Available online at www.sciencedirect.com
SCIENCE DIRECT®
ACADEMIC PRESS
Journal of Computer and System Sciences 66 (2003) 2–19
JOURNAL OF COMPUTER AND SYSTEM SCIENCES
http://www.elsevier.com/locate/jcss

Analysis and application of adaptive sampling ☆

James F. Lynch

Department of Mathematics and Computer Science, Box 5815, Clarkson University, Potsdam, NY 13699-5815, USA

Received 1 October 2000; revised 1 January 2002

Abstract

An estimation algorithm for a query is a probabilistic algorithm that computes an approximation for the size (number of tuples) of the query. One class of estimation algorithms uses a form of statistical sampling known as adaptive sampling. Several versions of adaptive sampling have been developed by other researchers. The original version has been surpassed in some ways by a newer version and a more specialized Monte-Carlo algorithm. An analysis of the cost of the original version is presented, and the different algorithms are compared. The analysis is used to derive an upper bound on the number of samples required by the original algorithm. Also, contrary to what seems to be a commonly held opinion, none of the algorithms is generally better than the other two. Which algorithm is superior depends on the query being estimated and the criteria that are being applied. Another question that is studied is which classes of logically definable queries have fast estimation algorithms. Evidence from descriptive complexity theory is provided that indicates not all such queries have fast estimation algorithms. However, it is shown that on classes of structures of bounded degree, all first-order queries have fast estimation algorithms.

Keywords: Relational databases; Queries; Sampling

1. Introduction

Estimating the size of a query can be described as the following problem. Given a database $\mathfrak{A}$, let R be the query evaluated on $\mathfrak{A}$. $|R|$ is the size of R, i.e., the number of tuples in it. Let A be a probabilistic algorithm such that when given $\mathfrak{A}$, it returns an estimate $\mathbf{X}$ for $|R|$. Suppose that there is $E > 0$ and a probability $p < 1$ such that

$$\mathrm{pr}(|\mathbf{X} - |R|\,| \leqslant E) \geqslant p.$$

☆ Research supported by NSF Grant CCR-9406809.
E-mail address: jlynch@clarkson.edu (J.F. Lynch).

0022-0000/03/$ - see front matter © 2003 Published by Elsevier Science (USA).
PII: S0022-0000(02)00027-2

Then we say that A estimates $|R|$ with error E and confidence p. We are interested in finding fast algorithms that can estimate $|R|$ to any desired degree of accuracy and confidence. "Fast" can be interpreted in two ways. Either, in a relative sense, the estimation algorithm runs in time $o(t(n))$, where $t(n)$ is the best-known time required to compute $|R|$ exactly, or, in an absolute sense, its running time has low complexity, say linear or even constant.

There are several reasons why database researchers are interested in fast algorithms for estimating the sizes of queries. Some queries, in particular those defined recursively, can be quite expensive to compute. If the only important question is their size, then a fast estimation algorithm has significant practical value. Even if an exact computation of the query is needed, a fast estimation of its size can help to decide whether it is feasible to actually perform the computation. Also, the speed of computing the composition of simpler queries can depend on the order of their execution. Estimations of their size can help determine the optimal order of execution.

An obvious question is whether such algorithms exist for all queries. The following argument based on descriptive complexity theory [7] indicates that this is unlikely. For any natural number n, any $s \in \{0,1\}^n$ can be associated with a finite structure $\mathfrak{A}_s$ whose universe is $\{1, \ldots, n\}$. For any PSPACE language $L \subseteq \{0,1\}^*$, there is a formula $\phi(x)$ in partial fixed point logic such that for all s, it defines a unary query $R = \{i \in \{1, \ldots, n\} : \mathfrak{A}_s \models \phi(i)\}$ such that $|R| = 0$ or $|R| = n$, and $s \in L$ if and only if $|R| = n$. Taking $E = |R|/3$, if there were a fast estimation algorithm for ϕ for every probability $p < 1$, then there would be a fast probabilistic decision algorithm A such that for all $s \in \{0,1\}^*$, with probability at least p, A accepts s if and only if $s \in L$. In particular, if the algorithm A can always be shown to run in polynomial time, then PSPACE would collapse to PP.

This does not preclude the possibility that there are large classes of databases and queries that do have fast estimation algorithms, or whose average runtime is fast. In particular, do there exist fast estimation algorithms for all first-order queries on all databases? We do not know the answer to this question, but we will show that there are fast average time algorithms for estimating all first-order queries on databases of bounded degree. We will define this notion precisely later; it is just the graph theoretic notion of bounded degree generalized in the natural way to arbitrary relational structures. A number of researchers, for example Abiteboul et al. [1] have suggested that structures of bounded degree occur frequently in practice, and thus algorithms that are more efficient on these structures than general purpose algorithms would be useful. Abiteboul, Compton, and Vianu were not studying estimation algorithms; rather, they considered probabilistic methods for speeding up the average time for exact evaluation of higher-order queries.

There are actually two notions of average time complexity for database algorithms. The first regards the database as fixed, and the algorithm makes random choices in computing its estimate. The second assumes the database itself is random. (The algorithm may or may not be probabilistic.) In this article, we concentrate on the first notion. Specifically, we investigate a statistical sampling method for estimating the sizes of queries in databases.

A variety of methods for sampling queries in databases have been proposed. One approach is a form of sequential sampling. It relies on a partitioning of the query into sets that agree on the value of some attribute. More abstractly, let n be a natural number and $R \subseteq \{1, \ldots, n\}^r$ be an r-ary relation, i.e., relational query. For $i = 1, \ldots, n$, let $R_i = \{(i, i_2, \ldots, i_r) \in R\}$. Then $|R| = \sum_{i=1}^{n} |R_i|$.

The sampling algorithm repeatedly makes random choices of $i \in \{1, \ldots, n\}$, calculates $|R_i|$, and adds it to a cumulative sum until a predetermined stopping condition is satisfied. Then it produces an estimate of $|R|$.

Statistical inference based on the sum of a sequence of random variables was used by Bartky [2] for other applications, and the general theory of sequential sampling was formulated by Wald [21,22]. These authors considered the more general case when the sum could have negative terms. Lipton and Naughton [17] developed a version of sequential sampling which estimated the sizes of queries as outlined above. Their algorithm stopped sampling when the sum exceeded a predetermined value. They called it adaptive sampling because the number of samples taken was not predetermined; it depended on how quickly the sum reached the upper bound. Actually, similar stopping conditions had been used by Bartky and Wald. In their case, the sampling stopped when the sum either exceed an upper bound or fell below a lower bound.

The upper bound used by Lipton and Naughton was of the form $cb(n)$, where $b(n)$ is an a priori upper bound on the size of the partitions. (Thus, with no additional knowledge, $b(n) = n^{r-1}$.) Their error term was of the form $\varepsilon|R|$, for some constant $\varepsilon > 0$. They claimed that, by making c large enough, any desired degree of accuracy can be obtained with arbitrarily high probability. That is, for any $\varepsilon > 0$ and $p \in [0, 1)$, the probability that the estimate is within $\varepsilon|R|$ of $|R|$ can be made as large as p. This is true, but their proof is incorrect. Our proof will be a consequence of a more general analysis of the cost of running the algorithm.

A later version of adaptive sampling is due to Haas and Swami [12]. It does not make any a priori assumptions about the data except that not all the $|R_i|$ are equal. Haas and Swami prove that, for fixed R, as $\varepsilon \to 0$, the probability that their method produces an estimate that is within $\varepsilon|R|$ of $|R|$ approaches p. Further, their algorithm is asymptotically efficient in the sense that as $\varepsilon \to 0$, the number of samples taken is close to the minimum number needed to get an estimate that is within $\varepsilon|R|$ of $|R|$ with probability p. Thus, their algorithm will perform better than the Lipton–Naughton algorithm on fixed samples as the error factor ε goes to 0. Although Haas and Swami clearly state in their paper that their estimate may not be accurate if ε does not get small, it appears to be the belief of some database researchers that the Haas–Swami algorithm is generally better than the Lipton–Naughton algorithm (personal communications; see also [20]). But the two algorithms are actually incomparable. We will show that, for fixed ε and highly skewed data, the Lipton–Naughton algorithm can give a better estimate than the Haas–Swami algorithm.

2. Adaptive sampling

Adaptive sampling is a general method that applies to estimating the sum of a finite sequence of nonnegative integers. For a natural number n, let $a_1, \ldots, a_n$ be a sequence of nonnegative integers such that $a_i \leqslant b(n)$ for $i = 1, \ldots, n$, where b is a function of n. The usual interpretation in databases is that $a_i = |R_i|$, $b(n) = n^{r-1}$, and $\sum_{i=1}^{n} a_i = |R|$. In [17], Lipton and Naughton introduced the following algorithm for estimating $A = \sum_{i=1}^{n} a_i$. The c is a fixed constant independent of n.

Algorithm 1.

$S \leftarrow 0$
$m \leftarrow 0$
repeat
  do $j \leftarrow \text{RANDOM}(1, n)$
    $S \leftarrow S + a_j$
    $m \leftarrow m + 1$
until $S \geqslant cb(n)$
return nS/m

In a later article [18], the same authors claimed that any degree of accuracy and confidence could be obtained from this algorithm by taking c large enough. Specifically,

Claim. *Assume* $a_i > 0$ *for* $i = 1, \ldots, n$. *For* $0 \leqslant p < 1$ *and* $d > 0$, *if* $c = d(d+1)/(1-\sqrt{p})$, *then*

$$\text{pr}(|nS/m - A| \leqslant A/d) \geqslant p.$$

Lipton and Naughton used an urn model to show that, by taking a sufficiently large constant α, the probability that the algorithm stops before $\alpha b(n)$ steps can be made arbitrarily small, and for any given $m \geqslant \alpha b(n)$, $\text{pr}(|nS/m - A| > A/d)$ can also be made arbitrarily small. However, this does not imply their claim because they still need to show either

1. the sum over all $m \geqslant \alpha b(n)$ of $\text{pr}(|nS/m - A| > A/d)$ is arbitrarily small, or
2. for any $m \geqslant \alpha b(n)$, the conditional probability that $|nS/m - A| > A/d$, given that the algorithm stops at m repetitions, is arbitrarily small.

Method 1 would require a very tight bound on the probabilities being summed. Obvious approaches are suggested by the Central Limit Theorem, but as we will show, the error term in the Central Limit Theorem is so large that such a tight bound cannot be obtained. Similar problems arise in attempting to use Chernoff [5,13] or Hoeffding [16] bounds. (But see Section 3 for an estimation method using Hoeffding bounds that can outperform adaptive sampling under certain circumstances.) Regarding method 2, P. Haas (personal communication) suggested that a "random index" version of one of these theorems might be useful. That is, the number of terms m in the sum S is a random variable that depends on the terms being summed, and thus what is needed is an inequality involving a random number of random terms.

The approach taken here is to derive formulas for the expectation and variance of the stopping time, i.e., the final value of m. From these formulas, it will follow that for sufficiently large c, m will be close to $cnb(n)/A$ with high probability. Since the final value of S is between $cb(n)$ and $(c+1)b(n)$, the estimate nS/m will be close to A. Our upper bound on c does not seem to give a direct comparison with c in the claim. For some sequences $a_1, \ldots, a_n$, our bound is lower. For small ε, our bound is generally slightly larger. However, the condition that $a_i > 0$ for all $i = 1, \ldots, n$ can be relaxed. We need to assume only that $a_i \geqslant 0$ for all such i, and $a_i > 0$ for at least one i.

A complementary approach is used by Watanabe [23]. (It is described only for Boolean-valued sequences $a_1, \ldots, a_n$, but it generalizes immediately to sequences of natural numbers.) Using elementary formulas for the expectation and variance of S after m steps, he shows that S is unlikely to be between $cb(n)$ and $(c+1)b(n)$ unless m is close to $cnb(n)/A$. The upper bound for c is not explicitly shown, but it is also slightly larger than the claim.

The stopping time may be regarded as the cost of running Algorithm 1 when each iteration costs one unit. We will consider more general cost functions. For each $i = 1, \ldots, n$, let g_i be the cost of computing a_i, given i. This is essentially the cost of one iteration of Algorithm 1 when $j = i$. Thus, if j_t is the choice of the variable j made during iteration t of the algorithm, and it runs for m iterations, then the total cost of the algorithm is $\sum_{t=1}^{m} g_{j_t}$. When the a_i's are the sizes of the partitions R_i, the cost could be the time required to compute R_i. For example, Lipton and Naughton [17] used adaptive sampling to estimate the size of the transitive closure of a graph. In their application, R_i was the size of the reachability set from vertex i, which can be computed in time proportional to the number of edges in the component containing i. Haas and Swami [12] considered the problem of estimating the size of an equijoin, and used the cost function $g_i = c_1 + c_2 a_i$ where c_1 and c_2 are constants. Specializing our results to the case when all $g_i = 1$ will enable us to derive an upper bound on c.

We will model Algorithm 1 as a one-dimensional random walk, where at each step, the particle makes a jump to the right, the distance being randomly chosen from $\{a_1, \ldots, a_n\}$. The cost of the jump a_i is g_i. Here, the particle's initial position is 0, and the final m is the time at which the particle passes through an absorbing barrier located at $cb(n)$. At any given time m, its location is some integer k. It will continue its walk if $k < cb(n)$, in which case its position at time $m+1$ will be $k + a_i$, for some randomly chosen $i \in \{1, \ldots, n\}$, where all choices are equally likely. Thus its location at each time m is the value of S in the algorithm after m iterations of the **repeat**…**until** loop, and the time until absorption is the final value of m. For each $k \geqslant 0$, let $\mathbf{T}_k$ be the random variable that is the total cost incurred by the particle, when initially located at k. We will derive recurrences for $\mathbf{E}[\mathbf{T}_k]$ and $\mathbf{Var}[\mathbf{T}_k]$. For $k, t \geqslant 0$, let $p_{k,t}$ be the probability that $\mathbf{T}_k = t$, and let $p_k(z) = \sum_{t=0}^{\infty} p_{k,t} z^t$ be the generating function of $\langle p_{k,t} : t \geqslant 0 \rangle$. Then,

$$\mathrm{pr}(\mathbf{T}_k < \infty) = p_k(1),$$
$$\mathbf{E}[\mathbf{T}_k] = p_k'(1)$$
and
$$\mathbf{Var}[\mathbf{T}_k] = p_k''(1) + p_k'(1) - p_k'(1)^2.$$

It can be seen that the $p_{k,t}$ satisfy the following recurrence:

$$p_{k,0} = 1 \quad \text{for } k \geqslant cb(n),$$
$$p_{k,t} = 0 \quad \text{for } k \geqslant cb(n) \text{ and } t > 0,$$
$$p_{k,t} = 0 \quad \text{for } k < cb(n) \text{ and } t \leqslant 0,$$
$$p_{k,t} = \sum_{i=1}^{n} p_{k+a_i, t-g_i}/n \quad \text{for } k < cb(n) \text{ and } t > 0.$$

Therefore, for $k < cb(n)$,

$$p_k(z) = \sum_{i=1}^{n} z^{g_i} p_{k+a_i}(z)/n, \tag{1}$$

$$p'_k(z) = \sum_{i=1}^{n} [g_i z^{g_i-1} p_{k+a_i}(z) + z^{g_i} p'_{k+a_i}(z)]/n \tag{2}$$

and

$$p''_k(z) = \sum_{i=1}^{n} [g_i(g_i-1)z^{g_i-2} p_{k+a_i}(z) + 2g_i z^{g_i-1} p'_{k+a_i}(z) + z^{g_i} p''_{k+a_i}(z)]/n. \tag{3}$$

In the following, $H = \{i{:}a_i > 0\}$, $h = |H|$, $M = \max(a_1, \ldots, a_n) - 1$, and $G = \sum_{i=1}^{n} g_i$.

Lemma 2.1. *For all integers k,*

$$p_k(1) = 1.$$

Proof. This follows from more general theorems about unlimited Bernoulli trials or random walks (see, e.g., Feller [8]), but we give an elementary proof. The lemma is obvious for $k \geqslant cb(n)$. We will prove it for the remaining values by decreasing induction on k. Thus, assume $k < cb(n)$ and the lemma holds for all $k' > k$. Recalling our assumption that $h > 0$, by (1) and the induction assumption,

$$p_k(1) = \frac{h}{n} + \left(\frac{n-h}{n}\right) p_k(1),$$

and the lemma follows. □

Lemma 2.2. *For $k \leqslant cb(n) + M$,*

$$\frac{(cb(n)-k)G}{A} \leqslant p'_k(1) \leqslant \frac{(cb(n)+M-k)G}{A}.$$

Proof. The special case when all $g_i = 1$ is a well-known result in renewal theory (see, e.g., Gut [11, Proof of Theorem II.4.1]). The proof in [11] extends without difficulty to our more general cost functions, but we give an elementary proof to keep this article self-contained.

We will prove the upper bound; the proof of the lower bound is similar. We use decreasing induction on $k = cb(n) + M$ down to 0. For $cb(n) \leqslant k \leqslant cb(n) + M$, $p_k(z) = 1$, and the result is obvious.

Now assume $k < cb(n)$ and the result holds for all k' such that $k < k' \leqslant cb(n) + M$. By (2) and Lemma 2.1,

$$p'_k(1) = \sum_{i=1}^{n} [g_i + p'_{k+a_i}(1)]/n$$

and

$$\begin{aligned} hp'_k(1) &= G + \sum_{i \in H} p'_{k+a_i}(1) \\ &\leqslant G + \sum_{i \in H} \frac{(cb(n) + M - k - a_i)G}{A} \text{ by the induction assumption} \\ &= \frac{h(cb(n) + M - k)G}{A}. \qquad \square \end{aligned}$$

From Lemmas 2.1 and 2.2, taking $k = 0$,

Theorem 2.1.

$$\mathrm{pr}(\mathbf{T}_0 < \infty) = 1$$

and

$$\frac{cb(n)G}{A} \leqslant \mathbf{E}[\mathbf{T}_0] \leqslant \frac{(cb(n) + M)G}{A}.$$

For Boolean valued $a_1, \ldots, a_n$,

$$\mathbf{E}[\mathbf{T}_0] = \frac{cb(n)G}{A}.$$

In particular, the average stopping time of the random walk, or equivalently, the number of iterations of Algorithm 1, is between $cb(n)n/A$ and $(cb(n) + M)n/A$. We will now derive an upper bound on the variance of the stopping time, which will be used to obtain an upper bound on c in Algorithm 1. In the remainder of this section, we take $g_i = 1$ for $i = 1, \ldots, n$, so $\mathbf{T}_k$ will be the stopping time of the random walk when started at k.

Lemma 2.3. *For* $k \leqslant cb(n) + M$,

$$p''_k(1) \leqslant \frac{(cb(n) + M - k)^2 n^2}{A^2} + \frac{(cb(n) + M - k)(\sum_{i=1}^n a_i^2)n^2}{A^3} - \frac{2(cb(n) - k)n}{A}.$$

Proof. We again use decreasing induction on k, and the result is obvious for $cb(n) \leqslant k \leqslant cb(n) + M$.

Now assume $k < cb(n)$ and the inequality holds for all k' such that $k < k' \leqslant cb(n) + M$. Then

$$\begin{aligned} p''_k(1) &= \sum_{i=1}^n [2p'_{k+a_i}(1) + p''_{k+a_i}(1)]/n \quad \text{by (3)}, \\ &= 2p'_k(1) - 2 + \sum_{i=1}^n p''_{k+a_i}(1)/n \quad \text{by (2) and Lemma 2.1.} \end{aligned}$$

Therefore,

$$\begin{aligned}
hp_k''(1) \leqslant & \frac{2(cb+M-k)n^2}{A} - 2n \\
& + \sum_{i\in H}\left[\frac{(cb(n)+M-k-a_i)^2n^2}{A^2} + \frac{(cb(n)+M-k-a_i)(\sum_{i=1}^n a_i^2)n^2}{A^3}\right. \\
& \left. - \frac{2(cb(n)-k-a_i)n}{A}\right] \quad \text{by Lemma 2.2 and the induction hypothesis} \\
= & \frac{2(cb(n)+M-k)n^2}{A} - 2n + \frac{h(cb(n)+M-k)^2n^2}{A^2} \\
& + \frac{(\sum_{i=1}^n a_i^2)n^2}{A^2} - \frac{2(cb(n)+M-k)(\sum_{i=1}^n a_i)n^2}{A^2} \\
& + \frac{h(cb(n)+M-k)(\sum_{i=1}^n a_i^2)n^2}{A^3} - \frac{(\sum_{i=1}^n a_i)(\sum_{i=1}^n a_i^2)n^2}{A^3} \\
& - \frac{2(cb(n)-k)n}{A} + \frac{2(\sum_{i=1}^n a_i)n}{A} \\
= & \frac{h(cb(n)+M-k)^2n^2}{A^2} + \frac{h(cb(n)+M-k)(\sum_{i=1}^n a_i^2)n^2}{A^3} \\
& - \frac{2h(cb(n)-k)n}{A}. \qquad \square
\end{aligned}$$

From Lemmas 2.2 and 2.3, taking $m = \mathbf{T}_0$.

Theorem 2.2.

$$\mathbf{Var}[m] \leqslant \frac{(2cb(n)+M)Mn^2}{A^2} + \frac{(cb(n)+M)(\sum_{i=1}^n a_i^2)n^2}{A^3} + \frac{(M-cb(n))n}{A}.$$

A weaker upper bound on $\mathbf{Var}[m]$ is given in Theorem III.9.1(ii) of [11].

Theorem 2.3. *For $0 \leqslant p < 1$ and $\varepsilon > 0$, there exists c such that for all n and $a_1, \ldots, a_n$,* $\mathrm{pr}(|nS/m - A| \leqslant \varepsilon A) \geqslant p$.

Proof. By Chebyshev's inequality, for any $\gamma > 0$,

$$\begin{aligned}
& \mathrm{pr}\left(|m - \mathbf{E}[m]| \geqslant \frac{\gamma cb(n)n}{A}\right) \\
& \leqslant \frac{\mathbf{Var}[m]A^2}{\gamma^2c^2b(n)^2n^2} \leqslant \frac{(2cb(n)+M)M}{\gamma^2c^2b(n)^2} + \frac{(cb(n)+M)\sum_{i=1}^n a_i^2}{\gamma^2c^2b(n)^2A} + \frac{(M-cb(n))A}{\gamma^2c^2b(n)^2n},
\end{aligned}$$

by Theorem 2.2. Since $M < b(n)$, $\sum_{i=1}^{n} a_i^2 \leqslant b(n)A$, and we can assume $c > 1$,

$$\mathrm{pr}\left(|m - \mathbf{E}[m]| \geqslant \frac{\gamma c b(n) n}{A}\right) \leqslant \frac{2c+1}{\gamma^2 c^2} + \frac{c+1}{\gamma^2 c^2} = \frac{3c+2}{\gamma^2 c^2}. \tag{4}$$

That is,

$$\mathrm{pr}\left(\mathbf{E}[m] - \frac{\gamma c b(n) n}{A} \leqslant m \leqslant \mathbf{E}[m] + \frac{\gamma c b(n) n}{A}\right) \geqslant 1 - \frac{3c+2}{\gamma^2 c^2}.$$

By Theorem 2.1

$$\frac{cb(n)n}{A} \leqslant \mathbf{E}[m] \leqslant \frac{cb(n)n(1+1/c)}{A}.$$

Therefore,

$$\mathrm{pr}\left(\frac{cb(n)n(1-\gamma)}{A} \leqslant m \leqslant \frac{cb(n)n(1+1/c+\gamma)}{A}\right) \geqslant 1 - \frac{3c+2}{\gamma^2 c^2}.$$

Also, with probability 1,

$$cb(n) \leqslant S < (c+1)b(n).$$

Therefore,

$$\mathrm{pr}\left(\frac{A}{1+1/c+\gamma} \leqslant \frac{nS}{m} \leqslant \frac{A(1+1/c)}{1-\gamma}\right) \geqslant 1 - \frac{3c+2}{\gamma^2 c^2},$$

and the theorem will follow if we can find γ and c such that

$$1 - \varepsilon \leqslant \frac{1}{1+1/c+\gamma},$$

$$\frac{1+1/c}{1-\gamma} \leqslant 1 + \varepsilon$$

and

$$\frac{3c+2}{\gamma^2 c^2} \leqslant 1 - p.$$

The first inequality can be satisfied by taking any $\gamma \leqslant \varepsilon/2$ and $c \geqslant 2/\varepsilon$. Then, having fixed γ at $\varepsilon/2$, and assuming without loss of generality that $\varepsilon < 1$, we then choose c large enough to satisfy the second and third inequalities. □

More specific bounds on c can be obtained, but they are cumbersome. However, there is a simple approximation for small ε. As $\varepsilon \to 0$, $\gamma \sim \varepsilon$, and the right-hand side of Eq. (4) is asymptotic to $3/(\gamma^2 c)$. Therefore, it suffices to take

$$\frac{3}{\varepsilon^2 c} < 1 - p,$$

i.e.,

$$c > \frac{3}{(1-p)\varepsilon^2}.$$

When p is close to 1, $1-p$ is approximately $2(1-\sqrt{p})$, and our bound is larger than the bound in the claim by a factor of about 1.5. If, in addition, the a_i's are Boolean, then the right-hand side of (4) is asymptotic to $1/(\gamma^2 c)$, and it suffices to take $c > 1/((1-p)\varepsilon^2)$.

3. Comparison to other sampling algorithms

Haas and Swami [12] proposed a variation on Algorithm 1. They pointed out that the stopping condition $S \geqslant cb(n)$ can be overly conservative if $b(n)$ is actually much larger than $\max(a_1, \ldots, a_n)$. Their stopping condition is based on the Central Limit Theorem. Let S_m be the value of S after m iterations of the **repeat**…**until** loop in Algorithm 1. Then, as m gets large,

$$\mathrm{pr}\left(\left|\frac{nS_m}{m} - A\right| \leqslant \varepsilon A\right) = \mathrm{pr}\left(\left|\frac{S_m - m\mu}{\sqrt{m}\sigma}\right| \leqslant \frac{\sqrt{m}\varepsilon\mu}{\sigma}\right) \approx 2\Phi\left(\frac{\sqrt{m}\varepsilon\mu}{\sigma}\right) - 1, \tag{5}$$

where $\mu = A/n$ is the expectation of a randomly chosen a_i, $\sigma^2 = \sum_{i=1}^{n} [a_i - \mu]^2/n$ is its variance, and

$$\Phi(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx$$

is the standard normal distribution function.

Let

$$\zeta_p = \Phi^{-1}((1+p)/2)$$

and

$$m = \frac{\zeta_p^2 \sigma^2}{\varepsilon^2 \mu^2}.$$

Then, as ε approaches 0, m gets arbitrarily large and the error term in (5) will go to 0. Therefore for sufficiently small ε, the estimate nS_m/m will be within εA of A with probability p.

Of course, μ and σ are not known a priori. The algorithm of Haas and Swami makes successive estimates of μ and σ as it samples $\{a_1, \ldots, a_n\}$. It can be shown that with high probability, the mth estimates μ_m and σ_m approach their true values, and when $\sigma_m > 0$ and $m > \zeta_p^2 \sigma_m^2/(\varepsilon^2 \mu_m^2)$, the probability that the estimate nS_m/m has the desired accuracy is at least p. Further, their method is asymptotically efficient, meaning that as ε gets small, their algorithm takes just enough samples to guarantee the desired accuracy and confidence.

Note that Theorem 2.3 does not make any assumptions about the asymptotics of the parameters, while Haas and Swami assume that $\varepsilon \to 0$. As they point out, if ε is fixed, then it is possible that the error term in (5) is quite large, and the probability that the estimate is within the desired accuracy range could be less than p. Specifically, Berry [3], and Blum and Rosenblat [4] have shown that the error term is bounded by

$$\frac{4\sum_{i=1}^{n} |a_i - \mu|^3}{\sqrt{m}\sigma^3},$$

which can be quite large if we know only that $a_1, \ldots, a_n \leqslant n$. This is the case when the size of a binary relation on $\{1, \ldots, n\}$ is being estimated. The following example illustrates this possibility.

For $i = 1, \ldots, n$ let

$$a_i = \begin{cases} 0 & \text{if } i<n \text{ and } i \text{ even}, \\ 1 & \text{if } i<n \text{ and } i \text{ odd}, \\ n & \text{if } i = n. \end{cases}$$

Then, with probability greater than $(1 - 1/n)^n \approx 1/e$, a_n is not chosen in the first n iterations of the **repeat**…**until** loop. If this happens, then the estimates of both μ and σ will rapidly approach $1/2$, which almost surely will result in a stopping time less than $\omega(n)$, for any function ω that increases monotonically to ∞. More specifically, if ζ_p is sufficiently large, i.e., p is sufficiently close to 1 or ε is sufficiently small, then with probability close to $1/e$, the algorithm will return an estimate close to $n/2$, i.e., the probability that it is within εA of A will not be p.

The following algorithm, which was suggested by an anonymous referee, is faster than adaptive sampling in certain cases, even though it is non-adaptive (the number of samples is predetermined). By Hoeffding's inequality [16], if Y_m is the average of m independent identically distributed random variables with mean μ, lower bound a, and upper bound b, then

$$\mathrm{pr}(|Y_m - \mu| \geqslant t) \leqslant 2e^{-2mt^2/(b^2-a^2)}.$$

In our context,

$$\mathrm{pr}(|nS/m - A| \leqslant \varepsilon A) \geqslant 1 - 2e^{-2m\varepsilon^2\mu^2/b(n)^2},$$

where $\mu = A/n$ as before. If we set the right side equal to p and solve for m,

$$m = \ln\left(\frac{2}{1-p}\right)\frac{b(n)^2}{2\varepsilon^2\mu^2}.$$

Since $\mu \geqslant 1/n$, we can estimate A by executing the body of the **repeat**…**until** loop in Algorithm 1 $\lceil T \rceil$ times, where

$$T = \ln\left(\frac{2}{1-p}\right)\frac{n^2 b(n)^2}{2\varepsilon^2}.$$

From Theorem 2.1 and our estimate of c following Theorem 2.3,

$$\mathbf{E}[\mathbf{T}_0] \geqslant \frac{cb(n)}{\mu} \approx \frac{3b(n)}{(1-p)\varepsilon^2\mu}$$

for small ε. Thus, for fixed n, $T \ll \mathbf{E}[\mathbf{T}_0]$ for small ε and p sufficiently close to 1. Under these conditions, the Hoeffding algorithm will achieve the accuracy and confidence of adaptive sampling much quicker. The essential reason for this is the use of an exponential inequality instead of the polynomial Chebyshev inequality in Theorem 2.3.

A very different method for estimating the sizes of relations is a Monte-Carlo algorithm due to Cohen [6]. It repeats the following process some predetermined number of times, say t. In each iteration $s = 1, \ldots, t$, it performs the following two steps:

1. It randomly assigns a ranking $r_s(i) \in [0, \infty)$ to each $i \in \{1, \ldots, n\}$.
2. Then, for each $i \in \{1, \ldots, n\}$, it computes $\rho_s(i) = \min(r_s(j) : j \in R_i)$.

After the final iteration, it estimates each $|R_i|$. There are several ways of doing this. Let $\hat{S}_i$ be the estimate for $|R_i|$. Using the average of the t samples:

$$\hat{S}_i = \frac{t}{\sum_{s=1}^{t} \rho_s(i)},$$

or, letting $\tilde{\rho}(i)$ be the $\lfloor t(1-1/e) \rfloor$-smallest value in $\{\rho_s(i) : 1 \leqslant s \leqslant t\}$,

$$\hat{S}_i = \frac{1}{\tilde{\rho}(i)}.$$

The idea behind this approach is that there is a strong correlation between $\rho_s(i)$ and $|R_i|$. The smaller the $\rho_s(i)$ is, the larger the $|R_i|$ is likely to be. As t increases, the probability that $\hat{S}_i$ is within a desired accuracy also increases. Specifically,

$$\mathrm{pr}(|\hat{S}_i - |R_i|\,| \leqslant \varepsilon |R_i|) = e^{-\Omega(\varepsilon t)}$$

for each $i = 1, \ldots, n$, and

$$\mathrm{pr}\left(\left|\sum_{i=1}^{n} \hat{S}_i - |R|\right| \leqslant \varepsilon |R|\right) = e^{-\Omega(\varepsilon t)}.$$

The efficiency of this algorithm depends on the speed of computing all the $\rho_s(i)$. At present, no efficient method applicable to all queries is known. An important case where there is a fast algorithm is the transitive closure of a binary relation. Letting $G = \langle \{1, \ldots, n\}, E \rangle$ be a graph on n vertices with $m = |E|$ edges and R_i be the set of vertices reachable from i by a path in G, $R = \bigcup_{i=1}^{n} R_i$ is the transitive closure of E. Cohen shows how to estimate each $|R_i|$ and $|R|$ in time $\Theta(n+m)$. Using adaptive sampling, Lipton and Naughton [18] show how to estimate $|R|$ in time $O(n\sqrt{m})$, and therefore Cohen's method is more efficient on general graphs.

However, if it is known that the sizes of the reachability sets are small, then adaptive sampling can estimate the transitive closure faster than the Monte-Carlo method. For example, suppose it is known that the sizes $a_1, \ldots, a_n$ of the reachability sets are bounded by k. The Monte-Carlo algorithm will run in time $\Omega(n)$, but if the adaptive sampling algorithm uses $b(n) = k$, then it will take only $O(k^2)$ time to reach the same level of accuracy and confidence. To see this, let $a_{j_1}, \ldots, a_{j_m}$ be the choices made by Algorithm 1. Then the total cost is

$$\begin{aligned} O\left(\sum_{t=1}^{m} a_{j_t}^2\right) &= O\left(k \sum_{t=1}^{m} a_{j_t}\right) \\ &= O(k(c+1)b(n)) \\ &= O(k^2). \end{aligned}$$

Thus, if $k^2 = o(n)$, adaptive sampling will be faster.

4. Estimating queries

A natural application of adaptive sampling is the estimation of queries, i.e., relations, on a finite structure. That is, we partition the relation R into disjoint sets and estimate $|R|$ by computing the

sizes of randomly chosen partitions. Although the partitioning is often determined by the value of the first coordinate of the tuples in the relation (as outlined in the Introduction), there is no reason to expect that this is the optimal partitioning in all cases. For example, in estimating the size of a join [12], it is more efficient to use the join key. Those relations in the join operation that do not have an index on the key are sampled, but the tuples contributed by those relations that do have an index on the key are counted by using the indices.

Let us consider possible ways of partitioning an r-ary relation R. For any $j \in \{1, \ldots, r\}$, and $I \subseteq \{1, \ldots, r\}$ of size j, R can be partitioned into n^j sets indexed by elements in $\{1, \ldots, n\}^j$. Each partition $R_{i_1,\ldots,i_j}$, where $(i_1, \ldots, i_j) \in \{1, \ldots, n\}^j$, is the subset of R consisting of the r-tuples in R that agree with $(i_1, \ldots, i_j)$ on the coordinates in I. (Note that when $j = r$, $R_{i_1,\ldots,i_r} = \emptyset$ or $\{(i_1, \ldots, i_r)\}$.) Then $|R|$ can be estimated by sampling $|R_{i_1,\ldots,i_j}|$ for random $(i_1, \ldots, i_j)$.

Replacing n in Theorem 2.1 by n^j and $b(n)$ by n^{r-j},

Theorem 4.1. *Let the cost of computing $|R_{i_1,\ldots,i_j}|$ on structures of size n be $g_I(n)$. Then for any $p \in [0, 1)$ and $\varepsilon > 0$, the average time required by adaptive sampling to produce an estimate nS/m such that*

$$\mathrm{pr}(|nS/m - |R|| \leqslant \varepsilon |R|) \geqslant p$$

is $O(n^r g_I(n)/|R|)$.

Thus, the efficiency of adaptive sampling can depend on the choice of j and I. In general, the fastest way to compute $|R_{i_1,\ldots,i_j}|$ may be to examine each of the n^{r-j} members of $\{1, \ldots, n\}^r$ that agree with $(i_1, \ldots, i_j)$ on I, and decide whether they belong to R. Letting $g(n)$ be the cost of deciding membership in R, the average time to estimate $|R|$ becomes $O(n^{2r-j} g(n)/|R|)$, which is minimal for $j = r$, i.e., $O(n^r g(n)/|R|)$. Since $|R|$ can be computed exactly in time $O(n^r g(n))$, adaptive sampling can be significantly faster if $|R|$ is large. However, even when $|R|$ is not large, the knowledge that $|R|$ is small relative to n^r can be useful, and it can be obtained by estimating $|\bar{R}|$, where $\bar{R}$ is the complement of R. A modified version of adaptive sampling can estimate $|\bar{R}|$ in parallel with the estimate of $|R|$. For example, Algorithm 1 would be modified as follows:

Algorithm 2.

$S \leftarrow 0$
$T \leftarrow 0$
$m \leftarrow 0$
repeat
  do $(i_1, \ldots, i_r) \leftarrow \textsc{Random}(1, \ldots, n^r)$
    $S \leftarrow S + |R_{i_1,\ldots,i_r}|$
    $T \leftarrow T + n^{r-j} - |R_{i_1,\ldots,i_r}|$
    $m \leftarrow m + 1$
until $S \geqslant cn^{r-j}$ **or** $T \geqslant cn^{r-j}$
return $n^r S/m$, $n^r T/m$

Since at least one of $|R|$ and $|\bar{R}|$ is $\Omega(n^r)$, we have

Theorem 4.2. *Let the cost of deciding whether* $(i_1, \ldots, i_r) \in R$ *on structures of size* n be $g(n)$. *Then for any* $p \in [0, 1)$ *and* $\varepsilon > 0$, *the average time required by Algorithm* 2 *to produce estimates* $n^r S/m$ *and* $n^r T/m$ *such that*

$$\text{pr}(|n^r S/m - |R|| \leqslant \varepsilon |R| \text{ or } \text{pr}(|n^r T/m - |\bar{R}|| \leqslant \varepsilon |\bar{R}|) \geqslant p$$

is $O(g(n))$.

The error bound $|n^r T/m - |\bar{R}|\,| \leqslant \varepsilon |\bar{R}|$ implies $|n^r S/m - |R|\,| \leqslant \varepsilon n^r$, so Algorithm 2 guarantees an estimate of $|R|$ with error bounded by εn^r.

4.1. First-order queries

The two theorems above are quite general. Here, we examine the special case when R is defined by a first-order formula. That is, let $\phi(x_1, \ldots, x_r)$ be a first-order formula, and for any structure $\mathfrak{A}$ appropriate to ϕ, let $R_\phi = \{(a_1, \ldots, a_r) \in \{1, \ldots, n\}^r : \mathfrak{A} \models \phi(a_1, \ldots, a_r)\}$. In general, the fastest way to evaluate $\mathfrak{A} \models \phi(a_1, \ldots, a_r)$ takes time $O(n^k)$, where k is the quantifier depth of ϕ. Therefore, the average times in Theorems 4.1 and 4.2 are $O(n^{r+k}/|R|)$ and $O(n^k)$, respectively. But there are faster evaluation algorithms for first-order formulas on finite structures of bounded degree. We will show how they yield fast estimation algorithms on such structures. They rely on characterizations of first-order properties in terms of local "neighborhoods" of elements, first used by Gaifman [10] and Hanf [14] on arbitrary structures. Let $\mathfrak{A} = \langle \{1, \ldots, n\}, P_1, \ldots, P_k \rangle$, where n is a natural number, and each $P_i \subseteq \{1, \ldots, n\}^{p_i}$. The Gaifman graph $\mathcal{G}(\mathfrak{A})$ of $\mathfrak{A}$ is $\langle \{1, \ldots, n\}, E \rangle$, where

$$E = \{(a, b) \in \{1, \ldots, n\} : \text{ there exists } i \in \{1, \ldots, k\}$$

$$\text{and } (a_1, \ldots, a_{p_i}) \in P_i \text{ such that } \{a, b\} \subseteq \{a_1, \ldots, a_{p_i}\}\}.$$

Letting δ be the usual notion of distance in the undirected graph $\mathcal{G}(\mathfrak{A})$, for any natural number d and $a_1, \ldots, a_i \in \{1, \ldots, n\}$, the d-neighborhood of $a_1, \ldots, a_i$ in $\mathfrak{A}$ is

$$N_d(a_1, \ldots, a_i) = \bigcup_{j=1}^{i} \{b \in \{1, \ldots, n\} : \delta(a_j, b) \leqslant d\}.$$

It is easy to define the property $\delta(x, y) \leqslant d$ by a formula in the first-order language of $P_1, \ldots, P_k$. A formula is said to be d-local if its quantifiers are of the form $(\forall x \in N_d(y_1, \ldots, y_i))$ or $(\exists x \in N_d(y_1, \ldots, y_i))$. Of course, every d-local formula is equivalent to a first-order formula. Further,

Theorem 4.3 (Gaifman [10]). *Every first-order formula is equivalent to a Boolean combination of d-local formulae for some d, and sentences of the form*

$$\exists x_1 \cdots x_i \left(\bigwedge_{j=1}^{i} \alpha(x_j) \wedge \bigwedge_{1 \leqslant h \neq j \leqslant i} \delta(x_h, x_j) > 2m \right) \tag{6}$$

for some m and m-local formula α.

Assuming ϕ is in the form described by Theorem 4.3, let us call sentences of the form (6) background sentences of ϕ. The intention is that in order to compute or estimate $|R_\phi|$, one first determines whether the background sentences are satisfied. Let $\phi'(x_1, \ldots, x_r)$ be the result of replacing all the background sentences of ϕ with their truth values in the model $\mathfrak{A}$. Then ϕ' is a d-local formula, and $R_\phi = R_{\phi'}$ on $\mathfrak{A}$.

A class of structures is said to be of bounded degree if there is some b such that for every structure in the class, its Gaifman graph has degree bounded by b. In such a class, for every i and d, there is an upper bound on the sizes of the d-neighborhoods of all i-tuples of elements in the members of the class, and therefore each d-neighborhood is isomorphic to some finite structure. As a consequence, on classes of structures of bounded degree, a d-local formula $\psi(x_1, \ldots, x_r)$ of quantifier depth k is equivalent to a disjunction of formulas, each describing an isomorphism type of the dk-neighborhood around $x_1, \ldots, x_r$. Thus, the truth of $\psi(a_1, \ldots, a_r)$ for any assignment of $a_1, \ldots, a_r$ to $x_1, \ldots, x_r$ can be determined in constant time.

The background sentences are simply Boolean queries, and not statistical queries. In any event, they can be evaluated quickly on structures of bounded degree.

Lemma 4.1. *On a class of finite structures of bounded degree, background sentences can be evaluated in time $O(n)$.*

Proof. Referring to (6), for any element a, $\mathfrak{A} \models \alpha(a)$ can be evaluated in constant time as described above. There is a constant k such that each a can be within a distance $2m$ of k elements b such that $\mathfrak{A} \models \alpha(b)$. Therefore to determine if (6) is true, it suffices to determine the set A of those a for which $\mathfrak{A} \models \alpha(a)$. If $|A| \geqslant ki$, then (6) is true. If not, but there are i elements in A all separated by a distance greater than $2m$, then (6) is still true. Determining A can be done in time $O(n)$, and checking the above conditions can be done in constant time. □

Since any first-order property is computable in AC^0, any sentence can be evaluated in constant parallel time with polynomially many processors. Counting is not in AC^0, so evaluation of the size of a query defined by a formula with free variables appears to be more difficult. But since a local formula can be evaluated in constant time on structures of bounded degree, we get

Theorem 4.4. *On classes of structures of bounded degree, for any $p \in [0, 1)$ and $\varepsilon > 0$ and any first-order local formula $\psi(x_1, \ldots, x_r)$, adaptive sampling can produce an estimate $n^r S/m$ such that*

$$\mathrm{pr}(|n^r S/m - |R_\psi|| \leqslant \varepsilon |R_\psi|) \geqslant p$$

in average time $O(n^r/|R_\psi|)$.

Theorem 4.5. *On classes of structures of bounded degree, for any* $p \in [0,1)$ *and* $\varepsilon > 0$ *and any first-order local formula* ψ*, adaptive sampling can produce estimates* $n^r S/m$ *and* $n^r T/m$ *such that*

$$\mathrm{pr}(|n^r S/m - |R_\psi|| \leqslant \varepsilon |R_\psi| \text{ or } \mathrm{pr}(|n^r T/m - |\bar{R}_\psi|| \leqslant \varepsilon |\bar{R}_\psi|) \geqslant p$$

in constant average time.

Corollary 4.1. *On classes of structures of bounded degree, for any* $p \in [0,1)$ *and* $\varepsilon > 0$ *and any first-order formula* ϕ*, adaptive sampling can produce estimates* $n^r S/m$ *and* $n^r T/m$ *such that*

$$\mathrm{pr}(|n^r S/m - |R_\phi|\,| \leqslant \varepsilon |R_\phi| \text{ or } \mathrm{pr}(|n^r T/m - |\bar{R}_\phi|\,| \leqslant \varepsilon |\bar{R}_\phi|) \geqslant p$$

in constant average parallel time.

5. Future work

The problem of whether all first-order queries on arbitrary structures have fast estimation algorithms has already been mentioned. There is also the question of whether queries in more powerful logics can be estimated rapidly. For many of these logics, including transitive closure logic, the locality properties described by Gaifman and Hanf fail. Estimating query sizes, even on structures of bounded degree appears difficult in these cases. For example, consider a structure on $\{1, \ldots, n\}$ whose relations include successor $\{(i, i+1)\colon 1 \leqslant i < n\}$. One application of transitive closure on successor can define $\leqslant$, and once this is done, other applications of transitive closure can define any NLOGSPACE property. Then, as was shown for NSPACE in the Introduction, if there were fast estimation algorithms for transitive closure queries, for every NLOGSPACE language L, there would be fast probabilistic algorithms that decide membership in L with arbitrarily high confidence. This still leaves open the possibility that a *single* application of transitive closure to a first-order formula could be estimated quickly on structures of bounded degree. One of the Lipton–Naughton papers [17] studied fast estimation algorithms for the usual transitive closure relation on a graph, which is a special case of this problem. It seems likely that this algorithm, combined with Gaifman's Theorem 4.3, would extend to the more general case.

Another potential area of investigation would be to consider broader classes of structures and logics that still obey locality conditions similar to Gaifman's and Hanf's. One example comes from a recent article by Frick and Grohe [9]. They developed fast algorithms for evaluating first-order sentences on locally tree-decomposable graphs, which are a generalization of bounded degree graphs.

An article by Hella et al. [15] shows that notions of locality extend to logics augmented by counting quantifiers. Since we are concerned here with approximate counting, a natural generalization would be to consider approximate analogues of counting quantifiers. Let $\psi_1(x, u_1, \ldots, u_k)$ and $\psi_2(y, v_1, \ldots, v_l)$ be formulas. The analogue of the Rescher quantifier would be $Q_{\mathscr{R},\varepsilon}$, where, for any structure $\mathfrak{A}$ on $\{1, \ldots, n\}$ and $i_1, \ldots, i_k, j_1, \ldots, j_l \in \{1, \ldots, n\}$,

$$\mathfrak{A} \models Q_{\mathscr{R},\varepsilon} xy(\psi_1(x, i_1, \ldots, i_k), \psi_2(y, j_1, \ldots, j_l))$$

if and only if

$$|\{b\colon \mathfrak{A} \models \psi_1(b, i_1, \ldots, i_k)\}| \leqslant (1+\varepsilon)|\{c\colon \mathfrak{A} \models \psi_2(c, j_1, \ldots, j_l)\}|.$$

The analogue of the Härtig quantifier would be $Q_{\mathscr{H},\varepsilon}$, where

$$\mathfrak{A} \models Q_{\mathscr{H},\varepsilon} xy(\psi_1 i(x, i_1, \ldots, i_k), \psi_2(y, j_1, \ldots, j_l))$$

if and only if

$$||\{b\colon \mathfrak{A} \models \psi_1(b, i_1, \ldots, i_k)\}| - |\{c\colon \mathfrak{A} \models \psi_2(c, j_1, \ldots, j_l)\}|| \leqslant \varepsilon|\{b\colon \mathfrak{A} \models \psi_1(b, i_1, \ldots, i_k)\}|.$$

One question related to our results is whether the truth of formulas in this logic can be evaluated quickly, at least to a given degree of confidence.

Acknowledgments

I thank Dr. Qiming Liu [19] for many stimulating conversations which exposed me to this area, Dr. Peter Haas of the IBM Almaden Research Center for several very helpful email responses, and an anonymous referee for a number of suggestions, in particular the use of Hoeffding bounds for sampling.

References

[1] S. Abiteboul, K. Compton, V. Vianu, Queries are easier than you thought (probably). in: Proceedings of the ACM Symposium on Principles of Database Systems, 1992, pp. 23–32.
[2] W. Bartky, Multiple sampling with constant probability, Ann. Math. Statist. 14 (1943) 363–377.
[3] A. Berry, The accuracy of the Gaussian approximation to the sum of independent variates, Trans. Amer. Math. Soc. 49 (1941) 122–136.
[4] J. Blum, J. Rosenblat, Probability and Statistics, W.B. Saunders Company, Philadelphia, 1972.
[5] H. Chernoff, A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations, Ann. Math. Statist. 23 (1952) 493–507.
[6] E. Cohen, Size-estimation framework with applications to transitive closure and reachability, J. Comput. System Sci. 55 (1997) 441–453.
[7] H.-D. Ebbinghaus, J. Flum, Finite Model Theory, Springer, New York, Heidelberg, 1995.
[8] W. Feller, An Introduction to Probability Theory and Its Applications, Vol. 1, Wiley, New York, 1968.
[9] M. Frick, M. Grohe, Deciding first-order properties of locally tree-decomposable graphs, in: Proceedings of the 26th ICALP, 1999, pp. 105–135.
[10] H. Gaifman, On local and nonlocal properties, in: J. Stern (Ed.), Logic Colloquium '81, North-Holland, Amsterdam, 1982, pp. 105–135.
[11] A. Gut, Stopped Random Walks: Limit Theorems and Applications, Springer, New York, Berlin, Heidelberg, 1988.
[12] P. Haas, A. Swami, Sequential sampling, procedures for query size estimation, IBM Research Report, RJ 9101 (80915), 1992.
[13] T. Hagerup, C. Rüb, A guided tour of chernoff bounds, Inf. Process. Lett. 33 (1989/90) 305–308.
[14] W. Hanf, Model-theoretic methods in the study of elementary logic, in: J. Addison, L. Henkin, A. Tarski (Eds.), The Theory of Models, North Holland, Amsterdam, 1965, pp. 132–145.
[15] L. Hella, L. Libkin, J. Nurmonen, Notions of locality and their logical characterizations over finite models, J. Symbolic Logic 64 (1999) 1751–1773.

[16] W. Hoeffding, Probability inequalities for sums of bounded random variables, J. Amer. Statist. Assoc. 58 (1963) 13–30.
[17] R.J. Lipton, J.F. Naughton, Estimating the size of generalized transitive closures, in: Proceedings of the 15th International Conference on Very Large Databases, 1989, pp. 165–172.
[18] R.J. Lipton, J.F. Naughton, Query size estimation by adaptive sampling, J. Comp. Systems Sci. 51 (1995) 18–25.
[19] Q. Liu, On algorithms for database query size estimation, Master's Thesis, Clarkson University, 1998.
[20] F. Olken, D. Rotem, Random Sampling from Databases—A Survey. Lawrence Berkeley Laboratories Tech. Report LBL-32755, 1994.
[21] A. Wald, On cumulative sums of random variables, Ann. Math. Statist. 15 (1944) 283–296.
[22] A. Wald, Sequential Analysis, Wiley, New York, 1947.
[23] O. Watanabe, Simple sampling techniques for discovery science, IEICE Trans. E83-D (2000) 19–26.