Proceedings of the 2004 Winter Simulation Conference
R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds.

ON-DEMAND COMPUTATION OF POLICY BASED ROUTES FOR LARGE-SCALE NETWORK SIMULATION

Michael Liljenstam
David M. Nicol

Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1308 W. Main St., Urbana, IL 61801, U.S.A.

ABSTRACT

Routing table storage demands pose a significant obstacle for large-scale network simulation. On-demand computation of routes can alleviate those problems for models that do not require representation of routing dynamics. However, policy based routes, as used at the interdomain level of the Internet through the BGP protocol, are significantly more difficult to compute on-demand than shortest path intradomain routes due to the semantics of policy based routing and the possibility of routing divergence. We exploit recent theoretical results on BGP routing convergence and measurement results on typical use of BGP routing policies to formulate a model of typical use and an algorithm for on-demand computation of routes that is guaranteed to terminate and produces the same routes as BGP. We show empirically that this scheme can reduce memory usage by orders of magnitude and simultaneously reduce the route computation time compared to a detailed model of the BGP protocol.

1 INTRODUCTION

As a result of efforts over the last several years, there now exists a number of network simulators that can simulate networks consisting of hundreds of thousands or even millions of devices. However, important hurdles still remain for these efforts whose overall goal is to scale as far as possible towards the size of the Internet. One such hurdle is to represent, in a single simulation, the very large amount of routing information that is kept in the Internet. Towards the core of the Internet, routers representing their administrative domain maintain routing information for reaching multiple subnetworks in all other administrative domains in the whole Internet. Thus, with the Internet currently consisting of more than 16,000 Autonomous Systems (administrative domains) and more than 130,000 subnetworks (network prefixes), the sheer volume of routing information poses a significant challenge.

1.1 Problem Statement and Related Work

Scalable simulation of routing has been recognized as a key problem for large-scale network simulation in studies such as (Hao and Koppol 2003, Huang and Heidemann 2001, Riley et al. 2000). Riley et al. (2000) proposed computing shortest path routes on-demand and using source-routing based on NIC Index vectors (NIx-vectors) to avoid building full routing tables in router nodes. That is, the source-routes contain a vector of indexes of the next outgoing Network Interface Card for each hop router which also facilitates fast packet forwarding. They discuss the implications for simulation accuracy and show, through empirical results, that the technique can give substantial savings in memory and also execution time. Generally speaking, on-demand computation of routes is permissible in simulations where routes can be considered to be mostly stable, so that effects of routing dynamics will not be a significant factor.

Huang and Heidemann (2001) proposed more aggressive abstractions to reduce the computational complexity of computing shortest path routes through an approximation technique. However, to the best of our knowledge, there have been no studies that consider the computation of policy-based routes as used at the interdomain level in the Internet (through the Border Gateway Protocol) to forward traffic in network simulations. Hao and Koppol (2003) is related to this problem in that they recognize the problem of simulating the operation of the Border Gateway Protocol (BGP) on an Internet scale. However, they are primarily interested in routing dynamics, not supporting traffic forwarding, and so describe some implementation techniques that allow them to simulate route convergence for a single destination prefix on a full AS level graph of the Internet (abstracting each AS to a single BGP router). Here, our goals are to: *i)* support accurate traffic forwarding in large-scale network simulations, *ii)* reduce memory demand as much as possible, and *iii)* also reduce computational complexity where possible. Since our focus is not on the routing dynamics, but in supporting traffic forwarding in large network models, we explore on-demand computation of routes, as in (Riley et al. 2000). However, policy based routing has some fundamental differences compared to shortest path routing, as will be described later, requiring very different techniques as we approach this problem.

1.2 Main Contributions

To reduce memory requirements we compute routes on-demand, as in (Riley et al. 2000), but design a novel algorithm that will compute the same interdomain routes that BGP would have, without fully emulating the behavior of BGP, as in (Hao and Koppol 2003). To ensure termination of our algorithm and reduce computational complexity, we rely on a theoretical result on BGP convergence (Gao and Rexford 2001) and model the usage of BGP according to the BGP configuration guidelines they suggest to ensure BGP routing convergence. The actual usage of BGP in the Internet has later been found to agree well with those guidelines (Wang and Gao 2003), and thus with our usage model. However, it should be noted that we have made one extension to the guidelines in order to turn them into a complete model, dealing with sibling peering relationships, and this extension has not been thus validated. Nevertheless, sibling relationships are uncommon compared to other peering relathionships, so it should not significantly affect the overall accuracy of the model. (Relying on models of peering relationships also has the advantage of making it possible to model effects of policy routing in the Internet without complete knowledge of the exact routing policy configurations of all Internet routers, as will be discussed later.)

We show, through experiments, that order of magnitude reductions in memory usage are possible on realistic topologies, depending on traffic density, for simulations where routing dynamics do not play a significant role. Moreover, we show that the algorithm can also reduce route computation time compared to full simulation of BGP routing convergence.

1.3 Roadmap

The remainder of this paper is organized as follows. We start in Section 2 by discussing the Border Gateway Protocol (BGP). We briefly describe its basic operation, the known risk of non-convergence, and restate Gao and Rexford's proposed constraints (Gao and Rexford 2001) on preference policies to ensure convergence. Non-convergence of BGP must be avoided or detected since it would result in non-termination of our algorithm for on-demand computation of routes. Since it is also known that determining whether a given policy configuration will converge is computationally intractable, we instead resolve this problem by restricting the policies according to the guidelines proposed by Gao and Rexford. In Section 3 we construct an algorithm to compute BGP routes on-demand. We first extend Gao and Rexford's result to deal with a case, of practical significance, that they did not include in their study, and then construct an algorithm based on their constructive proof of routing convergence. Section 4 compares the performance of the on-demand algorithm, in terms of memory and time usage, with a detailed model of the BGP protocol. Finally, Section 5 concludes and discusses future work.

2 BGP-4: THE BORDER GATEWAY PROTOCOL

The Border Gateway Protocol (BGP) (Rekhter and Li 1995, Stewart 1998) is the de-facto standard for interdomain routing in the Internet today. BGP is a fairly complex protocol, so space constraints prevent us from more than scratching the surface here. The Internet is divided into currently more than 16000 Autonomous Systems (ASes), where an AS is a network under a single administrative control that has been assigned an AS number. To achieve global connectivity, ASes must cooperate in the sense that some ASes transit traffic between other ASes. Traffic exchange and transiting is governed by business agreements between the entities owning the ASes. Internet routing is done hierarchically at two levels: intradomain routing within ASes, and interdomain routing between ASes. In contrast to intradomain routing based on shortest paths, routing at the interdomain level is done based on policies; and policies are designed based on business relationships between organizations.

2.1 Peering Relationships

Two ASes that agree to exchange traffic do so by setting up one or more BGP peering connections with policies that govern how routing information, and thus traffic, is exchanged. Hence, a network operator expresses his/her preference for forwarding traffic to and accepting traffic from a neighbor AS through his/her policies. (If there are multiple peering connections between two ASes, the policies need to be consistent across them.) The business relationships, and thus the peering relationships they translate into, can be coarsely classified into a few common categories (Gao 2001, Subramanian et al. 2002):

customer/provider: A customer AS buys transit from a provider. Thus, the customer will advertise all its routes (subnetwork prefixes it originates) and routes to any customers it might have to the provider. It would also accept routes to all other Internet destinations from the provider and readvertise them to its customers. However, it would make sure not to readvertise routes learned from one provider to another, or to any peers, since that would mean it would in effect be providing transit between them.

peer/peer: Two ASes, for instance equal sized ISPs, may find it mutually advantageous to exchange traffic, (typically) at no charge, from customers of one AS to customers of the other (anticipating roughly equal amounts of traffic). Hence, only customer routes would be advertised to a peer; and routes learned from a peer are only readvertised to customers.

sibling/sibling: (Gao 2001) called ASes that provide mutual transit for each other siblings; and (Subramanian et al. 2002) conjectured that sibling relationships may occur between ASes that have common ownership. (Mutual transit is different from a peer/peer relationship since traffic is carried for the other party to other networks besides ones own customers.)

Call these the common peering Type-of-Relationships (ToRs) (Gao 2001). Customer/provider and peer/peer are the most common relationships (Gao 2001, Subramanian et al. 2002). However, more complex relationships also occur. For instance, a ToR may not span all prefixes (AS *a* may be a customer of *b* for some prefixes and a provider for others), or ASes may have mutual backup agreements in case of failures (Gao and Rexford 2001).

2.2 Basic Operation

In BGP, information about a route to a given destination IP prefix, is passed through a number of BGP route attributes, including the *local preference* for the route, the *AS path* to the destination, etc. Policies specify whether a given route is permitted and what preference it should be given based on its route attributes. Typically, the choice is controlled by setting the *local preference* attribute based on the values of other attributes. BGP then selects its preferred route as follows (in order of importance): 1) **local preference**—higher *local preference* is preferred; 2) **AS path**—shorter *AS path* is preferred; 3) **origin type**—e.g. a route learned from IGP is preferred over BGP; 4) **Multi-Exit Discriminator**—smaller *MED* is preferred if the next-hop AS is the same; 5) **iBGP/eBGP**—routes learned from external BGP (eBGP) peers are preferred over internal peers; 6) **IGP cost**—lower *intradomain routing cost* is preferred; 7) **Arbitrary tie-breaking**—a route from a router with a smaller *BGP ID* is preferred. Its preferred route is then passed to its neighbors, if policies so permit, who examine it, make a choice, and so on.

Intradomain routing protocols, such as OSPF, RIP, IS-IS, essentially perform a distributed all-pairs shortest path computation. The computation performed by a collection of BGP speaking routers is fundamentally different, because *i)* path ranking is not based on any universal cost function and *ii)* policies may reject paths (including the shortest path) (Griffin et al. 2002). Instead, Griffin et al. (2002) model the computation performed in BGP through the "*Stable Paths Problem*".

2.3 BGP Convergence

Varadhan et al. (1996) showed that there exists combinations of policies that can cause BGP to diverge, i.e. cause persistent route oscillations. Griffin et al. (2002) provided conditions under which policy configurations are guaranteed to converge, but also showed that determining whether a given policy configuration will converge is an NP-complete problem.

Gao and Rexford (2001) created a set of guidelines for the choice of route preferences in BGP policies and showed that if these guidelines are followed, convergence of the BGP routing system will be guaranteed. We briefly restate one of their results here. Let $first(r.as\_path)$ denote the first AS number in the AS path of route r, and $r.loc\_pref$ denote the local preference of r. Furthermore, let $customer(a)$, $peer(a)$, and $provider(a)$ be the set of customers, peers, and providers respectively, of AS a. Their simplest guideline, "Guideline A", requires that an AS (a) must always prefer routes to a customer AS over routes to peers or providers:

$$\begin{array}{ll} \text{if} & first(r_1.as\_path) \in customer(a) \;\; \text{and} \\ & first(r_2.as\_path) \in peer(a) \cup provider(a) \\ \text{then} & r_1.loc\_pref > r_2.loc\_pref \end{array}$$

That is, customer routes are given higher local preference than peer or provider routes. That guideline A leads to BGP convergence is proven in two steps. The first step, the proof of Lemma 5.1, is of primary interest here. In the proof of Lemma 5.1 they construct an activation sequence over ASes, such that, if the BGP routers' decision processes are run in that sequence each router will immediately select a stable route. It is a sequence of ASes as they assume that routers in an AS are activated "simultaneously", stating that the internal sequence is irrelevant. The AS sequence is achieved by recognizing that the customer/provider relationships for a Directed Acyclic Graph (DAG), which is exploited to derive the activation sequence. The sequence is divided into two phases. The first phase, involves all ASes that have a direct customer route to the destination, i.e. not traversing any peer or provider links. Since customer routes have higher priority, a sequence obeying the customer-provider DAG results in stable route selections. The second phase of the activation sequence involves the remaining ASes, i.e. all those that have to traverse a peer or provider link to reach the destination. For the second phase, they again construct a sequence based on the customer/provider DAG, and show that export policy restrictions for peer and provider links result in stable choices of routes.

For the second step of the proof of convergence for guideline A, they show that given any initial state and a fair activation sequence the system will similarly converge. Gao and Rexford argued that their guidelines are justified from an economic perspective (reducing cost) and hypothesized that a reason that widespread divergence has not been observed in the Internet is because the guidelines are generally obeyed for those reasons. A later measurement study (Wang and Gao 2003) also found that customer routes are generally preferred over peer routes, which are generally preferred over provider routes, in agreement with this conjecture.

3 POLICY-AWARE ROUTING MODEL

Since interdomain routing relies on having intradomain routes available, we integrate both into a single algorithm, using shortest path (in terms of link cost) intradomain routes as given by OSPF or other common intradomain routing protocols. To ensure termination of our algorithm we need a BGP configuration where the routes converge. The way we achieve this is to model the situation where network administrators obey the guidelines of Gao & Rexford (Gao and Rexford 2001). This also means that the constructive proof of Lemma 5.1 tells us most of what we need to know about how to construct our algorithm. However, there are two issues we need to resolve to build an algorithm from their result: *i)* we need to consider sibling relationships and how to deal with them, and *ii)* we need to decide what it means to activate all routers in an AS at once.

3.1 Sibling Relationships

A significant obstacle in creating a model from Gao & Rexford's result is that they do not consider sibling relationships. Yet, routing data analysis has suggested that sibling relationships occur in practice (Gao 2001, Subramanian et al. 2002). Siblings are non-trivial to accomodate in their framework because they lack the directionality of customer-provider relationships, but also do not have the export policy constraints of peers or providers. Hence, they cannot, for instance, simply be equated with peers. We use a modified version of the technique they describe for "Constrained Peer-to-Peer Relationships", where we form *clusters* of ASes having sibling relationships. As in their technique, their "*Assumption P*" (dictating the absense of customer-provider cycles or self-cycles) must hold; and we do not permit any customer-provider relationships within the cluster. Since sibling relationships are typically formed in cases of shared ownership (Subramanian et al. 2002), this requirement appears quite reasonable. A cluster is activated as one unit, and the activation of ASes in a cluster will be described in what follows.

Using the notation of (Gao and Rexford 2001), we impose the constraint "*Guideline S*" on sibling routes as follows. Let S_a be the transitive closure set of siblings of a (i.e. the sibling *cluster* around a). Define a *sibling-customer route* $r_{sc}(a)$ to be a route where

$$\begin{aligned}&\exists x \text{ in } r.as\_path \text{ such that}\\&\quad x \in customer(b) \text{ and } b \in S_a\end{aligned}$$

and define a *sibling-provider/peer route* $r_{spp}(a)$ to be a route where

$$\begin{aligned}&(\exists x \text{ in } r.as\_path \text{ such that}\\&\quad x \in provider(b) \cup peer(b)) \text{ and } b \in S_a\end{aligned}$$

then let $R_{sc}(a)$ be the set of all sibling-customer routes of a and let $R_{spp}(a)$ be the set all sibling-provider/peer routes of a. *Guideline S* states that

$$\begin{aligned}
&\text{if} && r_1 \in R_{sc}(a) \text{ and}\\
& && first(r_2.as\_path) \in customer(a)\\
&\text{then} && r_1.loc\_pref < r_2.loc\_pref\\
&\text{if} && (r_1 \in R_{sc}(a) \text{ and } (r_2 \in R_{spp}(a) \text{ or}\\
& && first(r_2.as\_path) \in peer(a) \cup provider(a))\\
&\text{then} && r_1.loc\_pref > r_2.loc\_pref\\
&\text{if} && r_1 \in R_{sc}(a) \text{ and } r_2 \in R_{sc}(a)\\
&\text{then} && r_1.loc\_pref = r_2.loc\_pref\\
&\text{if} && r_1 \in R_{spp}(a) \text{ and } (r_2 \in R_{spp}(a) \text{ or}\\
& && first(r_2.as\_path) \in peer(a) \cup provider(a))\\
&\text{then} && r_1.loc\_pref = r_2.loc\_pref
\end{aligned}$$

That is, indirect customer routes learned from siblings should be less preferred than direct customer routes, but more preferred than (direct or indirect) peer and provider routes. Moreover, we require that all indirect customer routes have the same preference and that all direct or indirect peer or provider routes have the same preference. This latter part is undesirable in its restrictiveness but ensures that, for each route "category", routing inside a sibling cluster reduces to a shortest AS path computation that is thus guaranteed to converge. However, this does require *repeated* activations of ASes in sibling clusters. For a sibling cluster of n ASes, we activate all ASes n times and thus, in effect, compute the shortest path routes using Bellman-Ford. Also, routes must not be propagated outside the cluster until the n:th activation to ensure that only stable routes are propagated. (The shortest path computation is complete after $n - 1$ activations.)

Correctness: Any given sibling cluster c can belong to either phase 1 or phase 2.

In phase 1: An AS in c that has a direct customer route is stable on its first activation. Other ASes in c perform a distributed shortest path computation and will thus receive one or more indirect customer routes. These routes are preferred to any peer/provider route, so the AS will be stable after n activations.

In phase 2: An AS in c has no direct customer route. All peer/provider routes have equal preference. By the proof of Lemma 5.1, all routes that could influence c are stable at its activation. Thus, after the shortest path computation (n activations), all ASes in c have stable routes.

3.2 Model Foundation

Let $G_N = (V_N, E_N)$ be a node level network graph such that V_N is the set of routers and hosts, and E_N is the set of physical adjacencies between them. Note that the links in G_N do not exactly correspond to physical links between routers and hosts. A physical point-to-point link corresponds to a link in G_N, but a Local Area Network (a multiple access link or otherwise) would correspond to a local clique in G_N. The foundation of the model is the annotated AS overlay graph $G_{AS} = (V_{AS}, E_{AS})$ such that G_{AS} is a directed graph, V_{AS} is the set of ASes and E_{AS} is the set of peering relationships between them. Each edge $e_{AS} \in E_{AS}$ is annotated with a peering *Type-of-Relationship* (ToR) from the set

set	notation
$\{CUSTOMER\_PROVIDER,$	$C \rightarrow P$
$PROVIDER\_CUSTOMER,$	$P \rightarrow C$
$PEER\_PEER,$	$P \rightarrow P$
$SIBLING\_SIBLING\}$	$S \rightarrow S$

Hence, each $v_N \in V_N$ belongs to exactly one $v_{AS} \in V_{AS}$ and multiple edges $e_N \in E_N$ may correspond to the same $e_{AS} \in E_{AS}$. Heuristics for inferring peering type of relationships in the Internet from BGP routing tables have been proposed in (Gao 2001, Subramanian et al. 2002).

Model: We model the usage of policies in the Internet by assuming that *Guideline A* (Gao and Rexford 2001) and *Guideline S* are followed for route preference and that typical export restrictions (Gao and Rexford 2001) are in place (as described in Section 2.1). (Wang and Gao 2003) found supporting evidence for assuming guideline A, and guideline S appears reasonable for sibling ASes belonging to a single organization. Note also that although we do not explicitly consider selective announcement of routes (Wang and Gao 2003) here, it can be readily included in this scheme. We also make the simplifying assumption that BGP speakers in each AS form a full iBGP peering mesh, i.e. there are no route reflectors. However, we believe that our scheme can be extended to include more complex iBGP configurations by including techniques from (Feamster, Winick, and Rexford 2004).

3.3 Algorithm

This section provides a high-level description of our Policy-Aware On-demand (PAO) routing algorithm. In our current implementation, a route contains the following information: destination prefix ($prefix$), local preference ($local\_pref$), AS path (AS_path), Multi-Exit Discriminator (MED), intradomain routing cost to destination or border router ($intra\_cost$), BGP ID for closest border router in route ($BGP\ ID$), next hop IP ($next\_hop$), path state ($path\_state$); where $prefix$, $local\_pref$, AS_path, MED, $BGP\ ID$, and $next\_hop$ are the BGP route attributes. Finally, $path\_state$ describes the peering relationships traversed in the AS path, as defined in (Gao 2001). Thus, $path\_state \in$ {up, top, down}. Let r be a valid route to destination d at router v. Then the value of $r.path\_state$ depends on whether node v is in the *up* section of the AS path, directly following a *peering* link, or on the *down* section of the path. v is in the *up* section if it can reach d through customer links exclusively. $r.path\_state = top$ if v first has to traverse a peering link and then customer links, otherwise $r.path\_state = down$. Additional BGP route attributes specified in the RFCs, such as communities, could be added as needed.

Given a source node and a destination prefix, the algorithm works in two stages: 1) AS clustering and activation sequence computation, and 2) working through the AS/cluster activation sequence computing routes. In stage 2, routing information for the given destination is propagated throughout the network. Finally, a source-route is built by traversing the the next-hop chain from the source to the destination. Since the algorithm floods routing information (for the given destination) throughout the graph, it will require at least $O(N)$ storage globally for N nodes.

Stage 1: The first step is to group sibling ASes into sibling clusters. Thus, from the AS level graph G_{AS}, the graph of ASes and clusters $G_{AS/CL}$ is formed. Next, the AS/cluster activation sequence is found by performing a modified topological sort on the $G_{AS/CL}$ customer-provider DAG. The sorting has to be modified to distinguish between phase 1 and phase 2 activations (see Section 2.3), and thus ensure that all phase 1 activations appear in the sequence before phase 2 activations. Hence, the end result of stage 1 is an activation sequence over ASes and clusters.

Stage 2: We initialize the node level graph G_N by giving each node, except the destination, a best route to the destination with infinite cost. The destination has zero cost to itself. In stage 2, we iterate over the activation sequence, expanding clusters to schedule internal ASes, and activate route selection one AS at a time. Each cluster, as it is encountered in the activation sequence, is expanded into its constituent ASes and these ASes are scheduled into the sequence repeatedly (see Section 3.1) in an arbitrary order. When a cluster is encountered, exporting routes outside the cluster is also disabled until the last iteration through the cluster ASes. Thus after the expansion step, the activation sequence consists exclusively of ASes. Each AS that is not part of a sibling cluster will be activated exactly once, and

ASes in clusters are activated n times, where n is the size of the cluster.

Routers in an AS that receive a route from outside the AS are placed in an activation set S_{in}. At the outset, the destination node is placed in the activation set of its surrounding AS to get the process started. Activation of routers in an AS starts from each router $x_i \in S_{in}$ and propagate the selected route from x_i to all routers in the AS while calculating the intradomain cost using Dijkstra's algorithm. Non-BGP speaking routers pick the best route based only on the intradomain distance to a BGP speaker (or the destination if it is in the same AS), while BGP speakers select using the full BGP decision process.

A route export predicate is used to filter out routes before readvertising them to external peers (outside the same AS). This predicate uses peering Type-of-Relationship classifications to essentially check routing paths for the "valley free property" (Gao 2001) (corresponding to readvertising restrictions described in Section 2.1). This information is captured in the $path\_state$ of the route and the ToR for peering link the route is to be readvertised over. Additional export policies defined by the model user can also be included at this point. The user defined export policies must not conflict with the export predicate. The predicate also does route loop detection and checks if readvertisement is outside an AS cluster and should be blocked for this reason in this activation.

A route that is allowed to be readvertised has import policies applied to it before it is passed to the receiving router. Primarily this means that the local preference for the route is set according to Guideline A and Guideline S. User defined import policies may then modify the local preference (and other attributes) within the constraints set by the guidelines. One minor note in relation to this is that it is difficult to express the preferences in Guideline S without the use of communities. So, since we have not, at this point, implemented communities in the model we pass hints about preference in the $local\_pref$ attribute. Strictly speaking, this is not allowed according to the BGP specification, but works for the limited information that is needed here. Implementing communities will resolve this issue.

Correctness: The correctness of the interdomain routes produced by the algorithm follows from the fact that the algorithm adheres to the BGP decision process for route selection and: *i)* for customer-provider and peer-peer relationships from the proof of Lemma 5.1 (Gao and Rexford 2001), and *ii)* the correctness for sibling relationships was discussed in Section 3.1. The correctness of intradomain routes follows trivially from Dijkstra's algorithm. However, the interaction between inter- and intradomain routing in the algorithm deserves a brief discussion. In the current algorithm we have simply assumed that internal routers (i.e. non-BGP speaking routers) will pick a route to the closest BGP speaker, which will have more information regarding external routes. Other ways of dealing with external routes, such as redistributing external route information internally, can bring significant complexity and instability issues so this is a question we will not deal with here. However, we believe that the algorithm could be modified to incorporate such cases as well.

We have implemented the PAO routing algorithm in Java in the SSFNet network simulator (SSFNet 2000) and in C++ in the iSSFNet/DaSSFNet network simulator (iSSFNet 2003). In both cases we currently use a source-routing scheme, similar to the NIx-vector scheme (Riley et al. 2000), to compute routes on-demand from end host to end host and forward packets. To verify the algorithm, we have simulated random samples up to about 500 ASes from a collected AS level Internet topology, annotated with peering relationships through Gao's heuristic (Gao 2001). In these simulations we sent traceroutes between all possible $N(N-1)$ source/destination pairs and collected the routing paths. The paths generated using the PAO routing algorithm were compared with paths from running the detailed BGP protocol model (`SSF.OS.BGP4`) to verify that all paths were the same.

4 PERFORMANCE

We compare the performance of the PAO routing algorithm to full emulation of BGP in terms of memory usage and execution time. The BGP model in SSFNet is used as the comparison baseline.

Preliminary asymptotic complexity analysis indicates that performance for the current implementation of on-demand computation combined with source node cached source-routes is highly dependent on communication patterns. Memory demand for source-routing ranges from much larger than BGP to much smaller; while worst case time complexity is better than BGP. Here we will focus on empirical results to illustrate this.

Peak memory usage can be written as $m_{\text{peak}} = \max_t \{m_E + m_R + m_T\}$, where m_E is memory used by model entities (routers, hosts, links), m_R is routing information storage, and m_T is memory needed to simulate traffic forwarding (packets, queues). Similarly total time can be written as $t_{\text{tot}} = t_E + t_R + t_T$, where t_E is model entity build and configuration time, t_R is route computation time, and t_T is the time taken simulate traffic. The following experiments were run on a Dell Latitude D800 laptop with an Intel Pentium M 1.6 GHz CPU and 2 GB of memory running Red Hat Linux 9.0. Using Sun's JVM 1.4.1_02, we measure peak memory usage (maximum from garbage collector reports during the run) and the total execution time excluding model build time t_E.

Sampled AS Topologies: Using a BGP routing table (RIB) collected at the Route Views Project on Nov. 19, 2003, we build an AS topology and use Gao's heuristic (Gao 2001)

Table 1: Sampled AS Topologies

$N_{\max}$	$\bar{N}$	avg. outdegree
100	78.9	2.49
200	151.5	2.82
400	307.1	3.15
600	451.5	3.33
800	639.2	3.39

to infer AS peering type of relationships. From this approximation of the Internet AS topology we sample subgraphs in a manner so as to avoid reducing the average outdegree too much. First of all, "edge trees" are removed (by repeatedly removing leaves) to bias the sampling towards the graph core. Then edges are removed at random with a given probability p, here $p = 0.2$. Finally all nodes and edges not belonging to the largest connected component are removed. We check if the remaining number of ASes N is not above a given target threshold $N_{\max}$, otherwise this whole process is repeated until $N \leq N_{\max}$. In the experiments here we generate 10 samples for each $N_{\max}$ and each AS is modeled by a single BGP speaking router. Some characteristics of the sampled topologies are given in Table 1. For comparison, the whole AS topology built from the RIB has an average outdegree of 4.08. Since our samples still have smaller average outdegrees we can expect routing costs for BGP to grow somewhat faster than indicated by our experiments.

Figure 1 shows peak memory usage and total execution time for simulation where traffic is either a single ping to one destination, or $N(N-1)$ pings between all pairs of routers. Memory growth is very slow for the on-demand algorithm for a single ping, illustrating the fact that orders of magnitude reductions in routing table memory are possible for sparse communication patterns. Because the implementation includes less detail than the BGP model (interaction with TCP etc.) there is also a certain gain for all-to-all communication in this case. Note, however, that all-to-all communication amongst many end hosts H, where $H \gg N$, will result in much greater memory usage for this type of simple source-routing scheme. The results also illustrate the fact that it is possible to gain substantially in route computation time for sparse communication patterns. On the other hand, computing routes for every source/destination pair is wasteful since routes are essentially computed for the whole graph *per destination*. That is, once BGP has converged for a certain destination, every router has a route to that destination. Similarly in the on-demand computation, a byproduct of computing a route from a certain source to a certain destination is that all other routers will also, at that moment, have a route to the destination.

Given a certain amount of knowledge about the communication patterns in the model, it is possible to exploit the computation that has been done also in the on-demand algorithm. We can speculatively cache routes in other nodes than the given source node, and thus trade some memory for reduced computation. For instance, if we know that there will be all-to-all communication, we can speculatively cache results in all nodes when it is computed for one source. When other sources later want to send to the same destination, the route is already known. The results marked "with precaching" were obtain in this fashion and demonstrate that the on-demand algorithm can compute routes more efficiently than full BGP convergence emulation.

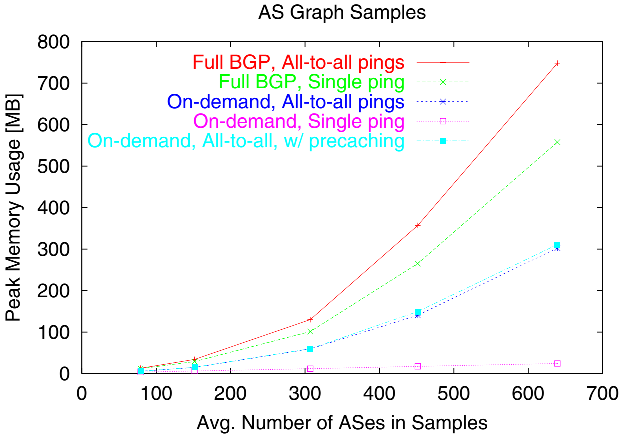
An Internet Backbone Topology: A second set of experiments uses a model of 6 large ISP networks, built primarily from Rocketfuel (Spring et al. 2002) data, to approximate an Internet backbone topology (Liljenstam et al. 2003). From the model described in (Liljenstam et al. 2003), we use the continental U.S. part of the 6 Rocketfuel topologies (cut level zero). Routers in an ISP network that have connections to other ISP networks are configured as BGP speakers.

Figure 2 shows memory usage for a series of experiments where the number of ISP networks included in the topology was increased from two to six. Since the majority of the routers are internal routers, we try to reduce the complexity of the intradomain routing simulation by choosing a simplified model of intradomain routing for the comparison. The experiments use a static computation of intradomain routes at the beginning of the simulation (the `SSF.OS.sOSPF` model). It does, consequently, maintain full intradomain forwarding tables, but reduces the state somewhat by not simulating full OSPF dynamics.

In these experiments a single traffic flow is started through the network. Using the full BGP model and the "static OSPF" model, it was not possible to increase the size beyond two ISP networks and get the simulation to run in the 2 GB available on the experiment machine. However, using on-demand route computation, there is a very slow growth in memory demand. Consequently, more complex internal network structure does not adversely affect the memory demands of the on-demand algorithm, and depending on traffic density, we can have orders of magnitude reductions in routing table storage also for this type of topology.

5 CONCLUSIONS

We have presented an algorithm for computing policy based interdomain routes (BGP routes) on-demand in a network simulation. The on-demand computation strategy is useful for simulations that require realistic forwarding paths for traffic in the network, but where the routes can be regarded as generally stable, so that routing dynamics is not a significant factor. (Consequently, it is not suitable for simulations where the primary objective is to investigate BGP routing convergence or other dynamics.) Some fairly mild and realistic constraints, proposed in (Gao and Rexford 2001) with



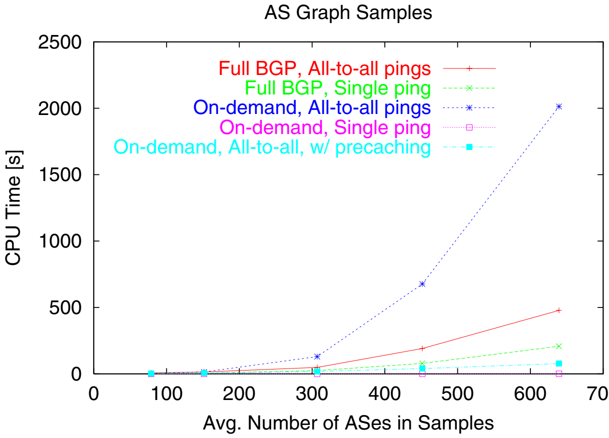


Figure 1: Memory and CPU Time Usage on Sampled Annotated Topologies from Internet AS Graph

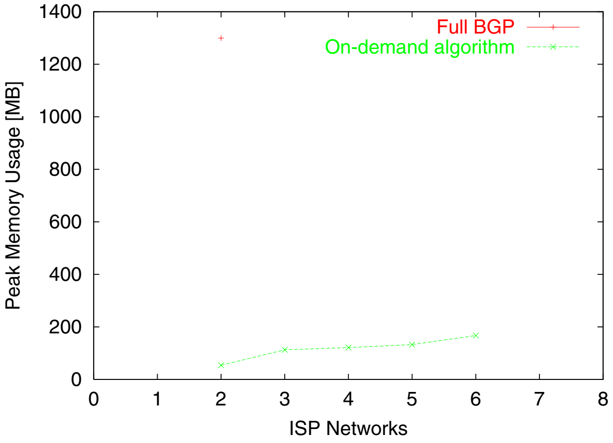


Figure 2: Memory Usage on Approximated Internet Backbone Topology

an extension to handle sibling relationships, are imposed on the BGP policies to ensure routing convergence and termination of the algorithm. Correctness was verified through experimental comparison with output from a detailed BGP model.

By computing routes on-demand, as proposed in (Riley et al. 2000), it is possible to reduce memory demands for routing storage by orders of magnitude compared to a full simulation of BGP; and by relying on the theoretical result in (Gao and Rexford 2001) we can reduce the computational cost compared to fully emulating BGP convergence. Our experiments show the possibility for large routing information memory savings for realistic topologies, such as sampled AS topologies and an Internet backbone model.

On-demand route computation opens up the possibility to explore space/time tradeoffs, something we see as an avenue for future work. Another important direction is to investigate ways to exploit route aggregation in this scheme.

ACKNOWLEDGMENTS

This research was supported in part by DARPA Contract N66001-96-C-8530, NSF Grant CCR-0209144. In addition this project was supported under Award No. 2000-DT-CX-K001 from the Office for Domestic Preparedness, U.S. Department of Homeland Security. Points of view in this document are those of the author(s) and do not necessarily represent the official position of the U.S. Department of Homeland Security.

REFERENCES

Feamster, N., J. Winick, and J. Rexford. 2004, June. A Model of BGP Routing for Network Engineering. *Proceedings of ACM SIGMETRICS*, 331–342.

Gao, L., and J. Rexford. 2001, Dec. Stable Internet Routing Without Global Coordination. *IEEE/ACM Transactions on Networking* 9 (6): 681–692.

Gao, L. 2001, Dec. On Inferring Automonous System Relationships in the Internet. *IEEE/ACM Transactions on Networking* 9 (6): 733–745.

Griffin, T., B. Shepherd, and G. Wilfong. 2002, April. The Stable Paths Problem and Interdomain Routing. *IEEE/ACM Transactions on Networking* 10 (2): 232–243.

Hao, F., and P. Koppol. 2003, July. An Internet Scale Simulation Setup for BGP. *Computer Communication Review* 33 (3): 43–58.

Huang, P., and J. Heidemann. 2001, Aug. Minimizing Routing State for Light-weight Network Simulation. *9th International Symposium on Modeling, Analysis and Simulation on Computer and Telecommunication Systems (MASCOTS)*, 108–116.

iSSFNet 2003. Available online at `www.crhc.uiuc.edu/jasonliu/projects/ssfnet` (accessed Aug 3, 2004).

Liljenstam, M., J. Liu, and D. Nicol. 2003, Dec. Development of an Internet Backbone Topology for Large-Scale Network Simulations. *2003 Winter Simulation Conference*, ed. Smith, Peters, White, Wilson, 694–702. New Orleans, LA.

Rekhter, Y., and T. Li. 1995, March. A Border Gateway Protocol 4 (bgp-4). RFC 1771.

Riley, G., M. Ammar, and R. Fujimoto. 2000, Sept. Stateless Routing in Network Simulations. *8th International Symposium on Modeling, Analysis and Simulation on Computer and Telecommunication Systems (MASCOTS)*, 524–531. San Francisco, CA.

Spring, N., R. Mahajan, and D. Wetherall. 2002. Measuring ISP Topologies with Rocketfuel. *ACM SIGCOMM Computer Communication Review* 32 (4): 133–145.

SSFNet 2000. Available online at <www.ssfnet.org> (accessed Aug 3, 2004).

Stewart, J. 1998, Dec. *BGP4: Inter-Domain Routing in the Internet*. Reading, MA: Addison-Wesley.

Subramanian, L., S. Agarwal, J. Rexford, and R. Katz. 2002, June. Characterizing the Internet Hierarchy from Multiple Vantage Points. *IEEE INFOCOM*, 618–627.

Varadhan, K., R. Govindan, and D. Estrin. 1996. Persistent Route Oscillations in Inter-domain Routing. Univ. of Southern California Information Sciences Institute, Marina del Rey, CA, ISI Tech. Rep. 96-631.

Wang, F., and L. Gao. 2003. Inferring and Characterizing Interent Routing Policies. *ACM SIGCOMM Conference on Internet Measurement*, 15–26.

AUTHOR BIOGRAPHIES

MICHAEL LILJENSTAM is a Post-Doc Research Associate at the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign. From 2000–2003 he was a Research Associate at the Institute for Security Technology Studies and Computer Science Department, Dartmouth College. His research interests include large-scale network simulation, security, routing, and modeling and simulation of wireless networks. He received his M.Sc. (1993) and Ph.D. (2000) from the Royal Institute of Technology, Stockholm, Sweden. His e-mail address is <mili@crhc.uiuc.edu>, and his web page is <www.crhc.uiuc.edu/~mili>.

DAVID M. NICOL is Professor of Electrical and Computer Engineering at the University of Illinois, Urbana-Champaign, and member of the Coordinated Sciences Laboratory. He is co-author of the textbook *Discrete-Event Systems Simulation*, and served as Editor-in-Chief at ACM TOMACS from 1997-2003. He is the General Chair of the 2004 Conference on Principles of Advanced and Distributed Simulation, and the General Chair of the 2006 Winter Simulation Conference. From 1996-2003 he was Professor of Computer Science at Dartmouth College, where he served as department chair, and at the Institute for Security Technology Studies served as Associate Director for Research and Development, and finally as Acting Director. From 1987-1996 he was on the faculty of the Computer Science department at the College of William and Mary; 1985-1987 he was a staff scientist at the Institute for Computer Applications in Science and Engineering. He has a B.A. in mathematics from Carleton College (1979), an M.S. (1983) and Ph.D. (1985) in computer science from the University of Virginia. His research interests are in high performance computing, performance analysis, simulation and modeling, and network security. He is a Fellow of the IEEE. His e-mail address is <nicol@crhc.uiuc.edu>.