

Efficient Design of End-to-End Probes for Source-Routed Networks

Srinivasan Parthasarathy[†], Rajeev Rastogi[‡], and Marina Thottan[‡]

[†]Department of Computer Science, University of Maryland, College Park, MD 20742

[‡]Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07920

Abstract

Migration to a converged network has caused service providers to deploy real time applications such as voice over an IP (VoIP) network. From the provider's perspective, the success of such emerging multimedia services over IP networks depend on mechanisms which help understand the network-wide end-to-end performance dynamics. In this work, we present a mechanism to design efficient probes for measuring end-to-end performance impairments such as network delay and loss for a specific service in the provider network. We address two main issues related to deploying network probes: (1) the need for correlating the topology data with the measured values and (2) reducing the amount of probe traffic. We use explicitly routed probe packets to alleviate the need for correlation with topology measurements. We also present a 3.5-approximation algorithm for designing probe-sets which cover all the edges in the network. Further, we explore techniques for using observed performance degradations in a given set of probes to isolate the miscreant-edge which caused the degradations. We state a precise characterization for probe-sets which isolate miscreant edges in the network; this also suggests a natural heuristic for miscreant-edge detection. Simulations on ISP topologies obtained from the RocketFuel project show that our algorithms perform much better than the analytically guaranteed bounds and are near-optimal in practice with respect to probe costs.

1 Introduction

Migration to a converged network implies that service providers will have to support emerging real time applications such as voice over an IP (VoIP) network. The widespread deployment of VoIP places stringent performance requirements on the IP network. Today's service provider networks have sufficient capacity to account for most throughput related performance issues. However, it has been shown that an appropriate metric for emerging real time applications is network wide service availability, which includes performance impairments due to routing protocol reconvergence times and service disruption

duration [7]. Routing convergence delay is the duration between a link failure/recovery and the instant when new routing tables are available. Service disruption time is the duration between the time at which packet forwarding stops and the time when it resumes. Thus the restoration of a service depends not only on available bandwidth but also on router architectures and the design of the control plane. Thus from a provider's perspective it is difficult to guarantee the necessary service availability by merely adding more capacity [7].

A good understanding of the end-to-end network-wide performance dynamics pertaining to a specific service is imperative for the service provider. Part of the solution to providing service availability metrics for emerging applications is the accurate and fast measurement of end-to-end network performance. *In this work we address the problem of designing efficient probes for measuring end-to-end performance impairments such as network delay and loss in a service provider network.*

Network performance can be measured using the standard SNMP based polling [9] or link level measurements. SNMP only provides a device centric view of performance while link-state metrics such as those based on *Hello* packets can be used to detect link failures/availability. Neither of these methods can be used directly to measure performance metrics such as end-to-end delay and loss. Providing service quality guarantees requires end-to-end measurements that are obtained through probing techniques. There are different types of probe mechanisms [1][2]: *one-packet* methods such as *pathchar* [4], the *packet pair* family [6] and IPMP [8]. *Pathchar* is used for estimating link bandwidths from round trip delays of packet sequences from successive routers [3]. Packet-pair methods can be used for estimating available bandwidth or the bottleneck link rate [5]. Using these techniques to obtain end-to-end measurements requires that the measured data be correlated with the topology information obtained from *traceroute*.

Further, the treatment received¹ from these probe packets is dependent on the protocol used to encapsulate the probe packet. The IP Measurement Protocol (IPMP) is used for measuring one-way delay [8] and is designed to overcome some of the limitations of the packet probes. The IPMP protocol combines both path and delay measurements thus alleviating the need for correlation with the *traceroute* measurement. Path information is determined from the path record field in the IPMP packet, which is populated by the routers. All of these probing techniques when applied to obtain end-to-end measurements will place a significant load in the network. Furthermore these path oriented measurement tools are capable of measuring latencies and bandwidth information for only a limited set of links in the node's routing tree.

In this paper we simultaneously address the two main issues related to deploying network probes: (1) the need for correlating the topology data with the measured values and (2) reduce the amount of probe traffic. We address the issue of correlation with topology by designing probes that use precomputed MPLS paths. Using complete knowledge of network topology the service provider can efficiently choose the MPLS probe paths for maximum coverage, thus reducing the total probe traffic. Once such a probe-set has been designed, one can either use one of the probing techniques mentioned above or design application specific probes that mimic specific services such as VoIP probes [?]. Using MPLS to design probe paths provides a deterministic map between the probe measurements and the associated links. Thus we can avoid the correlation problem with inaccurate *traceroute* data (due to transient congestions) as well as the limitation of path-record fields in IPMP.

Our end-to-end monitoring system would consist of efficient design of probe paths that provide maximum coverage of network links. Our goal is to obtain few probe paths of low cost and still provide good coverage for all links in the network. Probes are sent along the pre-computed paths at some predefined frequency and checked for violations in terms of end-to-delay or loss metrics. Based on the set of probe paths that experienced a performance degradation we identify the set of links that are common to these paths. We then initiate a miscreant-link detection algorithm on the common set of links to isolate the network link that is contributing to the performance degradation. We believe that this approach will significantly reduce the total probing load on the network as well as the amount of processing that must be done to correlate measured data with topology information.

The design of our minimal probe-paths is obtained us-

¹what does this mean?

ing the probe-cover algorithm. This algorithm is based on the solution to the Chinese Postman Problem and yields a 3.5 asymptotic approximation ratio for the covering problem. Further, we develop a necessary and sufficient condition which yields a precise characterization for probe-sets which isolate miscreant edges in the network; this suggests a natural heuristic for miscreant-edge detection. Simulations on ISP topologies obtained from the RocketFuel project [18] show that our algorithms perform much better than the analytically guaranteed bounds and are near-optimal in practice with respect to probe costs.

The rest of paper is organized as follows: the next subsection surveys related work. Section 2 presents our network and probe models, notation, and formal problem statements. Section 3 discusses our approximation algorithm for the probe cover problem and Section 4 discusses the algorithm to identify the miscreant-link. Section 5 presents the results of simulations of our algorithms on ISP topologies along with the discussion of the results and insights. We propose problems for future investigation as well as implementation challenges in Section 6.

1.1 Related Work

Network probing with low overhead has prompted a flurry of research activity in recent past. The IDmaps project [13] produces latency maps of the Internet from which latencies of any arbitrary path can be obtained. However since only a relatively few paths are actually monitored it is possible to make errors in estimating the latencies of any arbitrary path. Adler *et al.* [10] provide a solution to compute the minimum cost set of multicast trees that can cover links of particular interest in the network. Recently, the authors [?] have provided algorithms for selecting probe stations such that all links are covered and compute the minimal set of probe paths that must be transmitted by each station such that the latency of every link can be measured. In [17], the authors consider the problem of probe path design where they assume local flexibility - the probe paths can be selected as either the current IP route or one of the current IP routes of the immediate neighbors. The efficient probe node (beacon) placement strategy proposed in [16] provides the minimum number of probe nodes required to deterministically monitor all network links even in the presence of dynamism in IP routes. All of the above work has so far focussed on IP routes as potential probe paths. In the work presented here the focus is on explicitly routed probe packets. A closely related work was presented by the authors in prior work [19]. Here we studied the problem of measuring path latencies through explicitly routed packet probes while minimizing the overhead imposed

by the probe traffic. The probe packets originate from a central point in the network. Our probe-cover algorithm focuses on the design of probe paths and differs from [19], since we assume full knowledge of network topology and can choose probe paths that can originate and terminate from any given set of terminal nodes in the network.

The second part of this paper addresses the problem of miscreant link detection. The authors of [11] compute link level loss rates and delays from a given collection of multicast trees and corresponding end-to-end measurements. In [15], the authors isolate a link failure in the context of a large multicast distribution of trees. In [14] an algebraic approach has been proposed where, given a set of end-to-end delay measurements, it is possible to compute delays of smaller path segments. Our approach yields good heuristics for efficient design of probe paths and can be generalized to detect a range of performance degradations due to underlying link failures, including increased delay, loss, congestion and total failure of a link.

2 Preliminaries

We model the network as a connected weighted undirected graph $G = (V, E)$, with cost function $w : E \rightarrow R^+$ which denotes the cost of each edge (link). For an edge $e \in E$ and a probe path p , we let $e \in p$ denote the fact that path p contains edge e . The weight of the path p , $w(p) = \sum_{e \in p} w(e)^2$. The *terminal set* $\mathcal{T} \subseteq V$ is the set of nodes which can act as end points of a probe path. Note that the terminal set is responsible for initiating the probes and collecting end-to-end probe data.

In the probe-cover problem, we are given a graph $G = (V, E)$, a positive cost function on the edge set w , a set of terminals $\mathcal{T} \subseteq V$ and the number of probes k . A feasible solution to the probe-cover problem is a set of probe-paths \mathcal{P} such that **(1)** $\forall e \in E, \exists p \in \mathcal{P}$ such that $e \in p$; and **(2)** The endpoints of all probes $p \in \mathcal{P}$ belong to the set \mathcal{T} . The objective of the probe-cover problem is to minimize the maximum length of any probe $p \in \mathcal{P}$: i.e., $\min_{p \in \mathcal{P}} w(p)$.

In the miscreant-edge detection problem, we are given a graph $G = (V, E)$, the cost function w for E , the terminals \mathcal{T} , an bound on the number of probes k and an upper bound on the maximum length ℓ . A feasible solution to miscreant-edge-detection problem is a set of probe-paths \mathcal{P} which respect the upper bounds and which terminate in the vertex set \mathcal{T} such that the following condition hold: $\forall e \in E, |((\bigcap_{e \in p} p) \setminus (\bigcup_{e \notin q} q))| = 1$. As shown in Section 4, this condition is both necessary and sufficient for

²if edge e occurs $k > 1$ times in p , then the weight of edge is counted k times in $w(p)$

isolating the miscreant edge by using only the observations gathered from the end-to-end probes \mathcal{P} .

We remark that the cost function on the edges could be arbitrary positive scalars. However, two cost functions which are particular relevant is the unit function which and link delays. Setting the cost of each edge as one would minimize the maximum number of edges with a probe and setting the cost as the delay would minimize the delay on each probe. A lower cost probe general yields a higher level of confidence about the link-level conclusions and also localizes the miscreant edge to a small subset of edges. The number of the probes directly impacts the amount of probe traffic in the network. The set of terminals are which need to be instrumented for supporting probe initiation and collection of end-to-end probe data; low number of terminals thus imply a low set up cost for end-to-end probing. Our goal is to explore and evaluate rigorous techniques for probe-design which optimally trade-off these parameters for covering and miscreant-edge detection in the network.

3 Probe-Cover Algorithm

We now discuss our algorithm for the probe-cover problem. The following theorem states that the probe-cover problem is *NP*-complete.

Theorem 1 *The decision version of the probe-cover problem is NP-Complete.*

Hence, algorithms for obtaining optimal solutions for probe-cover are likely to be impractical; we now present a provably good approximation algorithm whose solution is guaranteed to be at most 3.5 times the optimal value. Our algorithm does not require any knowledge about the structure of the optimal set of probes, but relies only on an easily computable lower bound on the cost of any feasible solution. Our algorithm is composed of the following three steps:

Step 1: We first compute a Chinese Postman Tour \mathcal{C} of the network G . Recall that the CPT is the tour of least cost in the graph which contains all edges. It is well known that if each edge occurs exactly once in this tour, then every node has an even degree and the graph is *Eulerian*. We note that every connected graph has a CPT \mathcal{C} which can be computed in polynomial time such that the total cost of the tour is at most 1.5 times the cost of the $w(\mathcal{C}) \leq 1.5 \cdot \sum_{e \in E} w(e)$ [12].

Step 2: Let w_{\max} denote the maximum cost of any edge. We partition \mathcal{C} into a set R of k contiguous segments such that each segment in R has a length of at most $\frac{w(\mathcal{C})}{k} + w_{\max}$. A moment's reflection shows that such an equitable partitioning of the CPT \mathcal{C} exists and can be easily computed.

Step 3: We convert each segment $r \in R$ into a probe as follows. Let u_1 and v_1 be the end-points of r . Let u and v be the terminals which have the least distance to u_1 and v_1 respectively, and let $u \rightsquigarrow u_1$ and $v_1 \rightsquigarrow v$ denote the corresponding shortest paths. We construct a probe corresponding to path r by concatenating the paths $u \rightsquigarrow u_1$, r and $v_1 \rightsquigarrow v$ in that order. The final set of probes \mathcal{P} will contain k probes, each of which correspond to a segment $r \in R$.

The following theorem states that the probe-cover algorithm has a worst-case asymptotic approximation ratio of at most 3.5.

Theorem 2 *Let \mathcal{P} be the set of probes output by the probe-cover algorithm and let opt denote the cost of the optimal solution. Then, $\max_{p \in \mathcal{P}} w(p) \leq 3.5 \cdot opt + w_{\max}$.*

Proof Since every edge needs to be covered by the optimal solution, $opt \geq \frac{\sum_{e \in E} w(e)}{k}$. Consider any segment r in R with end points u_1 and v_1 . The length of the segment r is at most $\frac{1.5 \cdot \sum_{e \in E} w(e)}{k} + w_{\max}$. Let u and v be the closest terminals to u_1 and v_1 . Both the end points u_1 and v_1 occur in some probe in the optimal solution with a maximum probe length of opt . Hence, the shortest paths $u \rightsquigarrow u_1$ and $v_1 \rightsquigarrow v$ are of length at most opt . Hence, the final probe which includes r is of length at most $\sum_{e \in r} w(e) + 2 \cdot opt$, which concludes the proof of the theorem. ■

4 Miscreant-Edge Detection

We now present a necessary and sufficient condition as well as a natural heuristic motivated by this condition for miscreant-edge detection. Recall the setting for miscreant-edge detection: we monitor the end-to-end performance along a set of probe paths \mathcal{P} by periodically sending probe packets along these paths. When a miscreant edge $e \in E$ ‘fails’, it results in the failure of all paths in \mathcal{P} which contain this edge, while the other paths remain unaffected. The objective is to compute a set of probe paths \mathcal{P} subject to the constraints on the number of probes, the maximal path length, and the given terminal set \mathcal{T} . For any edge $e \in E$, let \mathcal{P}_e denote the set of probes which contain edge e . We now derive the following necessary and sufficient condition for detecting the failure of an edge e .

4.1 A Necessary and Sufficient condition

Theorem 3 *The set of probes \mathcal{P} can detect the failure of edge e if and only if $(\bigcap_{p \in \mathcal{P}_e} p) \setminus (\bigcup_{q \notin \mathcal{P}_e} q) = \{e\}$. As*

a corollary, if the set \mathcal{P} be can detect the failure of each edge in the set A , then $|\mathcal{P}| \geq \log_2 |A|$.

Proof Assume that the condition holds. Whenever an edge fails (which is detected by some probe failure), we check the following two conditions: 1) all probes in \mathcal{P}_e have failed 2) no other probe has failed. We note that if e has failed then both these conditions must hold and vice-versa. Hence, this check will determine if edge e has failed or not.

We now show that the condition need to hold for the edge failure to be detected. Assume the contrary, i.e., $(\bigcap_{p \in \mathcal{P}_e} p) \setminus (\bigcup_{q \notin \mathcal{P}_e} q) \supset \{e\}$. In particular, assume that this set also contains another edge e' which is different from e . We note that the failure of e or e' both result in the same set of probe failures, thus making these two cases indistinguishable from each other. Hence the necessary and sufficient condition holds.

Further, these conditions imply that for any two edges e and f in A , the set \mathcal{P}_e and \mathcal{P}_f are distinct. Hence, there are at least $|A|$ distinct subsets of the set \mathcal{P} . This concludes the proof of the theorem. ■

4.2 Heuristic for Miscreant-Edge Detection

We now describe our heuristic for miscreant-edge detection. Our heuristic is naturally motivated by the necessary and sufficient conditions of Theorem 3. Our heuristic proceeds as follows: we create an initial set of probes using the algorithm for the simple-probe cover problem presented earlier. Let \mathcal{Q} denote this initial set of probes and let \mathcal{Q}_e denote the set of probes in \mathcal{Q} which contain e . For each edge e , we compute the set $S(e)$ which is defined as follows:

$$S(e) = \left(\bigcap_{p \in \mathcal{Q}_e} p \right) \setminus \left(\bigcup_{q \notin \mathcal{Q}_e} q \right)$$

Let B be the set of edges e such that $|S(e)| > 1$. By Theorem 3, the edges in B are the set of edges whose failure can *not* be isolated using the set of probes \mathcal{Q} . We now ‘fix’ each of these bad edges as follows. Process edges in B in arbitrary order. Let the current edge being processed be e . Remove the edge e from G and obtain a ‘long’ path γ whose end points are in \mathcal{T} and which covers all edges in the set $S(e) \setminus \{e\}$. Such a path can be obtained by simply linking the end-points of edges in $S(e)$ using shortest paths. We then splice the path γ into segments R such that each segment $r \in R$ is of length at most $\ell + w_{\max}$. As in the probe cover algorithm of Section 3, we now obtain a probe $p \in \mathcal{P}$ corresponding to each segment $r \in R$ by linking the endpoints of r to their closest terminals using shortest paths. Due to lack

of space, we note without proof that for any biconnected network and for any given set of terminals, this heuristic is guaranteed to produce a feasible solution for the miscreant-edge detection problem.

5 Simulations

We now present the results of our simulations. We simulated our probe-cover algorithm and miscreant-edge detection heuristic on four ISP topologies obtained from the RocketFuel project [18]. Each link in the network was assigned a cost which is its inferred latency. The number of nodes, links, and the total cost of all the links for the four topologies is tabulated in Table 1.

Topology information			
ISP	Num. nodes	Num. links	Total edge cost
1	315	972	3119
2	87	161	469
3	161	328	1912
4	79	147	738

Table 1: Topology information for the 4 RocketFuel ISPs in our experimental setup

For our probe-cover experiments, we vary the number of probes (k), the number of terminal nodes, and investigate the maximum probe-cost. The set of terminals is chosen uniformly at random from the network nodes. All the data points were averaged over ten runs of the simulation.

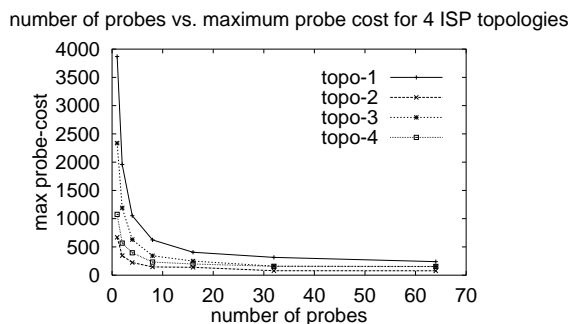


Figure 1: maximum probe-cost is inversely proportional to the number of probes

Figure 1 presents the impact of the number of probes on the maximum probe-cost. Clearly, the max. probe-cost is inversely proportional to the number of probes (the best functional dependency that can be achieved). The ratio of the total edge-weights and the number of probes represents an absolute lower bound on the optimal value of the max probe-cost. Hence, our algorithms

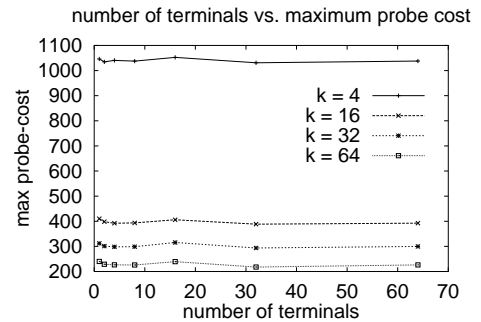


Figure 2: probe-cost is independent of the number of terminals

can be seen to achieve near-optimal values in practice instead of the 3.5-performance ratio which is analytically guaranteed in the worst case. Figure 2 presents the effect of varying the number of terminals on the max. probe cost. Interestingly, the number of terminals do not play a significant role in determining the value of the maximum probe-cost. This is intuitive since the probe-cost is dominated by the total cost of the chinese-postman tour, which is independent of the number of terminals chosen.

Table 2 presents the results for our miscreant-link detection heuristic. The first and the second column in the table represent the maximum probe-costs and the total number of probes respectively for ISP-1 and 64 randomly chosen terminals. Observe the non-monotonic relationship between these two quantities. We believe that the following factor is the cause for this non-monotonicity. Recall from Section 4 that the goal of the miscreant-link heuristic is to obtain probe sets \mathcal{P} such that for each edge e , the set $S(e)$ is a singleton set which contains only edge e . Long probes essentially affect the cardinality of $S(e)$ for only a small number of edges e . Hence, this results in too many probes being added. However, if probes are too short, then a large number of them is needed to even cover all the edges. This non-linear relationship between the two metrics is qualitatively different from that of the probe-cover case and we believe, merits further investigation.

6 Future Work

In this work we presented rigorous and provably good approaches for the design of end-to-end probes which cover edges and detect miscreant-edges in a network. Our simulations indicate that the algorithms presented here work much better in practice than the analytically guaranteed worst-case bounds. We propose several interesting research issues which merit future investigation.

The first issue pertains to implementation of our algorithms in existing source-routed MPLS networks. As

Miscreant-edge Detection	
Maximum probe cost	Number of probes
504	97
552	90
728	86
832	86
1172	94
2300	95
3800	104

Table 2: Variation of the the maximum probe cost vs. the number of probes for miscreant-edge detection for Topology-1 with 64 terminals: observe the non-monotonic relationship

presented here, our probe-paths are not required to be loop-free; this is at variance with current MPLS routing and network management protocols. Thus, efficient implementation of our algorithms would require extending our approaches to eliminate loops from all probe-paths or extending MPLS protocols to allow probe-paths which contain cycles. The second issue we propose for future investigation is that of minimizing the set-up cost of placing the terminals. Placing terminal nodes at the boundary of the network is more desirable than placing them at the core; we propose to investigate algorithms which trade-off the terminal set up cost w.r.t. to the max. probe-cost. Another interesting question for future investigation is to extend our approach to detect miscreant-edges *as well as* miscreant-nodes within the network.

7 Acknowledgements

We thank Neil Spring for helpful general discussions and suggestions about RocketFuel topologies.

References

- [1] K. Lai and M. Baker, "Measuring Link Bandwidths Using a deterministic Model of Packet Delay", in Proc. ACM SIGCOMM, Stockholm, 2000
- [2] C. Dovrolis, P. Ramanathan and D. Moore, "What do Packet Dispersion techniques measure?", in Proc. IEEE INFOCOM, Anchorage, Alaska, USA, April 2001
- [3] A. B. Downey, "Using Pathchar to estimate Internet link characteristics", in Proc. ACM SIGCOMM, Cambridge, MA 1999.
- [4] V. Jacobson, "Congestion avoidance and control", in Proc. ACM SIGCOMM, Stanford, CA, USA, 1988.
- [5] R. L. carter and M. E. Crovella, "Measuring bottleneck link speed in packet switched networks", Performance Evaluation, Vol. 27 & 28, pp 297-318, 1996.
- [6] A. Pasztor and D. Veitch, "Active probing using Packet quartets", in Proc. IMC.
- [7] G. Iannaccone, C. Chuah, R. Mortier, S. Bhat-tacharyya and C. Diot, "Analysis of link failures in an IP back bone".
- [8] M. J. Luckie, A. J. McGregor and H. W. Braun, "Towards Improving packet probing techniques".
- [9] W. Stallings, "SNMP, SNMPv2, SNMPv3 and RMON 1 and 2", Addison-Wesley Longman Inc. 1999, (Third Edition).
- [10] M. Adler, T. Bu, R. Sitaraman and D. Towsley, "Tree layout for internal network characterizations in multicast networks", In Proc. Of NGC, London, UK, Nov 2001.
- [11] T. Bu, N. Duffield, F. Lo Presti, and D. Towsley, "Network tomography on general topologies", in Proc. ACM SIGMETRICS, 2002.
- [12] G. N. Frederickson, M. S. Hecht, and C. E. Kim. "Approximation Algorithms for some routing problems", in SIAM Journal on Computing, 7(2):178-193, May 1978.
- [13] F. Francis, S. Jamin, V. Paxson, L. Zhang, D. F. Gryniewicz and Y. Jin, "in Proc. Of IEEE INFOCOM 1999, New York City, NY.
- [14] Y. Shavitt, X. Sun, A. Wool and B. Yener, "Computing the Unmeasured: An algebraic approach to Internet Mapping", in Proc. IEEE INFOCOM 2000, Tel Aviv, Israel, Mar 2000.
- [15] A. Reddy, R. Govindan and D. Estrin, "Fault isolation in multicast trees", in Proc. ACM SIGCOMM, 2000.
- [16] R. Kumar and J. Kaur, "Efficient Beacon Placement for Network tomography", In Proc of Internet Measurement Conference 2004.
- [17] J. D. Horton and A. Lopez-Ortiz, "On the number of distributed measurements points for network tomography", In Proc. Of Internet Measurement conference 2003.
- [18] N. Spring, R. Mahajan and D. Wetherall, "Measuring ISP topologies with rocket fuel", in Proc of ACM SIGCOMM 2002.
- [19] Y. Breitbart, C. Y. Chong, M. Garofalakis, R. Rastogi and A. Silberschatz, "Efficiently Monitoring bandwidth and latency in IP networks", in Proc of IEEE INFOCOM Tel Aviv, Israel, Mar 2000.