TECHNICAL RESEARCH REPORT

Hierarchical Modeling for Network Performance Evaluation

by Mingyan Liu, John S. Baras

CSHCN T.R. 2000-8
(ISR T.R. 2000-26)

Hierarchical Modeling for Network Performance Evaluation

Mingyan Liu, John S. Baras
Center for Satellite and Hybrid Communication Networks
Department of Electrical Engineering and the Institute for Systems Research
University of Maryland at College Park, College Park, MD 20742, USA *

abstract

In this paper we present a hierarchical network model to estimate the connection blocking for large hierarchical networks. As networks grow in size, nodes tend to form clusters geographically and hierarchical routing schemes are more commonly used, and it is important that network modeling methods have scale-up capabilities. Loss networks and reduced load/fixed point models are often used to approximate call blocking probabilities and hence throughput in a circuit switched network. We use the same idea for estimating connection blocking in a data network with certain QoS routing schemes. However so far most work being done in this area is for flat networks with flat routing schemes. We aim at developing a more efficient approximation method for networks that have a natural hierarchy and/or when some form of hierarchical routing policy is used. We present hierarchical models in detail for fixed hierarchical routing and dynamic hierarchical routing policies, respectively, via the notion of network abstraction, route segmentation, traffic segregation and aggregation. Computation is done separately within each cluster (local) and among clusters (global), and the fixed point is obtained by iteration between local and global computations. We present results from both numerical experiments and discrete event simulations.

keywords
Hierarchical routing, Scale-up, QoS routing, Loss network, Reduced load/fixed point approximation, Connection blocking.

Introduction

In this paper we present a hierarchical model for estimating connection blocking probabilities in a hierarchical network and/or when some form of hierarchical routing is used.

*This work was supported by the Center for Satellite and Hybrid Communication Networks, under NASA cooperative agreement NCC3-528

The objective is not to artificially introduce hierarchy into a network model, but rather to develop a more efficient and scalable way of performance analysis for networks that bear a natural hierarchy. The loss network model has been extensively studied and used to estimate call blocking probability in a circuit switched network [1, 2, 3], [4]. In recent years, studies claim that this technology can be applied to packet switched networks as well, via the technique of effective bandwidth [5, 6]. We observe that this type of modeling can be used to study performance for data networks employing QoS routing. The notion of quality of service (QoS) has been proposed to capture the qualitatively or quantitatively defined performance contract between the service provider and the user applications. QoS routing aims at satisfying requested QoS requirements for every admitted connection by selecting network routes with sufficient resources. QoS routing is normally connection-oriented with resource reservation to provide the guaranteed service. Therefore meeting the QoS requirement of each individual connection and reducing the connection blocking rate are important in QoS routing, while overall throughput, average response time, etc., are the critical issues in traditional best effort routing [7].

One important routing strategy is hierarchical routing. Modern networks are getting larger and larger, and it is necessary that network engineering tools have scale-up capabilities. With the increase in size, a network tends to have clusters of nodes geographically, and hierarchical routing schemes are more commonly used in order to cope with large network size. On the other hand, research in this area has almost all on flat networks with flat routing schemes [1, 8, 2, 9]. This has provided us with strong motivation as well as applicability to develop a hierarchical model to estimate connection blocking.

We examine two types of hierarchical routing schemes and the corresponding end-to-end connection level models. One is fixed or near fixed routing with the typical example being OSPF [10], which is widely used for the Internet IP based routing. Under this routing scheme, routes are established based on shorted distance principle, with ties broken according to lower IP address. Considering the fact that links normally fail on a much larger time scale compared to connection durations, this is a fixed routing scheme. In this case, the hierarchy of the network is primarily geographical, which comes from the fact that each workstation/network node within a LAN is connected to remote nodes via gateways on different levels. The abstraction of the physical network results in interconnected gateways on higher layer(s).

The other type is dynamic hierarchical routing with a typical example being PNNI. Various proposals for QoS routing in the Internet also fall under this category [11, 12]. In this case, the centering point is “partial information”. Networks are divided into clusters or peer groups that consist of neighboring nodes, some of which being “border nodes” that are connected to other peer groups. All non-border nodes are only aware of its own group, and all border nodes are aware of its own group and border nodes of other groups. Border nodes represent some form of aggregation of the rest of the network to non-border nodes within the same group. Routes are established on different layers based on complete information within a group and aggregated information between groups.

Our goal is to build a hierarchical end-to-end model that closely couples with the hierarchical nature of routing and uses only partial information on different layers. By segregating network into layers we can also develop models for situations where different routing schemes are used from group to group, or from layer to layer.

In the next section we briefly describe the loss network model and reduced load approximation we use as the building blocks for our hierarchical model. We then describe network abstraction and aggregation in Section 3. Hierarchical models for fixed hierarchical routing and dynamic hierarchical routing are presented in Section 4 and 5, respectively. In Section 6 we present numerical results for fixed hierarchical routing, which gained 3-4 fold improvement in computational cost. We also present the results of our model on dynamic hierarchical routing compared to simulation. Section 7 concludes this paper.

Reduced Load Approximation in a Hierarchical Model

For self-sufficiency purposes we briefly describe in this section the reduced load approximation method that has been extensively studied for loss networks.

In a loss network a connection requires certain amount of bandwidth on every link on a path between the source and the destination. If the network has the required bandwidth on those links when the request arrives (usually as a Poisson process), the connection is admitted and the requested capacities are reserved till the connection is completed. Otherwise the request is rejected. Connection blocking probability is the probability that a request finds the network unavailable when it arrives and is thus rejected.

The computational complexity of loss networks [13] leads to various type of approximation schemes, of which the *reduced load* approximation or *fixed point* approximation is extensively studied and widely used. The construction of a reduced load model relies on two assumptions, the *link independence assumption*, which assumes that the blocking occurs independently from link to link so that the probability that a call is accepted on a certain route is the product of the probabilities that the call is accepted on each individual link on that route, and the *Poisson assumption*, which assumes that the traffic flow onto each link is Poisson and that the corresponding traffic rate is the original offered rate thinned by blocking on other links of that route, thus called the reduced load. It can be shown that for fixed routing under certain limiting regime the assumptions hold [1].

Therefore for fixed routing, the reduced load ν_{js} on link j of traffic type s is

$$\nu_{js} = \sum_{r} \lambda_{rs} b_{js} I[j \in r] \prod_{i \in r, i \neq j} a_{is}$$

where λ_{rs} is the offered traffic load of class s on route r, b_{js} is the bandwidth requirement of class-s traffic on link j, I is the indication function, and a_{js} is the probability that a class-s connection is accepted on link j. a_{js} can be computed from the finite state Markov chain model of link j, e.g., the Kaufman recursion [3]

$$a_{js} = \sum_{n=0}^{C_j - b_{js}} p_j(n)$$

where C_j is the link capacity and the state distributions

$$np_j(n) = \sum_s b_{js} \frac{\nu_{js}}{\mu_{rs}} p_j(n - b_{js}), \quad n = 1, ..., C_j,$$

with μ_{rs} being the departure rate of the connection. The end-to-end blocking probability is thus

$$B_{rs} = 1 - \prod_{i \in r} a_{is}.$$

More complicated reduced load approximations, as well as the mapping from ν_{js} to a_{js}, have been developed to model different routing schemes, e.g., alternative routing and adaptive routing, and call admission control, e.g., trunk reservation, which is beyond the scope of this paper [14, 9], [2, 8].

To summarize, the input to a reduced load model is the offered traffic load between nodes and the iteration process computes updates of the reduced load and the blocking probability for each link until it converges. Finally the end-to-end blocking probability is calculated using the link blocking probability and the link independence assumption.

The idea behind the hierarchical models we present in subsequent sections is to apply the reduced load approximation to different parts (clusters and layers) of the network separately so that computation is localized. Each local computation (for a single cluster) takes as input the results from other local computation and global computation (for portions of the network between clusters – higher layer) by using the link independence assumption. For different layers or clusters different reduced load model can be used depending on the routing schemes under consideration. By doing so we want to achieve the following:

- Faster approximation while generating the same results compared to a flat approximation model;
- A method of modeling networks with hierarchy and where different routing schemes are used at parts of the network.

A central component of this model is thus to derive the offered traffic load for each clusters so that the reduced load model can be applied. We obtain this by route segmentation, traffic segregation and aggregation.

Network Abstraction We only consider large networks that have either physical hierarchies or routing hierarchies vs. a complete mesh since a hierarchical model promises clear incentives

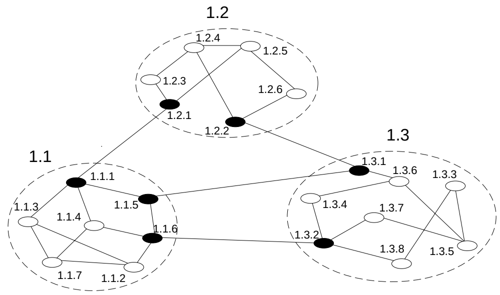

Figure 1: Network with three clusters– Layer One

only for the former if it is at all possible for the latter. Throughout the paper we use a two-layer example shown in Figure 1.

There are three clusters in this example, with the dash-circles surrounding each one. Each group has a label/address, e.g., 1.1 indicates Layer 1, Peer Group 1. Each node has an address as well, e.g., 1.1.3 is Node 3 of Peer Group 1.1. All border nodes are shown in black and non-border nodes are shown in white. A cluster can have a single or multiple border nodes. A border node can be connected to different clusters, e.g., Node 1.3.1. A non-border node does not necessarily have a direct link connected to border nodes, although this is often true with IP networks. Note that all links on this layer are actual, physical links.

The way aggregation and abstraction are done is as follows:

- All border nodes are kept in the higher layer – in this case Layer 2;
- Border nodes belonging to the same cluster are fully connected via "logical links".

This results in the Layer 2 abstraction shown in Figure 2.

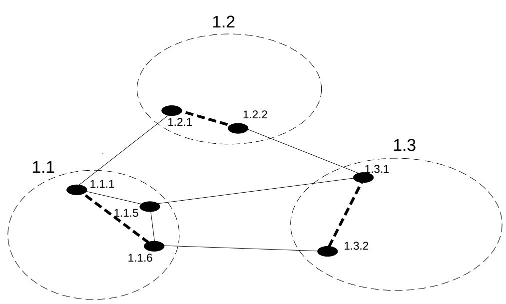

Figure 2: Network with three clusters – Layer Two

A logical link may corresponds to an actual link on the lower layer, e.g., Link 1.1.1 $\longleftrightarrow$ 1.1.5. The real logical links are shown in dashed lines, indicating a feasible path rather than a direct link.

As pointed out in [15], creating a logical link between each pair of border nodes is the *full-mesh* approach, while collapsing the entire group into a single point is the *symmetric-point* approach. Our aggregation approach is a full-mesh one. While it may not be the most economic way of aggregation, this model clearly couples best with the underlying network physical structure and routing structure. It's worth pointing out that a bandwidth parameter is often assigned to a logical link, e.g., representing the maximum/average available bandwidth on the paths between two border nodes, and this may cause problems when different paths overlap [12]. It is obvious that some form of aggregated information has to be associated with either the logical links or the border nodes. However, as we will see in subsequent sections a bandwidth parameter is not necessarily the parameter in our model for calculation on the higher layer, thus avoiding the aforementioned problem. In our model, for the fixed routing case, this parameter is the blocking probability resulted from previous iterations within the group, and for the dynamic routing case, this parameter can be implies costs, hop number or other criteria based on the dynamic/QoS routing policies being used.

Hierarchical Model for Fixed Routing

Notations

$G(1.n)$: the n^{th} cluster/peer group on Layer 1, where $n = 1, ..., N_1$, and N_1 is the total number of clusters in Layer 1.

$1.n.x_i$: node x in cluster $G(1.n)$, where $i = 1, ..., X_n$, and X_n is the total number of nodes in $G(1.n)$.

$1.n.y_i$: border nodes in cluster $G(1.n)$, where $i = 1, ..., Y_n$, and Y_n is the total number of border nodes in $G(1.n)$.

$1.n.x_1 \longrightarrow 1.n.x_2$: link from node $1.n.x_1$ to node $1.n.x_2$. Links in our model are directional.

$\lambda_s(1.n_1.x_1 \longrightarrow 1.n_2.x_2)$: offered load for class-s traffic from source $1.n_1.x_1$ to destination $1.n_2.x_2$, where $s = 1, ..., S$, and S is the total number of different traffic classes. It is also written as λ_{ps} with p as the p^{th} source-destination node pair.

$\mathcal{P} : (1.n_1.x_1 \longrightarrow 1.n_2.x_2)$: the route between node $1.n_1.x_1$ and $1.n_2.x_2$. $\mathcal{P}_p$ is the route for the p^{th} node pair.

Route and Route Segments

For our modeling purposes, each route is broken down into route segments as follows.

Path $\mathcal{P}(1.n_1.x_1 \longrightarrow 1.n_2.x_2)$ is a sequence of directed links with beginning and ending nodes. We break down a route into segments whenever a route exits or enters a cluster. So a route segment can be one of the following: source $\longrightarrow$ destination (when both source and destination nodes belong to the same cluster, same as original route); source $\longrightarrow$ border node; border node $\longrightarrow$ border node; border node $\longrightarrow$ destination.

Therefore, a typical route $\mathcal{P}(1.n_1.x_1 \longrightarrow 1.n_2.x_2)$ is segmented into the following k segments, assuming that $n_1 \neq n_2$ and that neither $(1.n_1.x_1)$ nor $(1.n_2.x_2)$ is a border node:

$$\begin{aligned}
&\mathcal{P}^1 : (1.n_1.x_1 \longrightarrow 1.n_1.y_2)\\
&\mathcal{P}^2 : (1.n_1.y_2 \longrightarrow 1.n_i.y_1)\\
&\mathcal{P}^3 : (1.n_i.y_1 \longrightarrow 1.n_i.y_2)\\
&...\\
&\mathcal{P}^{k-1} : (1.n_j.y_2 \longrightarrow 1.n_2.y_1)\\
&\mathcal{P}^k : (1.n_2.y_1 \longrightarrow 1.n_2.x_2)
\end{aligned}$$

where y_1 represents a border node from which traffic enters a cluster, and y_2 represents a border node from which traffic exits a group. These vary depending on the source-destination node pair. Note that when a segment $\mathcal{P}(1.n_i.y_1 \longrightarrow 1.n_i.y_2)$ exists, the route traverses an intermediate cluster before reaching the destination cluster. We denote the set of route segments for the p^{th} source-destination node by $\mathcal{P}_p^{'}$, and the segments $\mathcal{P}_p^1, ..., \mathcal{P}_p^k$, respectively.

The reason for a segmentation like the above is to segregate local computation (within each cluster) and higher layer (inter-cluster) computation.

Initial Offered Load and Local Relaxation

With the segmentation of routes, we need a corresponding way of representing the offered traffic load prior to running the hierarchical algorithm.

The offered load of class-s traffic of the p^{th} node pair $(1.n_1.x_1, 1.n_2.x_2)$ is $\lambda_s^0(1.n_1.x_1 \longrightarrow 1.n_2.x_2)$. We substitute this with a combination of the following, in a similar way as route segmentation:

$$\begin{array}{ll}
\lambda_{ps}^0(1.n_1.x_1 \longrightarrow 1.n_1.y_2) & \text{source cluster traffic}\\
\lambda_{ps}^0(1.n_1.y_2 \longrightarrow 1.n_i.y_1) & \text{inter-cluster traffic}\\
\lambda_{ps}^0(1.n_i.y_1 \longrightarrow 1.n_i.y_2) & \text{cluster } i \text{ traffic}\\
... & \\
\lambda_{ps}^0(1.n_j.y_2 \longrightarrow 1.n_2.y_1) & \text{inter-cluster traffic}\\
\lambda_{ps}^0(1.n_2.y_1 \longrightarrow 1.n_2.x_2) & \text{destination cluster traffic}
\end{array}$$

These terms all take the value of the initial offered load $\lambda_s^0(1.n_1.x_1 \longrightarrow 1.n_2.x_2)$. Thus we have complete traffic input information (together with route segments) for each cluster.

For the i^{th} cluster, offered loads indexed with same node pair are added up to represent the aggregated traffic for this node pair. We assume that at least one of the nodes is a border node since no aggregation process is needed in cases where both nodes are non-border nodes within the same group. Without loss of generality, assume that the destination node is a border node,

$$\lambda_s^1(1.n_i.x_1 \longrightarrow 1.n_i.y_2) = \sum_{\{p:(1.n_i.x_1 \longrightarrow 1.n_i.y_2) \in \mathcal{P}_p^{'}\}} \lambda_{ps}^0(1.n_i.x_1 \longrightarrow 1.n_i.y_2). \quad (1)$$

Note that the source node x_1 can also be a border node.

In doing so, we assume initial condition of zero inter-cluster blocking, and zero blocking in remote clusters. This is part of the initial values and blocking on inter-cluster links are calculated in the next step of the algorithm.

The reduced load model for fixed routing is then applied to every cluster separately using these offered loads to calculate group-wide blocking probabilities:

$$\begin{aligned}
B_s(\mathcal{P}^1) &= B_s(1.n_1.x_1 \longrightarrow 1.n_1.y_2) \\
B_s(\mathcal{P}^3) &= B_s(1.n_i.y_1 \longrightarrow 1.n_i.y_2) \\
&\ldots \\
B_s(\mathcal{P}^k) &= B_s(1.n_2.y_1 \longrightarrow 1.n_2.x_2)
\end{aligned}$$

Reduced Load and Higher Layer Relaxation
On the higher layer (second layer in our example), only border nodes exist. We construct a new network with border nodes, inter-group links and logical links as illustrated in Figure 2. For this logical network we consolidate the previous route segments into three parts: within the source cluster ($\mathcal{P}^1$), between source and destination clusters ($\mathcal{P}^o = \mathcal{P}^2 \cup \ldots \cup \mathcal{P}^{k-1}$) and within the destination cluster ($\mathcal{P}^k$). We have the following offered load for the second segment:

$$\begin{aligned}
&\lambda_s^1(1.n_1.y_2 \longrightarrow 1.n_2.y_1) = \\
&\lambda_s^0(1.n_1.y_2 \longrightarrow 1.n_2.y_1) + \\
&\sum_{\{p:(1.n_1.y_2 \longrightarrow 1.n_2.y_1) \in \mathcal{P}'_p\}} \lambda_{ps}^0(1.n_1.y_2 \longrightarrow 1.n_2.y_1) \cdot \\
&B_s(1.n_1.x_1 \longrightarrow 1.n_1.y_2) \cdot B_s(1.n_2.y_1 \longrightarrow 1.n_2.x_2)
\end{aligned}$$

i.e.,

$$\begin{aligned}
\lambda_s^1(\mathcal{P}^o) &= \lambda_s^0(\mathcal{P}^o) + \\
&\sum_{\{p:\mathcal{P}^o \in \mathcal{P}'_p\}} \lambda_{ps}^0(\mathcal{P}_p^o) \cdot B_s(\mathcal{P}_p^1) \cdot B_s(\mathcal{P}_p^k).
\end{aligned}$$

This is the initial offered load thinned by blocking in both the source and destination clusters.

We now have the complete traffic input on the higher layer. We apply again the reduced load approximation to this layer and calculate second-layer end-to-end blocking probabilities. Note that on this layer, we do not have a "capacity" parameter for the logically links, but instead, an end-to-end blocking probability resulted from previous approximation within the cluster (local approximation). This is kept fixed throughout the approximation on this layer and is only updated through local approximation. The result of this step is the blocking probability between border nodes:

$$B_s(1.n_i.y_1 \longrightarrow 1.n_i.y_2) \quad \text{and}$$
$$B_s(1.n_i.y_2 \longrightarrow 1.n_j.y_1).$$

Iterations
Using the results from the higher layer approximation, update the offered load in (1) with the blocking probabilities calculated from the higher layer:

$$\begin{aligned} &\lambda_s^1(1.n_i.x_1 \to 1.n_i.y_2) = \\ &\sum_{\{p:(1.n_i.x_1 \to 1.n_i.y_2 \in \mathcal{P}'_p\}} \lambda_{ps}^0(1.n_i.x_1 \to 1.n_i.y_2) \cdot \\ &\prod_{\{k:\mathcal{P}^k \neq (1.n_i.x_1 \to 1.n_i.y_2)\}} B_s(\mathcal{P}_p^k). \end{aligned} \tag{2}$$

This is essentially the original offered load thinned by blocking on inter-group links and remote groups. This becomes the new input for local relaxation. Local and higher layer relaxations are then repeated till the difference between results from successive iterations are within certain criteria.

Hierarchical Model for Dynamic Hierarchical Routing

There are numerous existing and proposed dynamic/QoS hierarchical routing schemes, each of which results in different end-to-end performances determined by the scope and design trade-off of the routing scheme. Our primary goal here is not to design an end-to-end model for each of these schemes. Rather, we attempt to present an end-to-end performance modeling framework that considers a generic type of dynamic hierarchical routing, which captures some of the most basic properties of a majority of such routing schemes. We make assumptions for simplicity purposes, but our work shows how an end-to-end performance model can be closely coupled with routing policies to provide an efficient way of analysis. Furthermore, our model enables us to analyze situations where different routing schemes are used on different levels of a networks.

Dynamic Hierarchical Routing
One key property of any dynamic hierarchical routing is *inaccurate/incomplete information* [16]. A node has complete information on its own group, but only aggregated information on other groups advertised by the border nodes. So a node typically sees its own group in detail but other groups "vaguely", in form of certain representation of the aggregated information, which is inevitably incomplete and inaccurate. This aggregated information can be one or more of various metrics specified by the routing algorithm: implied cost of a group (cost/additional blocking incurred by allowing a traffic stream to go through), maximum available bandwidth between border node pairs, delay incurred by going through a group, etc.. This information is typically associated with and advertised by border nodes.

In source routing, a path is selected with detailed hop-by-hop information in the originating group but only group-to-group information beyond the originating group. The detailed routing within each other group is determined locally. For example, a route from 1.1.7 to 1.3.3 is selected from the source point of view as 1.1.7 $\longrightarrow$ 1.1.6 $\longrightarrow$ 1.3, or as 1.1.7 $\longrightarrow$ 1.1.1 $\longrightarrow$ 1.2 $\longrightarrow$ 1.3. The choice of routes within a group is determined using shortest path routing, least loaded routing and so on, along with the aggregated information advertised by border nodes. We do not specify the details in formulating the model since they do not affect the general method of analysis. However we give specific examples for numerical and simulation studies.

In our model, we focus on dynamic hierarchical source routing where routes are selected in a way described above. We do not consider crankback in which a connection is routed on an alternative route if the first choice is not available. A call is blocked if the route selected according to the dynamic routing policy does not have the required bandwidth.

Probabilistic Offered Load Distribution and Traffic Aggregation

With dynamic hierarchical routing the model becomes more complicated because there is no longer a single fixed route between nodes. The key point of the model is to successfully separate traffic from cluster to cluster. In order to do so, we made the following observation. One of the main advantages of dynamic routing is load balancing, i.e., dynamically distribute traffic flow onto different paths of the network to achieve greater utilization of network resources. We argue that under steady state, a particular traffic flow (defined by class, source-destination node pair) is distributed among all feasible routes, and among multiple border nodes that connect to other groups. (This problem does not exist when there is only one border node. Routes are still dynamically chosen, but all routes ultimately go through that single border node.) The fraction of a traffic flow that goes through a certain border node is directly related to the aggregated information/metrics for the group-to-group route the border node advertises, and remains relatively fixed under steady state.

Based on this, for a pair of nodes belonging to different clusters, the feasible route set is divided into three subsets: route segments within the source cluster, route segments between clusters and route segments within the destination cluster. In this section we use $\mathcal{P}$ to represent a set of routes since multiple routes are allowed for each node pair in dynamic hierarchical routing.

To simplify notation, assume the route does not traverse an intermediate cluster (extensions can be made). For route set $\mathcal{P}(1.n_1.x_1 \longrightarrow 1.n_2.x_2)$ the subsets are:

$$\begin{aligned} &\mathcal{P}^1 : (1.n_1.x_1 \longrightarrow 1.n_1.y_i), \\ &\mathcal{P}^2 : (1.n_1.y_i \longrightarrow 1.n_2.y_i), \quad \text{and} \\ &\mathcal{P}^3 : (1.n_2.y_i \longrightarrow 1.n_2.x_2), \end{aligned}$$

where y_i indicates all possible border nodes. Each of these segments presents possibly many routes. We are simply breaking down the initial source-destination into multiple intermediate source-destination node pairs so that routes within a group and between groups are segregated.

We rewrite the second subset as $\mathcal{P}^2(1.n_1 \to 1.n_2)$, a collection of routes from source cluster $1.n_1$ to the destination cluster $1.n_2$. For each route in $\mathcal{P}^1$, we have the initial offered load

$$\lambda^0_{ps}(1.n_1.x_1 \to 1.n_1.y_i) = a_i \lambda^0_s(1.n_1.x_1 \to 1.n_2.x_2)$$

where $\lambda^0_s(1.n_1.x_1 \to 1.n_2.x_2)$ is the offered load of the class-s traffic for node pair $(1.n_1.x_1 \to 1.n_2.x_2)$, and each border node y_i gets to route a portion a_i of the traffic with $\sum_i a_i = 1$.

The traffic is further distributed to each route in $\mathcal{P}^2$ based on the underlying routing scheme used on the second layer:

$$\lambda^0_{ps}(1.n_1.y_i \to 1.n_2.y_j) = a_i q_{ij} \lambda^0_s(1.n_1.x_1 \to 1.n_2.x_2)$$

where $\sum_j q_{ij} = 1$ for all i, and can be calculated based on the reduced load mode used for this layer.

Finally for the routes in the destination cluster, $\mathcal{P}^3$, the initial offered load is

$$\begin{aligned} &\lambda^0_{ps}(1.n_2.y_j \to 1.n_2.x_2) = \\ &\sum_i a_i q_{ij} \lambda^0_s(1.n_1.x_1 \to 1.n_2.x_2) \end{aligned}$$

assuming the initial condition of zero blocking else where.

Within each cluster the traffic is then aggregated over all node pairs that have the same route in ones of there route sets, similar to what we described in the previous section.

Updates and Iterations

From the aggregate traffic we apply the reduced load model to each cluster and compute

$$\begin{aligned} &B_s(1.n_1.x_1 \to 1.n_1.y_i), \\ &B_s(1.n_1.y_i \to 1.n_2.y_j), \\ &B_s(1.n_1.y_j \to 1.n_2.x_2). \end{aligned}$$

As discussed earlier, the distribution of traffic flow onto the different inter-cluster routes should match the aggregated information (delay, blocking probability, implied cost, available bandwidth, etc.) advertised by different border nodes. Ultimately one of the goals for any dynamic routing scheme is to balance traffic load on different alternative routes, and the end result is that these alternative routes should have equivalent QoS under steady state. For example, if we use blocking probability as a criteria to adjust the traffic distribution $a_i, i = 1, 2, ..., n$, with n being the total number of border nodes in a cluster, then the border node with a blocking probability higher than median gets a decreased portion, and border nodes with a blocking probability lower than median gets an increased portion:

$$a_i := a_i + \delta \quad \text{if} \quad B_s(1.n_1.y_i \to 1.n_2) < B_m;$$
$$a_i := a_i - \delta \quad \text{if} \quad B_s(1.n_1.y_i \to 1.n_2) > B_m,$$

where $B_s(1.n_1.y_i \to 1.n_2)$ is the average blocking over all routes from $1.n_1.y_i$ to $1.n_2$, and δ is a small incremental value and B_m is the median blocking probability among all routes. Alternatively a_i can be set to be inversely proportional to the blocking probabilities with sum 1. Other means of relating traffic distribution to route QoS can also be specified.

q_{ij} is updated accordingly based on the reduced load model we use for the second layer. An example in least-loaded routing can be found in [17].

Using these new distribution values along with the blocking probabilities we update the aggregated traffic load for node pairs within the same cluster, similar to the fixed routing scenario. Another round of iteration is then started and the process continues until both the distribution a_i and the link blocking probabilities converge.

Numerical Results

In this section we present numerical experiment results for the network example shown in 1 using fixed hierarchical routing scheme, and dynamic hierarchical routing scheme. This is a 21-node, 30-link, 3-clusters, 2-layer network model. We use single class of traffic requiring unit bandwidth. Link capacities varies from 80 to 160. We use the offered traffic load between node pairs at a "nominal" level. The intensity of this traffic load is shown in Table 1, in which *load* is defined as the ratio between the total rate of traffic coming out of a node and the total out-going link bandwidth connecting to this node. At the nominal level, the value of this ratio for each node is around 0.05. In addition to this offered traffic load we also define a "weight" in our experiment as a multiplier to the nominal traffic, so that we get twice, three times of the nominal traffic, etc.. The complete data on link capacities and traffic rates can be found in [18].

Fixed Hierarchical Routing

Since when using fixed hierarchical routing a network can always be treated as flat, we compare the performance of flat fixed-point approximation (FPA) and the hierarchical fixed-point approximation. It can be shown [1] that under certain limiting regime for fixed routing the fixed point approximation is asymptotically correct.

Results of Flat FPA and Hierarchical FPA

Table 2 is a comparison between flat fixed-point approximation and hierarchical fixed-point approximation on individual link blocking probabilities (end-to-end blocking probabilities are computed directly from these for fixed routing). We used seven times nominal traffic (weight = 7).

Table 1: Nominal offered traffic load

Node	Rate	Cap.	load
1.1.1	11.15	270	0.041296
1.1.2	7.20	160	0.045000
1.1.3	10.60	180	0.058889
1.1.4	7.90	150	0.052667
1.1.5	8.60	210	0.040952
1.1.6	8.95	220	0.040682
1.1.7	10.45	180	0.058056
1.2.1	8.95	210	0.042619
1.2.2	8.80	220	0.040000
1.2.3	6.65	130	0.051154
1.2.4	9.15	180	0.050833
1.2.5	9.70	180	0.053889
1.2.6	7.95	140	0.056786
1.3.1	9.55	230	0.041522
1.3.2	12.0	290	0.041379
1.3.3	3.70	80	0.046250
1.3.4	7.90	140	0.056429
1.3.5	8.25	160	0.051562
1.3.6	9.00	180	0.050000
1.3.7	8.00	130	0.061538
1.3.8	5.75	100	0.057500

We see that the hierarchical scheme gives very close results compared to that of the flat approximation scheme, but achieved $3 \sim 4$-fold improvement in computation.

Varying Traffic Load

Figure 3 is a comparison between the runtime of flat FPA and hierarchical FPA while varying the traffic load by increasing the value of "weight", which is multiplied to the nominal traffic rate.

We see that as the traffic load increases, the gain in computation savings becomes significant. More importantly, the computation of the hierarchical FPA only increases marginally with the increase of traffic load.

Varying Cross-group Link Capacity

Since the sequence of iteration is determined by clusters, it is reasonable to expect that

Table 2: Comparison of results, weight = 7.

link	Hier. FPA	Flat FPA
(1.1.1-1.1.5)	0.235701	0.235704
(1.1.1-1.2.1)	0.409961	0.409955
(1.1.2-1.2.6)	0.526158	0.526159
(1.1.4-1.1.6)	0.148341	0.148333
(1.1.5-1.1.6)	0.006189	0.006190
(1.1.2-1.1.7)	0.000000	0.000000
(1.1.5-1.3.1)	0.004523	0.004523
(1.1.6-1.3.2)	0.054767	0.054766
(1.2.2-1.3.1)	0.122970	0.122967
(1.2.2-1.2.4)	0.000007	0.000007
(1.3.3-1.3.5)	0.000000	0.000000
time (sec)	13.90	42.76

changes to capacities of links in a particular cluster/layer will have an effect on how much faster the hierarchical scheme runs comparing to the flat scheme.

Figure 4 is a comparison between the runtime of flat FPA and hierarchical FPA while varying the capacities of links connecting two different clusters. We see that the run time difference does not change much with the increase in link capacities. However the gain slightly increases when the link capacities are reduced. A possible explanation is that these links can easily become bottlenecks when the capacities are reduced, and by separating the global computation from local computation we get faster convergence.

Dynamic Hierarchical Routing

We use the same network example, with shortest path routing within each cluster, but use least-loaded routing between clusters. Least-loaded routing (LLR) is a form of bandwidth-optimization QoS routing [7]. A source node chooses a border node based on the advertised average blocking between the border node and the destination cluster $B_s(1.n_1.y_i \rightarrow 1.n_2))$, and a border node chooses the route that has the most free capacity among all route from itself to the destination cluster. We used the reduced load model for least-loaded routing we developed in [17]. The distribution of traffic among border nodes is inversely proportional to the advertised blocking probability. and the distribution q_{ij} is the probability that route $(1.n_1.y_i \rightarrow 1.n_2.y_j)$ has the most free bandwidth.

The following tables show the comparison between the results of the hierarchical model and the discrete event simulation (DES), with weight being 5, 10, 15 and 20, respectively.

We see as the traffic increases, the model generates better approximations. Overall the approximation is satisfactory. The run time for the approximation is around 11-15 seconds while the simulation typically takes 5-20 minutes to converge depending on the traffic load.

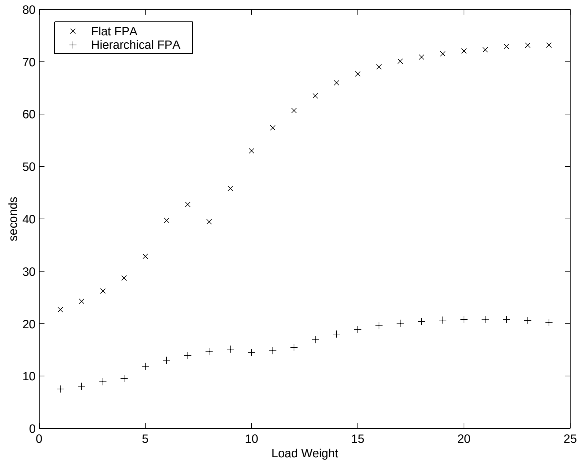

Figure 3: Run time vs. traffic load.

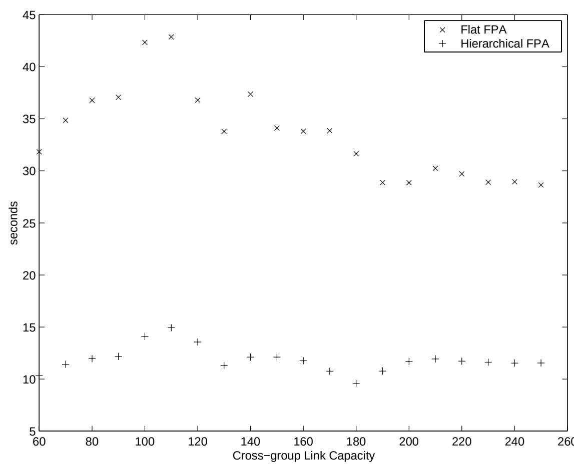

Figure 4: Run time vs. link capacities.

Conclusion

In this paper we presented the hierarchical reduced load approximation method for networks with either fixed hierarchical routing or dynamic hierarchical routing policies. It can also be used in cases where different routing schemes are used in different regions of a network. Our numerical experiment results showed significant improvement in computational cost, and the validity of this method. Our experiment network is relatively small, however we believe that this is a novel approximation method for efficient and scalable performance analysis for much larger networks.

Table 3: Comparison of results, weight = 5.

Node Pair	Hier. FPA	DES
(1.1.3-1.3.3)	0.00077	0.00000
(1.1.2-1.2.4)	0.00000	0.00000
(1.1.6-1.3.7)	0.00000	0.00000
(1.1.1-1.2.3)	0.00000	0.00000
(1.2.1-1.2.6)	0.00191	0.00453
(1.3.1-1.3.8)	0.00000	0.00000
(1.3.5-1.3.6)	0.00000	0.00000

Table 4: Comparison of results, weight = 10.

Node Pair	Hier. FPA	DES
(1.1.3-1.3.3)	0.28784	0.25235
(1.1.2-1.2.4)	0.00470	0.00166
(1.1.6-1.3.7)	0.09745	0.09995
(1.1.1-1.2.3)	0.10185	0.11089
(1.2.1-1.2.6)	0.35567	0.36216
(1.3.1-1.3.8)	0.06322	0.05275
(1.3.5-1.3.6)	0.00000	0.000000

References

[1] F. P. Kelly, "Loss Networks," *The Annals of Applied Probability*, vol. 1, no. 3, pp. 319–378, 1991.

[2] S. Chung, A. Kashper, and K. W. Ross, "Computing Approximate Blocking Probabilities for Large Loss Networks with State-Dependent Routing," *IEEE Trans. Networking*, vol. 1, no. 1, pp. 105–115, 1993.

[3] J. S. Kaufman, "Blocking in a Shared Resource Environment," *IEEE Trans. Comm.*, vol. 29, no. 8, pp. 1474–1481, 1981.

[4] J. A. Morrison, K. G. Ramakrishnan, and D. Mitra, "Refined Asymptotic Approximation to Loss Probabilities and Their Sensitivities in Shared Unbuffered Resources," *SIAM J. Appl. Math.*, vol. 59, no. 2, pp. 494–513, 1998.

[5] F. P. Kelly, "Notes on effective bandwidth," in *Stochastic Networks Theory and Applications*. 1996, F. P. Kelly, S. Zachary and I. Ziedins, pp. 141–168, Eds. Oxford, U. K.: Science.

Table 5: Comparison of results, weight = 15.

Node Pair	Hier. FPA	DES
(1.1.3-1.3.3)	0.54327	0.52136
(1.1.2-1.2.4)	0.05856	0.04963
(1.1.6-1.3.7)	0.29617	0.30721
(1.1.1-1.2.3)	0.23567	0.24354
(1.2.1-1.2.6)	0.53049	0.54061
(1.3.1-1.3.8)	0.26342	0.25285
(1.3.5-1.3.6)	0.00000	0.00000

Table 6: Comparison of results, weight = 20.

Node Pair	Hier. FPA	DES
(1.1.3-1.3.3)	0.66527	0.66421
(1.1.2-1.2.4)	0.16374	0.16717
(1.1.6-1.3.7)	0.42122	0.43112
(1.1.1-1.2.3)	0.33676	0.34187
(1.2.1-1.2.6)	0.62975	0.62494
(1.3.1-1.3.8)	0.39753	0.39476
(1.3.5-1.3.6)	0.00000	0.00000

[6] A. Ephremides and B. Hajek, "Information Theory and Communication Network: An Unconsummated Union," *IEEE Trans. Information Theory*, vol. 44, no. 6, pp. 2416–2434, 1998.

[7] S. Chen and K. Nahrstedt, "An Overview of Quality of Service Routing for Next-Generation High-Speed Networks: Problems and Solutions," *IEEE Network*, 1998.

[8] D. Mitra, R. Gibbens, and B. D. Huang, "State-Dependent Routing on Symmetric Loss Networks with Trunk Reservations – I," *IEEE Trans. Communications*, vol. 41, no. 2, pp. 400–411, 1993.

[9] A. G. Greenberg and R. Srikant, "Computational Techniques for Accurate Performance Evaluation of Multirate, Multihop Communication Networks," *IEEE/ACM Trans. Networking*, vol. 5, no. 2, pp. 266–277, 1997.

[10] A. Orda R. Guérin and D. Williams, "QoS Routing Mechanisms and OSPF Extensions," *2nd IEEE Global Internet Mini-Conference*, November 1997, Phoenix, AZ.

[11] J. Behrens and J. J. Garcia-Luna-Aceves, "Hierarchical Routing Using Link Vectors," *Proc. INFOCOM*, 1998.

[12] M. Montgomery and G. de Veciana, "Hierarchical Source Routing Through Clouds," *Proc. INFOCOM*, 1998.

[13] Graham Louth, Michael Mitzenmacher, and Frank Kelly, "Computational Complexity of Loss Network," *Theoretical Computer Science*, vol. 125, pp. 45–59, 1994.

[14] S. Chung and K. W. Ross, "Reduced Load Approximations for Multirate Loss Networks," *IEEE Trans. Comm.*, vol. 41, no. 8, pp. 1222–1231, 1993.

[15] W. C. Lee, "Spanning tree method for link state aggregation in large communication networks," *Proc. IEEE Infocom*, vol. 1, pp. 297–302, 1995.

[16] R. Guérin and A. Orda, "QoS Routing in Networks with Inaccurate Information: Theory and Algorithms," *IEEE/ACM Trans. Networking*, vol. 7, no. 3, pp. 350–364, 1999.

[17] M. Liu and J. S. Baras, "Fixed point approximation for multirate multihop loss networks with adaptive routing," Tech. Rep., Institute for Systems Research, University of Maryland, College Park, MD, 1999, Technical Report TR99-44.

[18] M. Liu, "Network Performance Modeling, Design and Dimensioning Methodologies," Tech. Rep. PhD 2000-3, Center for Satellite and Hybrid Communication Networks, University of Maryland, College Park, 2000.