Modeling and Performance Analysis of Telephony Gateway REgistration Protocol

Kushal Kumaran and Anirudha Sahoo
Kanwal Rekhi School of Information Technology
Indian Institute of Technology, Bombay, Powai, Mumbai, India 400076.
email: {kushal, sahoo}@it.iitb.ac.in

***Abstract*—Telephony gateways are devices that interface between IP telephony networks and the PSTN. A telephony proxy or location server (LS) that attempts to connect a call for an end user may need to choose between multiple candidate gateways. Gateways need to send updates of their dynamic resource information periodically to the LS which is used by LS while choosing a gateway. Telephony Gateway REgistration Protocol (TGREP) is an IETF proposed standard used by gateways for this purpose. To the best of our knowledge, there is no study done on the performance of TGREP and we believe that this is the first paper on performance modeling of TGREP system. In this paper, we propose an analytical model for TGREP system and then use the model to evaluate different gateway selection algorithms proposed by us in this paper. We validate our model using simulation results. We also present an adaptive update mechanism in which the overhead of sending dynamic resource information is reduced without significant effect on call blocking probability. We outline a few important conclusions from this study that are quite contrary to what IP telephony providers might resort to while deploying TGREP.**

I. Introduction

Use of Voice over IP (VoIP) applications has been on the rise. Therefore, more and more VoIP service providers are setting up their networks for such services. Reachability information of different telephony destinations is required for the service providers to successfully complete the VoIP calls. Most of the current implementations of VoIP network either use static configuration or some proprietary protocol to get the reachability information. But as the size of the network increases and interoperability with other providers becomes necessary, they have to resort to some standardized dynamic protocol to get the reachability information. Telephony Routing over IP (TRIP) is an IETF standard which enables voice service providers to exchange their telephony reachability information with others [1]. When calls from an IP device is made to a regular PSTN phone, the call needs to be completed by a *telephony gateway*. The telephony gateway is responsible for conversion of signalling between the IP network and PSTN network.

In general, there might be more than one gateway that can route a given call to its destination. Hence a routing protocol is needed that can select one of these gateways based on some criteria. The appropriate choice of gateway depends on many factors, which are summarized in [2]. Some of them are described below:

This research was partially supported by IRCC, IIT Bombay under grant number 03IR059.

- **Number of candidate gateways**: Since gateways are connected to the *global switched telephone network (GSTN)*, theoretically any given gateway could connect to a given telephone number. So the number of candidate gateways can be very large.
- **Business relationships of the gateway operator**: Connecting a call to the GSTN requires resources at the gateway and there is monetary cost involved. The gateway operator may want to accept calls from certain users and deny calls from certain others.
- **Gateway capacity**: Telephony gateways have only a finite capacity. The capacity of a gateway is normally measured as the number of calls it can support simultaneously. At any given time, the available capacity may be less than the total capacity, since some of the resources may already be allocated for the ongoing calls. Hence, the routing protocol may make the choice based on total capacity and available capacity.
- **Protocol and feature compatibility**: The user may be using a signalling or media protocol that may not be supported by all gateways. Hence, the choice of gateway will take this compatibility into consideration.

Since the policies of various parties (gateway operators, service providers, end users) drive the selection of gateways, it is not possible to simply use some kind of "global directory" of gateways that the users can look up when initiating a call. Rather, information on availability and capacity of gateways must be shared among providers and, based on policy, made available to other providers and users. The TRIP protocol is used for this purpose [1]. VoIP networks have Location Servers (LS) which use TRIP to exchange gateway information with each other. The Telephony Gateway REgistration Protocol (TGREP) is a protocol used by gateways to report reachability and resource information to LSs [3].

While discussing routing algorithms in this paper, we will assume that the set of candidate gateways has been pruned down to only those gateways that satisfy all policy considerations and hence the routing decision is based on states of the gateways in terms of resources.

A. Overview of Operation

An IP telephony network is organized as a collection of Internet Telephony Administrative Domains (ITAD). An ITAD is a set of resources (gateways and location servers) administered by a single authority (the service provider). End users are customers of ITADs. Typically the service providers

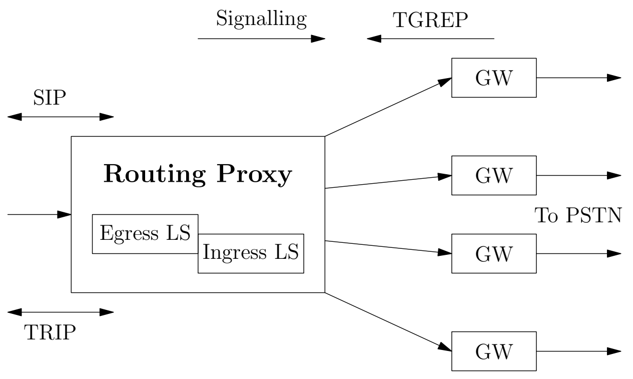


Fig. 1. Gateway and LS Configuration in a TGREP System

divide their ITAD into few Points of Presence (POP). Each POP consists of some gateways and a proxy server, as shown in Figure 1. The proxy server can be a SIP proxy or a H.323 gatekeeper, depending on the protocols that the provider supports.

The proxy server is the signalling entity responsible for routing the call to one of the gateways. The *Ingress LS* is the receiving end of TGREP connections from gateways. It receives routing information from them and passes this to the *Egress LS*, which in turn uses TRIP to disseminate it to other location servers in the network. The Ingress LS is the "TGREP Receiver" and the gateway is the "TGREP Sender".

The operation of TGREP is quite similar to that of TRIP. The TGREP Sender establishes a session with the TGREP Receiver. The TGREP sender then sends reachability information and some dynamic information about its resources such as remaining capacity to the TGREP receiver. The TGREP sender sends an update whenever there is any change in the reachability information associated with it or when it wants to send more current information about resources.

B. Connecting a Call

When an IP node wishes to make a call to a PSTN node, it contacts the local signalling server (SIP proxy or H.323 gatekeeper). The server contacts the LS in its ITAD to determine the gateway to use for completing the call. If the destination number is not in the local ITAD, then the signalling traverses multiple ITADs to the home ITAD of the called number. If multiple gateways can complete the call in this home ITAD, then the LS in the destination network has to make a routing (or gateway selection) decision. The decision might depend on various parameters, including gateway capabilities and administrative policies.

C. Gateway Updates

Gateways have finite capacity. In the initial update sent after establishing a TGREP session, they send the number of total circuits to the LS. Each ongoing call uses up one of these circuits. Hence, the gateways send periodic updates about the available circuits so that the LS has accurate information about dynamic resources. The LS makes its routing decision based on the available circuit information reported to it by the gateways. Following information are included in the updates sent by gateways:

TABLE I
SYMBOLS USED

Symbol	Definition
λ	call arrival rate (Poisson)
λ_i	IP-to-PSTN call rate
λ_o	PSTN-to-IP call rate
$1/\mu$	average call holding time
T	total capacity
p_b	blocking probability
α	update rate

Total Circuits	is the total number of PSTN circuits available on the gateway.
Available Circuits	is the number of free PSTN circuits on the gateway
Call Success Rate	provides information about the number of calls successfully completed and total number of attempted calls.

TGREP is a fairly new protocol which is currently being considered for standardization. There is an implementation of the protocol on Cisco IOS gateways [4]. To the best of our knowledge, there is no study done on TGREP to model the protocol and evaluate its performance. Also, there is no known gateway selection algorithm published in the literature. In this paper, we model the gateway capacity and then propose few gateway selection algorithms based on the model. The TGREP standard does not provide any guideline regarding the update rate of the dynamic information. This paper proposes an adaptive update rate scheme which is shown to have similar performance as the fixed update rate scheme (in terms of blocking probability), but incurs less overhead in terms of bandwidth usage. We believe this is the first paper to address these aspects of the TGREP protocol.

II. ANALYTICAL MODELING OF GATEWAY CAPACITY

Previous work in the area of IP telephony performance has focused on providing quality of service to media flows based on parameters such as delay, jitter and packet loss. Schlesener et al. measured the effect of signalling delay and trunk failures on call blocking probability [5]. But little work seems to have been done in the area of telephony routing. We will focus here on gateway selection and its effect on blocking probability. In particular, we will assume that a call can fail only if the chosen gateway does not have any circuits free, i.e., there can be no failure in connecting the call in the PSTN.

We start by abstracting all the available physical gateways as a single logical gateway having total circuits equal to the sum of the total circuits in the individual physical gateways. We present how a gateway operator can provide a service guarantee in terms of call blocking probability. Then a gateway selection algorithm should be implemented by the service provider to choose one of the multiple available gateways to actually provide that level of service.

Table I defines symbols that are used in this section. We model the logical gateway capacity as an M/M/T Markov chain. Thus, the call arrival and call departure processes

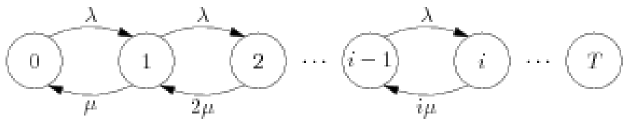


Fig. 2. M/M/T Model of Logical Gateway

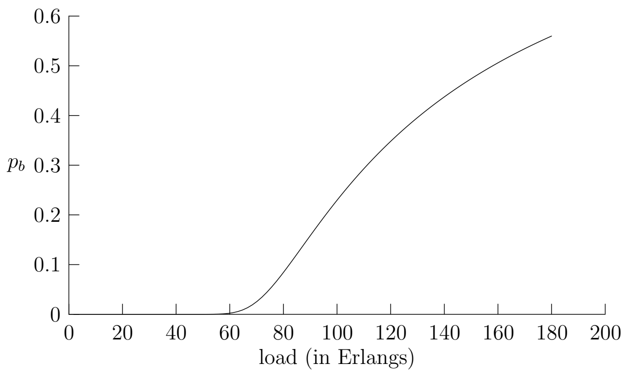


Fig. 3. Blocking Probability vs Load for Logical Gateway

are modelled as Poisson processes. The state of the model indicates the number of ongoing calls on the logical gateway. The model is shown in Figure 2.

Since there is no queueing at the gateway, the probability that an incoming call will be dropped is the probability of the logical gateway being in state T. This is the blocking probability of the gateway given by

$$p_b = \frac{(\lambda_i/\mu)^T/T!}{\sum_{j=0}^{T}(\lambda_i/\mu)^j/j!}$$

which is the well-known Erlang's B loss formula [6]. Blocking probability increases with increase in call arrival rate as shown in Figure 3 for $T = 80$. The load, expressed in Erlangs, is defined as λ/μ.

A gateway will also support calls incoming from the PSTN into the IP network. If λ_o is the rate of calls incoming from the PSTN, then the total call rate for the gateway is $\lambda = \lambda_i + \lambda_o$ and the blocking probability is given by

$$p_b = \frac{(\lambda/\mu)^T/T!}{\sum_{j=0}^{T}(\lambda/\mu)^j/j!} \tag{1}$$

VoIP service providers can provide quality of service in terms of blocking probability by incorporating admission control. The admission control in this scenario is on the basis of incoming call rate. The gateway operator can guarantee a particular call blocking probability if the call rate is less than a threshold. This threshold can be calculated by solving (1) for λ. (1) can be rewritten as (2). Hence, for a given blocking probability p_b, corresponding call rate threshold λ can be obtained by solving (2). Since a closed form solution is not possible, numerical methods have to be used to solve (2).

$$\frac{(p_b-1)(\lambda/\mu)^T}{T!} + \sum_{j=0}^{T-1}\frac{(\lambda/\mu)^j}{j!} = 0 \tag{2}$$

Note that the above analysis provides a theoretical solution for guaranteeing a blocking probability. The analysis presented here serves as the limit of the quality of service that can be achieved. When there are actually multiple gateways to choose from, then the quality of service will be lower than that of the logical gateway. Of course, if there is only one physical gateway present in a real deployment, then the above analysis can be applied directly.

III. Modeling Gateway Selection Algorithms

We now consider the actual scenario where the LS needs to choose between multiple available gateways. The gateways send updates to the LS with the total capacity and the current available capacity from which the LS can calculate the number of ongoing calls. Since LS gets synchronized with the gateways only when it receives updates, in between two successive updates LS can potentially have stale information. Hence, to analyze the performance of the routing system, we need a model of the entire routing system. Such a model can also be used to analyze how the update rate affects system performance.

The idea behind the analysis is to explicitly model the routing algorithm used by the LS with the values received in updates from the gateways. The system is modelled as a Continuous Time Markov Chain (CTMC). For this model, we consider a system with two gateways G_1 and G_2, of capacities T_1 and T_2 respectively. The state space S of the CTMC is described by four variables:

a_1	the actual number of calls ongoing at G_1
a_2	the actual number of calls ongoing at G_2
i_1	the number of ongoing calls for G_1 as per the last update
i_2	the number of ongoing calls for G_2 as per the last update

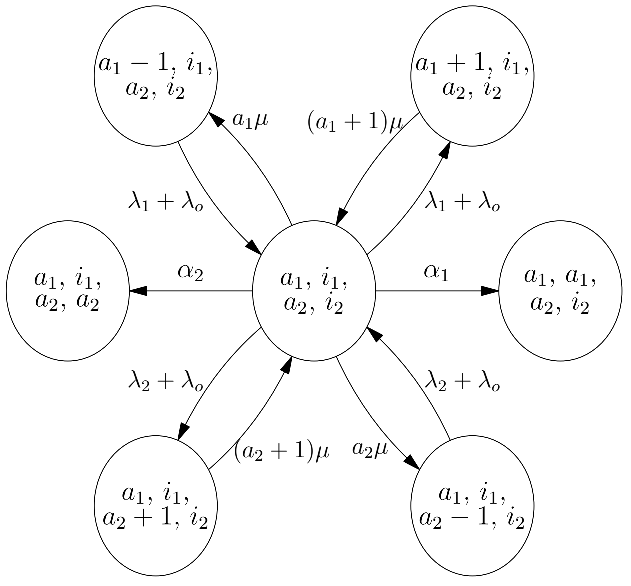
Obviously, $0 \le a_1 \le T_1$, $0 \le i_1 \le T_1$, $0 \le a_2 \le T_2$, and $0 \le i_2 \le T_2$.

The parameters of the model are:

λ_i	IP to PSTN call arrival rate
$1/\mu$	average call holding time
α_1	update rate for G_1
α_2	update rate for G_2

The general state of the model is shown in Figure 4. When the routing algorithm chooses to route a call to G_1, a_1 increases by one and when it chooses to route a call to G_2, a_2 increases by one. The rates λ_1 and λ_2 depend on the routing algorithm in use. From a state with a_1 ongoing calls at G_1 and a_2 ongoing calls at G_2, calls end with rate $a_1\mu$ and $a_2\mu$, resulting in transitions to states with their respective actual number of ongoing calls decreased by one.

States in which i_1 (i_2) is not equal to a_1 (a_2) are the states where the LS is not in sync with the gateway. From these states, the gateways may send updates with rates α_1 and α_2 respectively, resulting in i_1 (i_2) becoming equal to a_1 (a_2). To keep the model simple, we assume updates are Poisson,



$a_1-1,\, i_1,\, a_2,\, i_2$
$a_1+1,\, i_1,\, a_2,\, i_2$
$a_1\mu$
$(a_1+1)\mu$
$\lambda_1+\lambda_o$
$\lambda_1+\lambda_o$
$a_1,\, i_1,\, a_2,\, a_2$
α_2
$a_1,\, i_1,\, a_2,\, i_2$
α_1
$a_1,\, a_1,\, a_2,\, i_2$
$\lambda_2+\lambda_o$
$\lambda_2+\lambda_o$
$(a_2+1)\mu$
$a_2\mu$
$a_1,\, i_1,\, a_2+1,\, i_2$
$a_1,\, i_1,\, a_2-1,\, i_2$

Fig. 4. CTMC Model of TGREP System with Two Gateways

whereas the simulation uses periodic updates. In the following sections, we can see from the comparison between model based results and simulation based results that this assumption has minimal effect on blocking probability.

If a call is routed to a gateway when it has no circuits available, i.e., $a_1 = T_1$ or $a_2 = T_2$, then the call is blocked. The objective is to minimize the probability of being in such a state.

The approach for evaluating a routing algorithm with this model is to decide the rates λ_1 and λ_2 for that algorithm. These rates will be some function of i_1 and i_2. Then the steady-state probabilities P_i of the system being in state i can be calculated. It is easy to see that the state space of the CTMC can become quite large as the total capacity of gateways become large. In fact, a system with two gateways with capacities T_1 and T_2 is represented as a CTMC with $(T_1+1)^2(T_2+1)^2$ states. Since the number of states is quite large, it is not possible to get a closed form solution of steady state probability P_i of being in state i. Hence, we have used PRISM software tool [7] for this purpose.

To calculate blocking probability for the system, we construct a reward model for the system, with the reward r_i for state i equal to the probability that a call will be blocked when the system is in state i. Then the overall system blocking probability is given by

$$p_b = \sum_{i \in S} r_i P_i$$

where S the set of states in the CTMC.

The model can be generalized to more than two gateways. In general, for n gateways, the state will be the n-tuple of pairs (a_j, i_j), where a_j is the number of calls ongoing at gateway G_j and i_j is the value received in the last update from G_j.

IV. Gateway Selection Algorithms

Now we will consider specific gateway selection algorithms and evaluate their performance in terms of blocking probability. This evaluation will be using the analytical model and it will be verified using simulation. We first consider the scenarios with two gateways to simplify discussion and then generalize the algorithm for n number of gateways. We consider two different algorithms for gateway selection.

A. Minimum Utilization Based Selection (MUBS)

The most useful data received from the gateway updates is the Available Circuits (AC) attribute. This algorithm uses the AC values of each gateway to calculate the utilization of each gateway and select the gateway which has minimum utilization. In an ideal scenario, if an LS always has current information about the gateways, intuitively, this algorithm will produce optimal performance. This is because it always routes calls to the gateway which is least loaded until utilization of the gateway reaches the utilization of the other gateway. Thus, it will tend to balance the load across the gateways. This algorithm is similar to the classical load balancing used in web servers [8]. However, in our case, the resource information at the LS becomes stale between two successive updates. Hence, this algorithm does not perform that well as demonstrated below.

For this algorithm, the rates λ_1 and λ_2 in the CTMC model (given in Figure 4) are

$$\lambda_1 = \begin{cases} 0 & \text{if } i_1/T_1 > i_2/T_2, \\ \lambda_i & \text{if } i_1/T_1 < i_2/T_2, \\ \lambda_i/2 & \text{otherwise.} \end{cases}$$

and

$$\lambda_2 = \begin{cases} \lambda_i & \text{if } i_1/T_1 > i_2/T_2, \\ 0 & \text{if } i_1/T_1 < i_2/T_2, \\ \lambda_i/2 & \text{otherwise.} \end{cases}$$

Thus, if a gateway has utilization strictly less than the other, then the algorithm will send all the calls to that gateway. But if the utilization of the two gateways are equal, then the calls are equally split between the two gateways.

The rewards assigned to the states to calculate blocking probability are

$$r = \begin{cases} 1 & \text{if } a_1 = T_1 \text{ and } a_2 = T_2 \\ 1 & \text{if } a_1 = T_1 \text{ and } i_1/T_1 < i_2/T_2 \\ 1 & \text{if } a_2 = T_2 \text{ and } i_2/T_2 < i_1/T_1 \\ 0.5 & \text{if } a_1 = T_1, a_2 \neq T_2 \text{ and } i_1/T_1 = i_2/T_2 \\ 0.5 & \text{if } a_2 = T_2, a_1 \neq T_1 \text{ and } i_1/T_1 = i_2/T_2 \\ 0 & \text{otherwise} \end{cases}$$

In general, for a set G of n gateways, we first consider the states where a single gateway j has the minimum utilization, i.e., $i_j/T_j < i_k/T_k$ for all $k \in G$ and $k \neq j$. Then the rate for the k^{th} gateway is

$$\lambda_k = \begin{cases} \lambda_i & \text{if } k = j \\ 0 & \text{otherwise} \end{cases}$$

and the reward assignment is

$$r = \begin{cases} 1 & \text{if } a_j = T_j \\ 0 & \text{otherwise} \end{cases}$$

TABLE II
PARAMETERS FOR DIFFERENT SCENARIOS

Parameter	Value for	
	equal	*skewed*
T_1	40	20
T_2	40	60
update interval	30 sec	30 sec

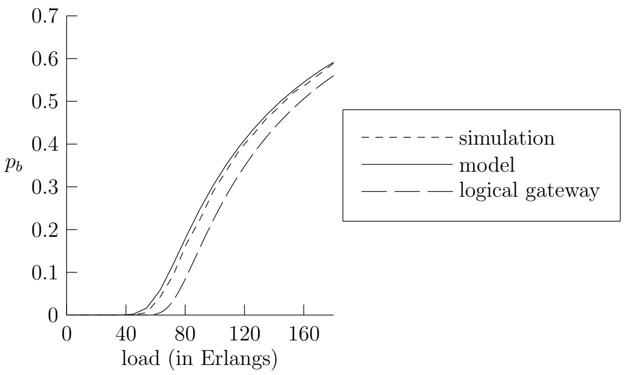


Fig. 5. Blocking Probability vs. Load using MUBS Algorithm (Gateways having Equal Capacity)

If a set M of m gateways have the same utilization i/T and $i/T < i_k/T_k$ for all $k \in G$ and $k \notin M$, then the incoming calls are split equally between the m gateways.

$$\lambda_j = \begin{cases} \lambda_i/m & \text{if } j \in M \\ 0 & \text{otherwise} \end{cases}$$

If k out of the m gateways have actual number of ongoing calls equal to the total capacity, then the reward assigned to that state is k/m and reward 0 is assigned to the other states.

1) Experimental Results: We have run experiments using the CTMC model given in Figure 4 (using PRISM) to solve the steady state probabilities with two gateways. The parameters used for these scenarios are summarized in Table II. Gateway capacities used are small for quick computation of blocking probability using the Markov chain. We also have developed a TGREP simulator and implemented the gateway selection algorithms in it. We provide the results based on the model as well as using the simulator. All simulation results are presented with a 95% confidence interval of less than 0.01. Finally, we also run the same experiment on logical gateway, which has capacity equal to the sum of capacities of the two individual gateways.

Two scenarios are considered here for the experimental results:

equal	with the two gateways having equal capacities.
skewed	with one of the gateways having much higher capacity than the other.

As is clear from Figures 5 and 6, the blocking probability when using the CTMC model closely follows the blocking probability obtained from the simulator as the load is varied on the system. This validates that our CTMC model is correct.

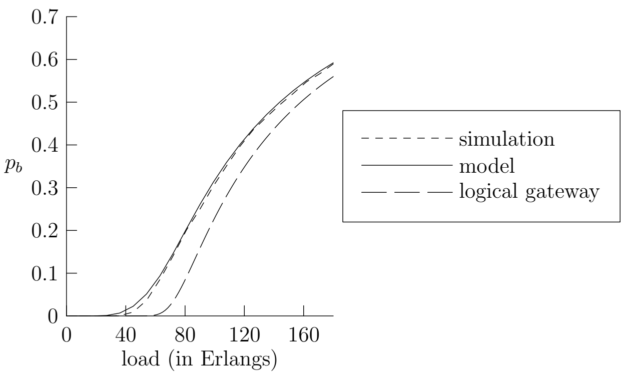


Fig. 6. Blocking Probability vs. Load using MUBS Algorithm (Gateways having Skewed Capacity)

Also, as expected, the performance of the system with one logical gateway is always better in terms of blocking probability in both the scenarios.

B. Probabilistic Selection Based on Utilization (PSBU)

A problem with minimum utilization based selection is that between two updates, it will always pick the same gateway. Thus, if the update rate is low, that particular gateway can quickly reach its capacity and then it will start dropping calls. This will adversely affect the performance of the system. Hence, a better way to select a gateway is to assign probabilities to each gateway based on their current state in terms of capacity and select gateways based on those probabilities.

There are two ways that the probability can be assigned. The first method is to have the probability of choosing a gateway be inversely proportional to the utilization of the gateway. In this case,

$$\lambda_1 = \frac{\lambda_i(i_2/T_2)}{i_1/T_1 + i_2/T_2}$$
$$\lambda_2 = \frac{\lambda_i(i_1/T_1)}{i_1/T_1 + i_2/T_2}$$

The second method is to have the probability directly proportional to the remaining capacities of the gateways, i.e.,

$$\lambda_1 = \frac{\lambda_i(1 - i_1/T_1)}{2 - i_1/T_1 - i_2/T_2}$$
$$\lambda_2 = \frac{\lambda_i(1 - i_2/T_2)}{2 - i_1/T_1 - i_2/T_2}$$

Evaluation of the two methods using the CTMC model and simulation suggests that the second method results in more balanced load and lower blocking probability. As an example, consider a case where two gateways have utilizations 0.7 and 0.9 respectively. The values of λ_1 and λ_2 as calculated the two methods are given in Table III. The first method results in almost the same rate for both gateways, leading to unbalanced load. The second method assigns much lower probability for the more utilized gateway, which leads to lower system blocking probability.

Hence, we use the second method for the probabilistic gateway selection algorithm and disregard the first method. We finesse it a little bit to handle the case when the denominator

TABLE III
AN EXAMPLE SHOWING THE TWO METHODS OF THE PROBABILISTIC SELECTION ALGORITHM

	Gateway 1	Gateway 2
utilizations	0.7	0.9
First method (probabilities in inverse proportion of utilization)	0.5625	0.4375
Second method (probabilities in direct proportion of remaining capacity)	0.75	0.25

becomes zero, which happens if both gateways are fully utilized. With this change, the rates λ_1 and λ_2 are given by

$$\lambda_1 = \begin{cases} \frac{\lambda_i}{2} & \text{if } i_1 = T_1 \text{ and } i_2 = T_2 \\ \frac{\lambda_i(1-i_1/T_1)}{2-i_1/T_1-i_2/T_2} & \text{otherwise} \end{cases}$$

and

$$\lambda_2 = \begin{cases} \frac{\lambda_i}{2} & \text{if } i_1 = T_1 \text{ and } i_2 = T_2 \\ \frac{\lambda_i(1-i_2/T_2)}{2-i_1/T_1-i_2/T_2} & \text{otherwise} \end{cases}$$

Reward assignment for the states is

$$r = \begin{cases} 1 & \text{if } a_1 = T_1 \text{ and } a_2 = T_2 \\ \frac{1-i_1/T_1}{2-i_1/T_1-i_2/T_2} & \text{if } a_1 = T_1 \text{ and } (i_1 \neq T_1 \text{ or } i_2 \neq T_2) \\ \frac{1-i_2/T_2}{2-i_1/T_1-i_2/T_2} & \text{if } a_2 = T_2 \text{ and } (i_1 \neq T_1 \text{ or } i_2 \neq T_2) \\ 0.5 & \text{if } i_1 = T_1 \text{ and } i_2 = T_2 \text{ and } a_1 = T_1 \\ 0.5 & \text{if } i_1 = T_1 \text{ and } i_2 = T_2 \text{ and } a_2 = T_2 \\ 0 & \text{otherwise} \end{cases}$$

The above analysis can be generalized to n gateways. Then the call rate is given by

$$\lambda_j = \begin{cases} \lambda_i/n & \text{if } i_1/T_1 = i_2/T_2 = \cdots = i_n/T_n = 1 \\ \frac{\lambda_i\left(1-i_j/T_j\right)}{n-\sum_{k=1}^{n} i_k/T_k} & \text{otherwise} \end{cases}$$

The reward assignment for a state in which $i_1/T_1 = i_2/T_2 = \cdots = i_n/T_n = 1$ is k/n where k is the number of gateways for which $a = T$. For other states, the rewards are assigned as follows. Let B, $B \neq \phi$, be the set of k gateways for which actual number of ongoing calls is equal to the total capacity. Then the reward assigned is

$$\frac{k - \sum_{g \in B} i_g/T_g}{n - \sum_{k=1}^{n} i_k/T_k}$$

When none of the gateway has reached its capacity in terms of the actual number of ongoing calls (i.e. when $B = \phi$), the reward assigned is 0.

1) Experimental Results: We again ran experiments using the CTMC model and also using our simulator. The blocking probabilities vs. load for this algorithm are shown in Figures 7 and 8 using the parameters in Table II. Here again we see that results using our simulator very closely follow the CTMC model, again validating our model. Comparing these graphs with those of minimum utilization based algorithm (i.e., comparing Figure 7 with Figure 5 and Figure 8 with Figure 6), it can easily be seen that the PSBU performs better than the MUBS algorithm in terms of blocking probability.

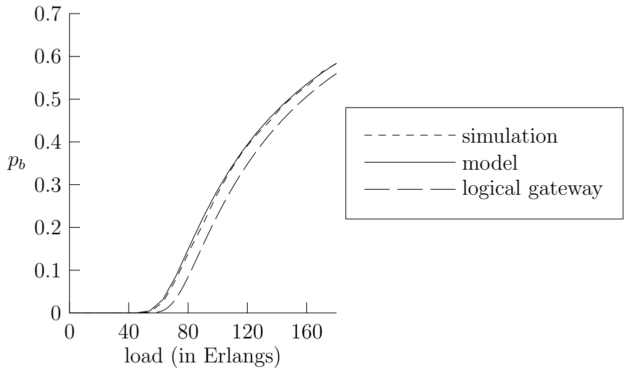


Fig. 7. Blocking Probability vs. Load using PSBU Algorithm (Two Gateways with Equal Capacity)

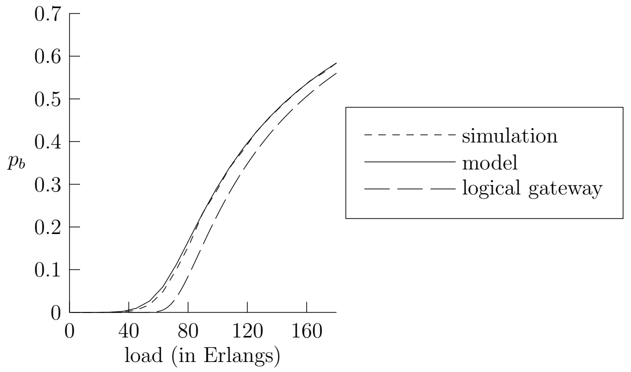


Fig. 8. Blocking Probability vs. Load using PSBU Algorithm (Two Gateways with Skewed Capacity)

We ran some experiments with different number of gateways keeping the total capacity of the system constant. Figures 9 and 10 show the effect of increasing the number of gateways on blocking probability (when the system capacity is constant). The parameters for these simulation runs are summarized in Table IV. The difference in blocking probabilities between the two algorithms is much higher here because the gateway capacities are higher. Thus, relative performance of MUBS becomes worse than PSBU as the total capacity of the system increases.

Note that the two algorithms have been plotted to different scales. This is because the blocking probability attained by the MUBS algorithm is an order of magnitude worse than that attained by the PSBU algorithm.

V. EFFECTS OF UPDATE RATE

Another parameter that we have not discussed so far is the update rate. Intuitively, decreasing update rate results in the information at the LS getting increasingly stale, which should result in increase of blocking probability. The problem that a gateway provider might face here is how to choose the proper update rate to be able to provide the appropriate quality of service in terms of blocking probability.

TABLE IV
PARAMETERS FOR SIMULATION WITH MORE GATEWAYS

Parameter	Value
Total circuits	1150
Update interval	30 sec

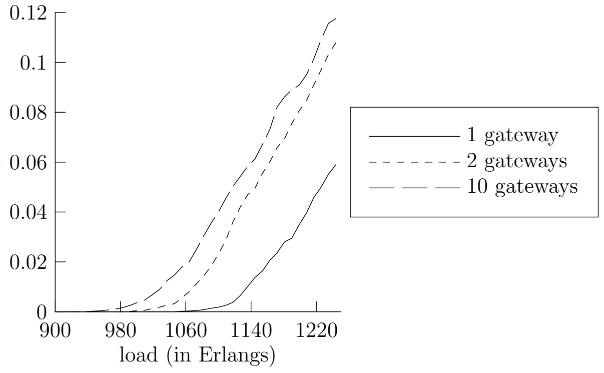


Fig. 9. Blocking Probability vs Load with Different Number of Gateways Using PSBU Algorithm

We use the CTMC model presented in Section III to see the variation of blocking probability with update rate. The parameters in Table II are used in all further model evaluations regarding update rate. Figure 11 shows the variation of blocking probability as update rate is increased. The graph shown is for a load of 64.8 Erlangs (the value of load for which a logical gateway would give a blocking probability of 0.01) using the PSBU algorithm.

Clearly, increasing update rate decreases blocking probability, although not indefinitely. Theoretically, the update rate could be increased to infinity, in which case the LS is always in sync with the gateway. This results in the minimum achievable blocking probability. The value of the minimum achievable blocking probability is the one given by (1).

VI. Adaptive Update Rates

The final problem we address is to reduce the amount of data being sent from gateways to the LS in the update messages. We notice that if only some of the gateways are full, calls will not be dropped as long as the LS has accurate information about the number of ongoing calls on those gateways. Intuitively, this means that the LS does not need as many updates when fewer calls are ongoing in a gateway. This observation leads to an adaptive update rate where update rate is increased as a gateway approaches its maximum capacity. We have evaluated a system with total capacity T where a gateway with a ongoing calls has update rate of $a/T \times \alpha$, where α is the maximum update rate for that gateway.

The results, with the probabilistic gateway selection algorithm PSBU, are compared in Figures 12 and 13. It can be seen that the blocking probability of the constant update rate is still obtainable with fewer updates being sent using the adaptive update scheme without affecting the blocking probability.

Let the size of a TGREP UPDATE message be U bytes long. With constant update rate, the average overhead bandwidth

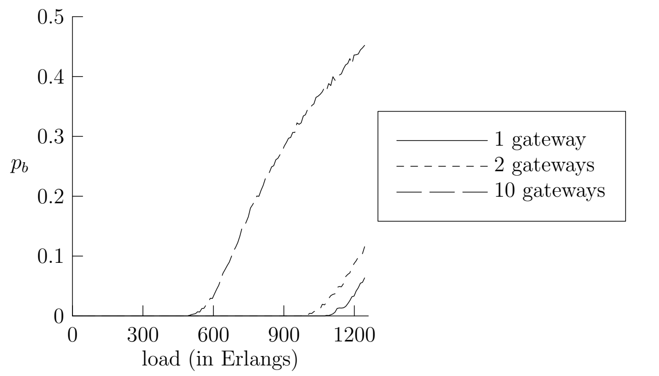


Fig. 10. Blocking Probability vs Load with Different Number of Gateways Using MUBS Algorithm

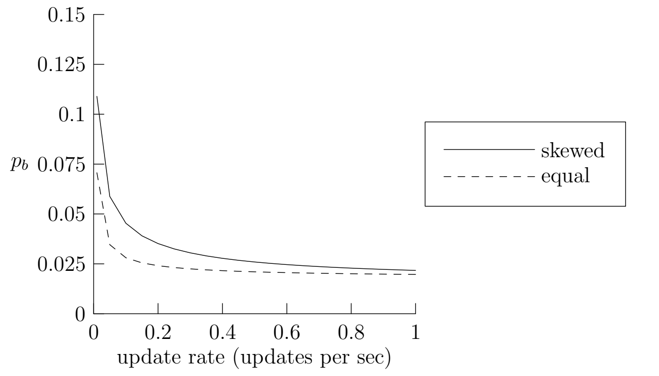


Fig. 11. Blocking Probability vs Update Rate

(to send UPDATE messages) in any state is $U\alpha$ bytes/sec, where α is the update rate. If adaptive update rate is used, the overhead bandwidth in a state with a_1 ongoing calls at G_1 and a_2 ongoing calls at G_2 is $U\alpha(a_1/T_1+a_2/T_2)$. To calculate the average overhead bandwidth, we use a reward model similar to the one described in Section III for blocking probability. The overhead bandwidth reward b_i for a state is the overhead bandwidth in that state. Then the average overhead bandwidth is calculated as

$$b_{avg} = \sum_{i\in S} b_i P_i \tag{3}$$

When adaptive update scheme is configured for a gateway, while sending an update, the gateway calculates the next update time as $\frac{1}{\frac{a}{T}\alpha}$.

The improvement in terms of average overhead bandwidth when using adaptive update over fixed update rate is shown in Figures 14 and 15 for $U = 11$.[1] Thus, adaptive update rates can be effectively used to reduce the amount of data being sent, without adversely affecting the blocking probability.

VII. Conclusion

We presented a model of a theoretical logical TGREP gateway which represents the best performance achievable by a TGREP system in terms of call blocking probability. Then we proposed a model for gateway selection algorithm using

[1]According to [3], a TGREP UPDATE message with a single Available-Circuits attribute is at least 11 bytes long.

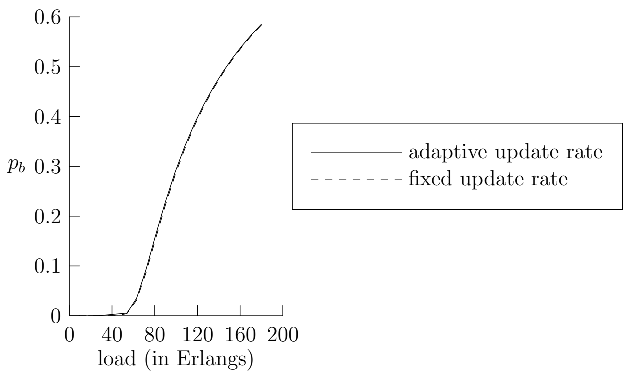


Fig. 12. Blocking Probability vs. Load Using Adaptive Update Rate (Gateways with Equal Capacity)

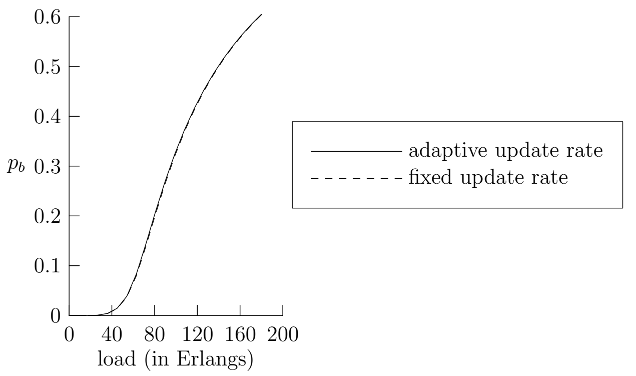


Fig. 13. Blocking Probability vs. Load Using Adaptive Update Rate (Gateways with Skewed Capacity)

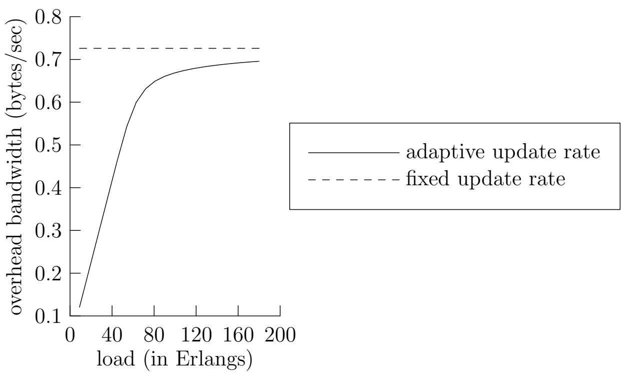


Fig. 14. Overhead Bandwidth with Adaptive Update Rate (Gateways with Equal Capacity)

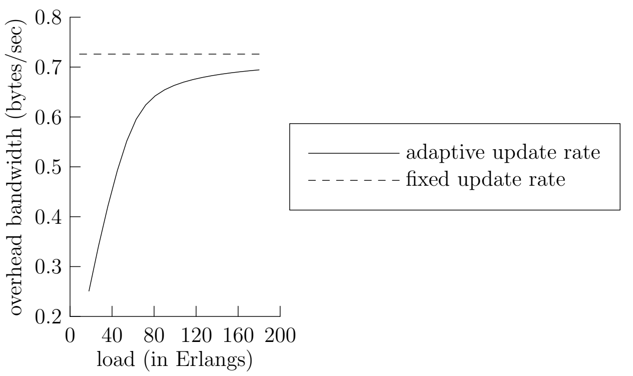


Fig. 15. Overhead Bandwidth with Adaptive Update Rate (Gateways with Skewed Capacity)

CTMC. The model was validated by our simulation results. Finally, we presented an adaptive update mechanism which reduces the update overhead without significantly affecting the call blocking probability. There are few important lessons learnt from this study which are quite contrary to what most IP telephony providers might resort to.

- For a given total capacity of a TGREP system, it is better to have less number of large gateways than to have more number of small gateways. This is because, as shown in this paper, one logical gateway (having same total capacity as the sum of capacity of individual gateways) gives the best performance in terms of call blocking probability.
- Intuitively, MUBS looks like a good gateway selection algorithm. But that is only true, if the LS is always in sync with the gateways. However, MUBS performance becomes bad when LS can have stale information about available capacity of the gateways. Hence, it is better to use PSBU, which gives better performance, in practical systems in which the LS has stale information between two successive updates. Moreover, performance of MUBS relative to PSBU becomes worse as the total capacity of the TGREP system increases.
- To reduce overhead of bandwidth consumed in sending updates, adaptive updates can be used without having any significant impact on the performance in terms of call blocking probability.

References

[1] J. Rosenberg, H. Salama, and M. Squire, “Telephony Routing over IP (TRIP),” RFC 3219 (Proposed Standard), Jan. 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3219.txt

[2] J. Rosenberg and H. Schulzrinne, “A Framework for Telephony Routing over IP,” RFC 2871 (Informational), Jun. 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2871.txt

[3] M. Bangalore, R. Kumar, H. Salama, J. Rosenberg, and D. Shah, “A Telephone Gateway REgistration Protocol (TGREP),” IETF IPTEL WG Draft, Jan. 2007. [Online]. Available: http://tools.ietf.org/id/draft-ietf-iptel-tgrep-08.txt

[4] “Telephony Gateway Registration Protocol on Cisco IOS Gateways,” 2003. [Online]. Available: http://www.cisco.com/univercd/cc/td/doc/product/software/ios123/123cgcr/vvfax_c/callc_c/triplite.htm

[5] M. C. Schlesener and V. S. Frost, “Performance evaluation of telephony routing over IP (TRIP),” in *3rd IEEE Workshop on IP Operations and Management, 2003*, 2003, pp. 47 – 53.

[6] K. S. Trivedi, *Probability and statistics with reliability, queuing and computer science applications*. Chichester, UK, UK: John Wiley and Sons Ltd., 2002.

[7] M. Z. Kwiatkowska, G. Norman, and D. Parker, “PRISM: Probabilistic symbolic model checker,” in *Computer Performance Evaluation / TOOLS*, 2002, pp. 200–204. [Online]. Available: http://citeseer.ist.psu.edu/article/kwiatkowska02prism.html

[8] V. Cardellini, M. Colajanni, and P. S. Yu, “Dynamic load balancing on web-server systems,” *IEEE Internet Computing*, pp. 28 – 39, 1999.