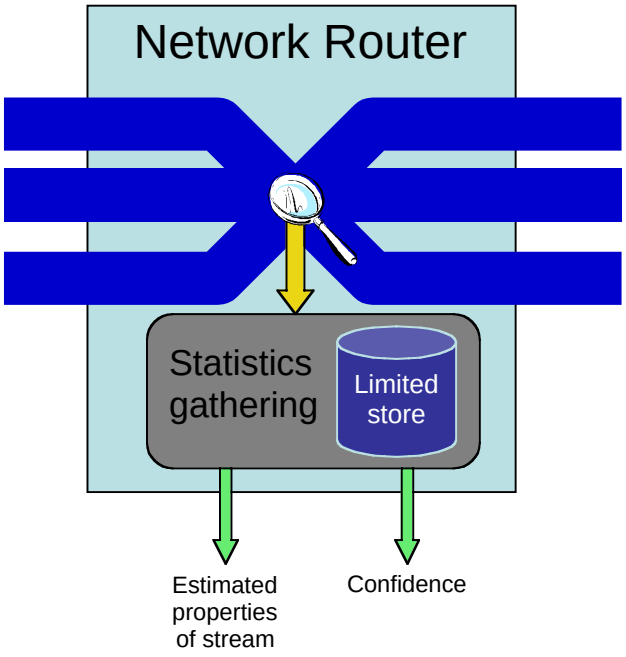
Monitoring Network Routing and Traffic with Low Space
MIT2001-09

Progress Report: November 1, 2001—December 31, 2001

Erik D. Demaine

Project Overview

The goal of this research is to develop algorithms that extract essential characteristics of network traffic streams passing through routers, such as the most common destination address, subject to a limited amount of memory about previously seen packets. Such characteristics are essential for designing accurate models and developing a general understanding of Internet traffic patterns, which are critical for such applications as efficient network routing, caching, prefetching, information delivery, and network upgrades.



We proposed to design and analyze efficient algorithms and data structures that have provable guarantees on the quality of gathered statistics based on weak assumptions on the traffic distribution. We proposed to consider the range from deterministic, fully guaranteed algorithms (which we expected to be extremely difficult if not impossible) to randomized, probabilistically guaranteed algorithms (which are more powerful). On the other hand, we proposed to prove lower bounds on the possible success of any algorithm.

Progress Through December 2001

In the past two months, we have made major progress in collaboration with Prof. Alejandro López-Ortiz and Prof. Ian Munro of the University of Waterloo. Our research has focused on perhaps the most practical and most difficult statistic to gather from a stream of data while using little memory: finding the k most popular packet destination addresses, for a desired value of k (smaller than the amount of memory available). This problem arose in an applied setting while Prof. López-Ortiz was Director of Core Research at Internap Network Services

Corporation. Our work on this problem can be divided into four main categories: (1) designing practical models of allowed computation, (2) designing various levels of reasonable assumptions on network traffic distributions, (3) developing algorithms under these models, and (4) proving lower bounds under these models. We have made breakthrough advances in all four of these categories.

1. Practical Models of Computation

As shown in the figure above, our basic model of computation is that a ***statistics gathering*** process watches a stream of *n* packets passing through an Internet router or similar device. The stream is rapid, so the process can make only one pass through the data, and furthermore can perform little computation per packet. Specifically, we limit the amount of computation to O(1) operations per packet. The storage space available to the process is limited, but a more important limiting factor is that the ***working store*** of the process is very small: all actively used variables (e.g., counters) must fit in a small cache in order to keep up with the data stream. Thus, in some settings, we may be willing to record a significant amount of data (but still much less than one item per packet) to external storage, and make a final pass through these records at the end of the computation.

A key operation that the statistics gathering process can perform is ***counting***. The process is limited to having at most *m* active counters at any time. Each counter has an associated destination address that it ***monitors***. A counter can be incremented, decremented, or reset to monitor a different address. A representative example of how counters can be used is the following: when a packet streams by, the process can check whether its destination matches any of the *m* currently monitored addresses, and if so, increment that counter. The idea is that the destination address with the highest counter is likely to be the most popular destination address.

The primary difficulty in counting with very few counters is to know which destination addresses to monitor. If we never reset the counters and start counting newly discovered destination addresses, we may never notice the most popular destination addresses, thus never counting them and discovering their popularity. On the other hand, if we reset counters too frequently, we will not gain enough statistics to be sure which counter is significantly higher than the others.

We believe that this model of computation captures essentially the entire spectrum of possible algorithms, while capturing all of the important limiting factors in the application.

2. Network Traffic Distributions

We have developed three natural assumptions on the network traffic distributions that enable us to prove guarantees on quality. All of these models lead to interesting theoretical results which are closely related to the practical problem.

The two most general models are ***worst-case distributions***. In this context, the network traffic is essentially arbitrary, and at any moment, an imaginary adversary can choose the next packet's destination address. Algorithms in this model are difficult but surprisingly turned out to be possible. There are two subtly different versions of the model. In the ***omniscient adversary*** model, the adversary knows everything about the algorithm's execution, and can choose the packet sequence to be the absolute most difficult. In the slightly less powerful but highly natural ***oblivious adversary*** model, the adversary knows the entire algorithm, but does not

know the results of any random coin tosses made by the algorithm. Thus the algorithm can hope to win over the adversary with high probability by using random bits.

Of course, these worst-case models are overly pessimistic, and limit the provable strength of any algorithm. Fortunately, real traffic is not worst-case, but rather follows some sort of distribution. A natural such distribution is the ***stochastic*** model: an arbitrary probability distribution specifies the relative frequencies of the destination addresses, but in what order these addresses occur in the packet stream is uniformly random. While this model may not precisely match reality, we feel that it is sufficiently representative to lead to highly practical algorithms. (We plan to evaluate this statement experimentally.)

In any of these models, we can also assume a ***spread*** on the probabilities of the various addresses, because if all the addresses were almost equally likely (probabilities ª 1 _ *n*), every answer would be just as good. We identify the probability of the most popular destination as p_1. We assume that this probability is reasonable large, say at least 1 _ ÷*n*. Such a bound is required to make the problem interesting, and will certainly hold in practical situations.

3. Algorithms

We have designed and analyzed efficient algorithms with provable guarantees in both the stochastic model and the omniscient adversary model. We have also developed a general technique for transforming an algorithm that simply returns an answer, which might have poor quality in rare circumstances, into an algorithm that returns an answer together with a confidence measure of the answer's quality.

In the stochastic model, the basic algorithm works roughly as follows. We divide the stream into a collection of rounds, carefully sized to balance the counter-reset trade-off described in the first section. At the beginning of each round, the algorithm ***samples*** the first *m* distinct packet destination addresses, and counts their occurrences for the duration of the round. Applying Chernoff bounds on tails of probability distributions, we prove that the counts obtained during a round are close to the actual frequencies of the addresses. The *k* addresses with the maximum counter values at the end of the round are the ***winners*** for that round. If extra nonworking storage is available to the algorithm, we record these winners and their counts for a final tournament at the end of the algorithm. Otherwise, we reserve a constant fraction of the working storage for the current best winners, and only compare against those. In either case, we prove that with high probability the true frequencies of the final winners are close to the frequencies of the truly most popular addresses. The probabilities are somewhat higher when extra nonworking space is available.

The ideal choice for the size of a round in this algorithm depends on the length *n* of the stream and on the probability distribution on addresses. Of course, the algorithm does not generally know the probabilities, and may not even know for how long it will be monitoring the stream: imagine a scenario in which the statistics gathering process is running constantly, and at will a networks designer can request the current guess and confidence of the most popular addresses; as time passes, the confidence increases. To solve these problems, we harness the algorithm in an ***adaptive*** framework that gradually increases the round length until the confidence is determined to suffice. This flexible framework requires monitoring the stream for only slightly longer.

In the oblivious adversary worst-case model, we are working on adapting a similar algorithm by randomly perturbing the sizes of the rounds. The idea is that such perturbations prevent the adversary from knowing when

the actual samples occur. The probabilistic analysis of this variant is significantly more complicated, and it remains to be soon how good a bound can be achieved.

In the omniscient adversary worst-case model, where randomization is not possible, we have developed an algorithm that guarantees to find all sufficiently popular addresses: $p > 1 \_ (m + 1)$. The basic idea is to increment the counter of an address that appears on the stream, and furthermore decrement all of the other counters. Whenever a counter reaches zero, it resets and starts counting the next new address on the stream. We prove the surprising result that sufficiently popular addresses are guaranteed to remain in working memory by the end of the stream, even though they may be kicked out part way through the stream. We have also developed an efficient data structure so that, instead of changing every counter for each packet, only O(1) work is performed per packet.

4. Lower Bounds

We have proved a precise matching lower bound in the omniscient adversary worst-case model, and therefore that algorithm is optimal under this most powerful network-traffic model. In addition, we have proved that the algorithm for the stochastic model is within a constant factor of optimal. The latter result is particularly difficult, and we omit the technical description here.

Research Plan for the Next Six Months

To summarize, tremendous progress has been made in all directions of the problem. Nonetheless, several important open problems remain. The most prominent problem is to determine to what extent randomization and an oblivious adversary can allow us to improve the bounds even for the worst case. As described above, we are hopeful that a random perturbation strategy will lead to interesting results in this context.

In the future, we plan to implement the algorithms and test them on real-world data. We are currently on the lookout for appropriate students for such an experimental project, and are investigating possible sources for real-world data.