An A*-based EM Algorithm for Network Link Delay Distributions Inference

Hongjie Sun† ,

Chengdong College, Northeast Agriculture University, Harbin ,CHINA

Summary
Inference of network internal link characteristics has become an increasingly important issue for network monitor and network management. In this paper, an A*-based EM algorithm was proposed for network link delay distributions inference. We use A* algorithm to accelerate the convergence speed of EM algorithm and ant colony algorithm is used for clustering. Experiment results show the A*-based EM algorithm is faster than the traditional EM algorithm. It is also effective and suitable for solving such problem in the field of network tomography.

Key words:
network tomography; link delay distribution; Expectation-Maximization(EM) algorithm; A algorithm; ant colony*

1. Introduction

Today's Internet has evolved into large complex decentralized multilayered loosely controlled system. The heterogeneous and largely unregulated structure of Internet renders tasks such as dynamic routing, optimized service provision, service level verification and detection of anomalous/malicious behavior extremely challenging. Network monitor and management need to deal with a large number of network performance parameters, such as link loss and packet delay. The problem is compounded by the fact that one cannot rely on the collaboration of individual routers or servers to measure network traffic directly. Estimation of performance parameters can only be based upon measurements made at a limited subset of computers.

Network tomography [1] is a new concept proposed recently to solve such problem. It is a new method to estimate network performance characteristics based on a limited set of end-to-end measurements. Probing packets are sent from a source node to a destination node and their loss and delay characteristics are observed and recorded at the destination nodes, then individual link characteristics can be inferred from the end-to-end measurements.

EM algorithm is a natural approach for solving inference problem in the field of network tomography. At present, most of the EM algorithm uses the Breadth-First algorithm to search the whole state space, the operation is along a circle and goes forward the destination. Obviously, the Breadth-First algorithm can definitely find the best solution, but with the increase of states, the speed of operation will decrease sharply. Therefore, this algorithm can meet the real-time requirement of network tomography. Heuristic search is one of the most effective search techniques to cope with very large problem spaces, such as A*, IDA*, or others. In this paper, we present an improved EM algorithm based on A* algorithm. This improved algorithm is very universally applicable which can improve greatly the convergence speed and meet the real-time requirement of network delay distributions inference. Therefore, it's preferable suitable for the research on a group of network tomography problems.

The remaining sections of the paper are organized as follows. In Section Ⅱ we describe the delay model and EM algorithm. In Section Ⅲ we describe the improved EM algorithm based on A* algorithm. In Section Ⅳ we give an experiment to verify the efficiency of the improved EM algorithm. Section Ⅴ concludes the paper.

2. Link Delay Distributions Inference

In 1996, Vardi introduced the term "network tomography" to describe estimation problem. Network tomography deals with the study of estimating the internal characteristics of a network from its end-point measurements. It relies on sending probe traffic through the network periodically and making corresponding measurements at the terminal nodes. Internal network parameters are then inferred by exploiting the correlation in performance observed in multicast receivers. In this paper we focus on the estimation of link delay distributions based on information gathered from end-to-end measurements. We will restrict attention to networks whose topologies can be represented by trees whose regularity renders the inference problem computationally tractable. The definition and notation are given as follows.

The network is represented as a weighted tree $T=(V,L,D)$ comprising a set of nodes V joined by links in L. D denotes the set of weight (delay, loss, traffic and delay jitter) on each link. A packet source is located at the root node 0,

Manuscript received March 5, 2010
Manuscript revised March 20, 2010

while a set of destinations are located at the leaf nodes *R*. The interior nodes of the tree represent the branch points of the routing tree from the source to the destinations, and the links *L* are the logical links that link these branch points.

2.1 Link Delay Model

The intuition behind internal delay estimation is that closely time-spaced packets should experience the same delay on each shared link in their path, and therefore delay aberrations at different receivers must be caused by delays on the individual links. Thus associated with each individual link in the network is a probability mass function. The delay distributions inferring problem can be roughly approximated by the linear model:

$$Y=AX+\varepsilon \tag{1}$$

where Y is a vector of end-to-end delays; A is a routing matrix; X is a vector of link delay; ε is a noise term which can result from random perturbations of X about its mean value and possibly also additive noise in the measured data Y. A is a binary matrix (the i,jth element is equal to one or zero) that captures the topology of the network. The delay distributions inferring problem refers to the problem of estimating the network parameters X given Y and either a set of assumptions on the statistical distribution of the noise ε.

We focus on discrete delay distributions, on each link, delay falls in the set $\{0,q,2q,\ldots,bq\}$, where q is the unit of measurement and b is an integer that defines the maximum delay for each link. Hence, for a path containing k links, the end-to-end delay takes values in $\{0,q,2q,\ldots,kbq\}$. We will consider inference under the stochastic assumption that the individual link delays X_k are mutually independent. Let $a_k(i)=P\{X_k=iq\}$, $i=1,\ldots,b$ and $k\in V$. In the rest of the paper, for convenience, we will drop the use of the universal measurement unit q. Further, we will denote by $\vec{a}_k=(a_k(1),a_k(2),...,a_k(b))'$, a column vector containing all of the link k delay probabilities. Let $\vec{a}=\{\vec{a}_k,k\in V\}$, a column vector containing all the parameters of interest that have to be estimated. Only the accumulated (end-to-end) delays at the receiver nodes are recorded, we observe only $\vec{Y}=\{Y_j;j\in R\}$. Note that for $j\in R$, $Y_j\in\{0,\ldots,Lb\}$ where L is the number of layers in the tree. Each multicast probe packet experiences a delay on each link along its path. Let $x=\{0,1,...,b\}^{|\varepsilon|}$ be the space of all possible link delays. Hence, $\vec{x}\in X$ is an $|\varepsilon|$-tuple describing the individual link delays that the probe experienced. Let $\vec{y}(\vec{x})$ be the multicast end-to-end measurement that results when $\vec{x}$ occurs. Note that this is a many-to-one function; there are several $\vec{x}$ outcomes that result in the same $\vec{y}$. Denote by $Y=\{\vec{y}(\vec{x});\vec{x}\in X\}$ the space of all possible multicast results.

2.2 Expectation-Maximization Algorithm

In this section, we introduce the nonparametric maximum likelihood estimation (MLE) of the delay distribution and describe the expectation-maximization (EM) algorithm for computing the MLE.

Let $N_{\vec{y}}$ be the number of probes that resulted in outcome $\vec{y}\in Y$. Let $g(\vec{y};\vec{a})=P\{\vec{Y}=\vec{y}\}$. Then the observed data correspond to a multinomial experiment in terms of the observed end-to-end link delays, and the log-likelihood can be expressed as $l(\vec{a})=\sum_{\vec{y}\in Y}N_{\vec{y}}\log[g(\vec{y};\vec{a})]$. This likelihood is a complicated function and is difficult to maximize directly.

The EM algorithm is a natural approach for computing the MLE in this kind of missing data problem. It is an iterative algorithm that starts with some initial estimate of the desired parameter values. The EM algorithm alternates between computing the conditional expectation of data log likelihood given the observations (the E-Step) and maximizing this quantity over probability (the M-Step). The process is repeated until the likelihood converges to a maximum. Each step of the algorithm is guaranteed to increase the likelihood. Let $M_{\vec{x}}$ be the number of times that a particular individual link delay set occurred. Given an estimate of $\vec{a}$, we can calculate new estimate of $\vec{a}$ and repeat the process. Formally, let the q-th step estimate of the delay distribution of all the links in the tree topology be denoted by $\vec{a}^{(q)}$. Using this estimate, we can compute $P^{(q)}\{\vec{X}=\vec{x}\}$ and $P^{(q)}\{\vec{Y}=\vec{y}(\vec{x})\}$. With these values, we can now impute the required quantities in the E-step:

$$M_{\vec{x}}^{(q+1)}=N_{\vec{y}}\frac{P^{(q)}\{\vec{X}=\vec{x}\}}{P^{(q)}\{\vec{Y}=\vec{y}(\vec{x})\}} \tag{2}$$

If we let $X_{k,i}=\{\vec{x}\in X\mid x_k=i\}$, then the M-step is :

$$a_k^{(q+1)}(i)=\frac{1}{n}\sum_{\vec{x}\in X_{k,i}}M_{\vec{x}}^{(q+1)} \tag{3}$$

3. A*-based EM Algorithm

Traditional EM algorithm uses the Breadth-First algorithm to search the entire state space X for each $\vec{y}$. With the increase of state, the speed of operation decreases sharply. Heuristic search is one of the most effective search techniques to cope with very large problem space, such as A*, IDA*, or others. In this paper, we present an improved

EM algorithm based on A* algorithm. This improved algorithm is very universally applicable which can improve greatly the convergence speed and meet the real-time requirement of network delay distributions inference.

According to the convergence property of EM algorithm in network link delay distribution inference [2], we use A* algorithm to accelerate the speed of EM algorithm. In A* algorithm, we use an evaluation function h to determine the closeness of a node to a goal state. It will move us closer to the goal faster than other paths. Two databases are used here to do the state search, one is A and the other is B. In the beginning, A is a database containing all the states and B is empty. Problem solving is usually described as a search through a state space, the operators that allow the vector $\vec{x}$ to move from state database A to state database B according to the cost h, we define the heuristic evaluation as follows:

$$h: f(n) = g(n) - h(n) \tag{4}$$

In A*, we want the highest score for the goal node, $g(n)$ is the score for appearance time of $\vec{x}$ multiply 2, and $h(n)$ is a heuristic score from a random in [0,0.01] multiply time of $\vec{x}$ that not appearance each search time.

In order to reduce the time cost of large dimension, we first compute m cluster center among dataset n, then select the nearest cluster center and find the suitable result in this cluster.

Ant colony algorithm is used to do clustering at first to sort the search space. Equation(5) is used as rule of the comparability of M_i and $\vec{x}_j$:

$$f(M_i, \vec{x}_j) = \sum_{\vec{x}_j \in area(M_i, r)} (1 - t(M_i, \vec{x}_j)) \tag{5}$$

where $f(M_i, \vec{x}_j)$ represents the similarity of center of M_i and $\vec{x}_j$, area(M_i,r) represents the cluster M_i with radius r, $t(M_i, \vec{x}_j)$ represents Euclidean Distance between center of M_i and $\vec{x}_j$ as equation (6):

$$t(M_i, \vec{x}_j) = \sqrt{q_k (\vec{x}_i - \vec{x}_j)^2} \tag{6}$$

q_k is user-defined parameter and it can be adjusted for good performance, it let $f(M_i, \vec{x}_j)$ distribute over the interval[0,1]. Where $f(M_i, \vec{x}_j)$=0, the similarity between center of M_i and $\vec{x}_j$ is minimum, where $f(M_i, \vec{x}_j)$=1, the similarity between center of M_i and $\vec{x}_j$ is maximum, $\vec{x}_j$ belongs to M_i. We use similarity to compute move probability, the pick up probability: p_p=1-$f(M_i, \vec{x}_j)$, then put down probability: p_d= $f(M_i, \vec{x}_j)$. p_0 represents probability threshold. We use parameters δ, ε and ϕ to specify the impact of trail and attractiveness, respectively for each $\vec{x} \in X$, each $M_i \in M$ and each $\vec{x} \in M_i$. Fig. 1 shows ant-based clustering algorithm and fig. 2 shows the A* algorithm. Fig. 3 shows A*-based EM algorithm.

Initialization: Create a database A, consisting all the link delay state in X. Select m clustering center at random: $M = \{M_i | M_i = (x_{i1}, x_{i2}, \ldots x_{im})\}$;
create a database B consisting the m clustering center, for all $M_i \in M$ set M_{ip}=0; select n ants at random where n<<m, set a radius of $L_{\vec{x}}$; for all $\vec{x} \in X$ set $\vec{x}_p = 0$
for (i=1 to n) do
do while not empty (B)
{compute $f(\vec{x}_i, M_j)$; p_p; p_d;
if (P_p>θ) then { $\vec{x}_{jp} = \vec{x}_{jp}$ +δ; M_{jp}=M_{jp}+ε; }
else { $\vec{x}_{jp} = \vec{x}_{jp}$ -δ/10; M_{jp}=M_{jp}-ε/5;}
if (P_d>θ and $\vec{x}_{jp}$>δ) then
{ for (each $\sqrt{(\vec{x}_k - \vec{x}_i)^2} < L_{\vec{x}}$) do {$M_j$= $\vec{x}_k \cup M_j$;M_{xk}=i}
update clustering center of M_j;
recorder B using M_{jp} with highest evaluation on the top;
goto **skip**;
}
}
skip:
endfor

Fig. 1 Ant-based clustering algorithm.

Initialization: Use database A, B. Use an additional item h to denote the value of heuristic function; Create a database C that is initially empty, the similar structure to database A.

FunctionA* ($\vec{y}_k$) {
if (B is empty) then goto **middle;**
else {
select the set of $\vec{X}$ satisfying $\min |\vec{y}_k - \vec{y}_l(\vec{x})|$ from B;
select the set of $\vec{X}$ satisfying $\vec{y}_k = \vec{y}_l(\vec{x})$ from A;
if (found) then
{ recompute the value of heuristic function h and select the items that their value of h less than the threshold θ;
recorder C using the evaluation function $f(n)$ with highest evaluation on the top;
goto **end**;
}
else goto **middle**;
}
middle: select the set of $\vec{X}$ satisfying $\vec{Y} = \vec{y}(\vec{x})$ from A;
recomputed the value of heuristic function h ;
move the set of $\vec{X}$ to C;
recorder C using the evaluation function $f(n)$ with highest evaluation on the top;
recorder A using the evaluation function $f(n)$ with highest evaluation on the top;
end: return($\vec{X}$);
}

Fig. 2 A* algorithm.

Initialization: Set the initial link delay distribution:
$\alpha_k^{(q+1)}(i)$,$k\in V$,q=0,$i\in\{1,\ldots,b\}$;
Input topology T and end-to-end measurements $\{\vec{y}_k\}_{k\in N}$.

for (each $\vec{y}_k$) {
call **FunctionA***($\vec{y}_k$);
expectation : $M_{\vec{x}}^{(q+1)} = N_{\vec{y}} \dfrac{P^{(q)}\{\vec{X}=\vec{x}\}}{P^{(q)}\{\vec{Y}=\vec{y}(\vec{x})\}}$;
maximization: $\alpha_k^{(q+1)}(i) = \dfrac{1}{n}\sum_{\vec{x}\in X_{k,i}} M_{\vec{x}}^{(q+1)}$;
}

Fig. 3 A*-based EM algorithm.

4. Experience

We use ns2 to perform the network simulation and test the improved EM algorithm for network link delay distributions inference. We use a three-layer tree as the network topology shown in fig. 4. Node 0 is root node, and 4, 5, 6, 7, 8 are leaf nodes. We use end node's number of a link as the number of the link, such as link number between node 0 and node 1 is 1. Link 1 has bandwidth 5Mb/sec with latency 4ms. Link 2 has bandwidth 4 Mb/sec with latency 5ms. Link 3 has bandwidth 3 Mb/sec with latency 6ms. Link 4, link 6 and link 7 all have bandwidth 2 Mb/sec with latency 8ms. Link 5 and link 8 have bandwidth 3 Mb/sec with latency 7ms. Each link was modeled as a Drop-Tail queue.

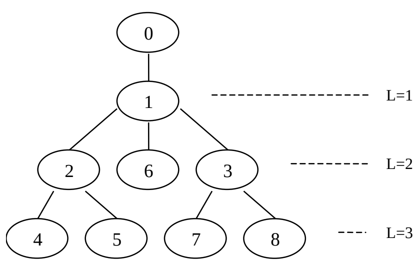

Fig. 4 Network topology.

We use multicast probing schemes, sending probes across the network according to a Poisson process with mean interarrival time being 0.02s and keeping track of the length of time it takes packets to travel from the root node to the leaf nodes. In order to close to the real IP network environment, we use 200 TCP flows to comprise the exponential on-off background traffic. Experiment put on under subscribed case q=1ms, b=10 and we set window size to be 4s. A set of initial link delay distribution values is given as follows: $\alpha_k(1)$=0.025 , $\alpha_k(2)$=0.05 , $\alpha_k(3)$=0.075 , $\alpha_k(4)$=0.15 , $\alpha_k(5)$=0.3 , $\alpha_k(6)$=0.15 , $\alpha_k(7)$=0.1 , $\alpha_k(8)$=0.75 , $\alpha_k(9)$=0.05 , $\alpha_k(10)$=0.025 , $k\in\{1,2,\ldots,8\}$. We assume all links have delay during the experiment, so $\alpha_k(0)$=0.

Fig. 5 shows convergence of the estimates of $\vec{\alpha}_8$ in the first window under subscribed case b=10, with the increasing number of probing packet, the $\vec{\alpha}_8$ convergence to the stable true value. With the same background, we set b to 5,6,7,8,9,10,11,12,13,14,15 and compare the computing time between traditional EM algorithm and the improved EM algorithm. Fig. 6 shows the result. We can see on the increase value of b, the A*-based EM algorithm uses less time than the traditional EM algorithm. The value of b is getting larger, the performance of the A*-based EM algorithm is getting better than the traditional EM algorithm and the time is saved more. The comparison made between A*-based EM algorithm and traditional EM algorithm shows the A*-based EM algorithm operates quickly and meets the real-time requirement of network tomography, therefore it is more suitable for the research on a complicated large-scale networks tomography, and it is universally applicable and effective.

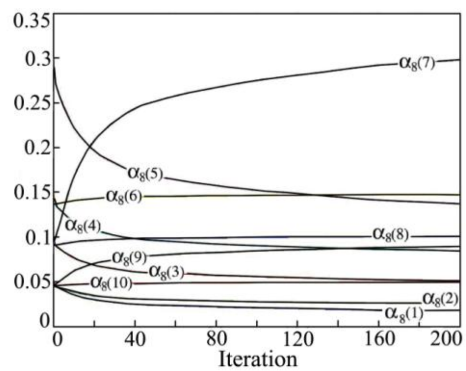

Fig. 5 Convergence of the estimates of $\vec{\alpha}_8$.

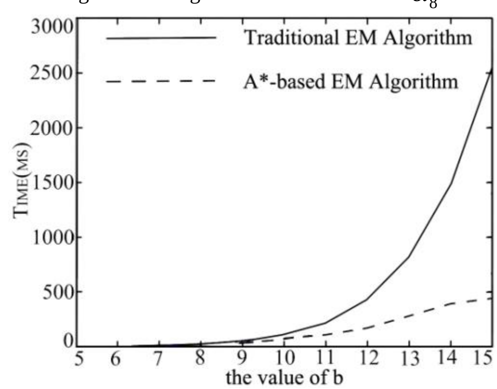

Fig. 6 Comparison between traditional EM algorithm and A*-based EM algorithm.

5. Conclusion

In this paper, we propose an A^*-based EM algorithm for network link delay distributions inference. We use A* algorithm to accelerate the convergence speed of EM algorithm. Experiment results show the A^*-based EM algorithm operates quickly and meets the real-time requirement of network tomography. It is also effective and suitable for solving such problem in the field of network tomography.

Acknowledgments

This work is supported by Foundation of Northeast Agriculture University.

References

[1] Y.Vardi, Network Tomography: Estimating Source-Destination Traffic Intensities From Link Data, Journal of the American Statistical Association, vol.91, no.433, pp.365-377, 1996.
[2] M.Coates, A.Hero, R.Nowak, and B.Yu, Internet Tomography, IEEE Signal Processing Magazine, vol.19, no.3, pp.47-65, 2002.
[3] ns-Network Simulator. For more information http://www-mashcs. berkeley. edu/ns/ns.html.
[4] F.L.Presti, N.Duffield, J.Horowitz, and D.Towsley, Multicast-based inference of network-internal delay distributions, AT&T Laboratories and University of Massachustts, Tech. Rep., 1999.
[5] Korf, R. E., Depth-first iterative-deepening: An optimal admissable tree search, Artificial Intelligence, Vol. 27, No. 1, pp. 97-109, 1985.
[6] Korf, R. E., Recent progress in the design and analysis of admissible heuristic functions, Proceedings of AAAI-2000, pp. 1165-1170, 2000.
[7] Dorigo M, Maniezzo V,Colorni A. Ant system optimization by a colony of cooperating agents. IEEE Transaction on Systems, Man, and Cyberneties.1996,26(1):29-41.

Hongjie Sun received the Ph.D degree in Computer Science and Technology from Harbin Institute of Technology in 2007. Her research interests include the theory and technique of computer network and information security, parallel computing.