilde{p} \geq u/2(1+\epsilon)$ then **scan** returns $\tilde{p}$ as an "estimate" and we have $|\tilde{p}-p| \leq \epsilon\tilde{p}$. Otherwise $\tilde{p} < u/2(1+\epsilon)$ and **scan** returns "bound", which is correct since we have $p \leq \tilde{p} + \epsilon u/2(1+\epsilon) < \frac{1}{2}u$.

$\square$

We now have enough tools to state our overall algorithm ($d, r, L(\cdot), \delta, \epsilon$ are defined as before).

procedure **est**$(d, r, L(\cdot), \delta, \epsilon)$
  set $w = \lceil \log_2 r \rceil + 1$
  for $s = 0, \cdots, w$ **do** {
    set (message$_s$,$\tilde{p}_s$) $=$ **scan**$(d, r, 2^{-s}, L(\cdot), \delta/(w+1), \epsilon)$
  }
  set (message$_{w+1}$,$\tilde{p}_{w+1}$) $=$ ("failure",1)
  set $s = 0$
  while message$_s$ is "bound" **do** {
    set $s = s + 1$
  }
  return (message$_s$,$\tilde{p}_s$)
endprocedure

We point out that all of the results from the "for s" loop of **est** can be computed in a single pass through the domain of $L(\cdot)$ using only $O\left(\log r \left(\log\log r + \log 1/\delta\right)/\epsilon^2\right)$ words of storage. This can be accomplished by reversing the nesting of the "for s" loop in **est** with the "while i" loop in **scan** and maintaining a separate copy of each variable in **scan** for each value of $s = 0, \cdots, w$. Therefore we can apply our algorithm in an online setting.

After this data is assembled, **est** examines some of the returned "messages" in order. Let k be the larger integer such that $2^{-k} >= p$. For each $s = 0, \cdots k$ algorithm **scan** was called with a correct upper bound for p. For each of these calls to **scan** Theorem 2 applies and we have that the returned information (message$_s$,$\tilde{p}_s$) is correct with probability at least $1 - \delta/(w+1)$. The disjoint-union bound tells us with probability at least $1 - (k+1)\delta/(w+1) \geq 1 - \delta$ we have: all of (message$_0$,$\tilde{p}_0$) through (message$_k$,$\tilde{p}_k$) are correct. When this is the case we have message$_s$ = "bound" for $s < k$, message$_k$ = "estimate" and no (message$_s$,$\tilde{p}_s$) with $s > k$ is examined by **est**. So with probability at least $1 - \delta$ algorithm **est** returns a bound within the desired tolerance.

We note an obvious variant of the above algorithm would be to change the upper bound by a factor of $1+\epsilon$ (instead of 2) at each stage. Then **scan** would only have to verify the given estimate, but this would blow up the number of stages needed by **est** by a factor of $O(1/\log(1+\epsilon)) \approx O(1/\epsilon)$.

3 Sampling

Another problem is generating a sample x such that $x \in R$ and $\Pr[x = y | y \in R] \approx 1/|R|$. One can, of course, use the counting methods of [2], but a simple direct method would be of interest.

A natural candidate method is to choose a uniformly at random from $\{0, \cdots, r-1\}$, choose b uniformly at random from $\{0, \cdots, r-1\}$ and set $S_{a,b} = R \cap \{i | i = 1, \cdots, r \text{ and } < a \times i + b >_r < m\}$. If $|S_{a,b}| \leq l$ (l as in procedure **scan**) we return x picked uniformly at random from $S_{a,b}$, otherwise we return "failure." Unfortunately, x is not always nearly uniformly distributed in R. This is *not* due to the rejection of sets $S_{a,b}$ where $|S_{a,b}| > l$. The reason is that, even though the random variables

$$\chi_x = \begin{cases} 1 & x \in S_{a,b} \\ 0 & \text{otherwise} \end{cases}$$

are identically distributed for all $x \in R$ (even pairwise-independent) the random variables

$$\rho_x = \begin{cases} 1/|S_{a,b}| & x \in S_{a,b} \\ 0 & \text{otherwise} \end{cases}$$

are not identically distributed for all $x \in R$.

An example of this is found by examining the multiplication table of the group $\mathbb{Z}_7$ which corresponds in our algorithm to: $d = 7, r = 7, m = 4, R = \{1, 2, 3, 4\}$. We see that the image of symbol 4 (under the mapping $x \rightarrow < a \times x + b >_r$) can occur alone (pick $a = 1, b = 3$) but that the image of symbol 2 never occurs in a sample by itself. The image of symbol 2 is always in-between the image of two other symbols from $\{1, 3, 4\}$ that are within 5 units from each other (so any 4-unit window over the image of 2 picks up one of the other symbols). Thus 4 occurs in sets $S_{a,b}$ where $|S_{a,b}| = 1$, and 2 does not. In fact the sampling method mentioned above generates the symbol 4 more often than the symbol 2.

4 Acknowledgements

The author would like to thank the editor and referees for their patient assistance.

References

[1] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, McGraw Hill, 1990.

[2] M. Jerrum, L. G. Valiant, and V. Vazirani, *Random generation of combinatorial structures from a uniform distribution*, Theoretical Computer Science, 1986, pp. 169–188.

[3] M. Luby, *Pseudorandomness and Cryptographic Applications*, Princeton University Press, 1996.

[4] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.

[5] M. Plunkett and J. Ellman, *Combinatorial chemistry and new drugs*, Scientific American, 276, 1997, pp. 68–73.

[6] M. Sipser, *A complexity theoretic approach to randomness*, in 15th Symposium on the Theory of Computing, ACM, 1983, pp. 330–335.

[7] L. Stockmeyer, *The complexity of approximate counting*, in 15th Symposium on the Theory of Computing, ACM, 1983, pp. 118–126.