

Optimal Server Placement for Streaming Content Delivery Network on the Internet

Xiaojun Hei and Danny H.K. Tsang

Department of Electronic and Computer Engineering

Hong Kong University of Science and Technology

Clear Water Bay, Kowloon, Hong Kong

{heixj, eetsang}@ece.ust.hk

Brahim Bensaou

Department of Computer Science

Hongkong University of Science and Technology

Clear Water Bay, Kowloon, Hong Kong

brahim@cs.ust.hk

Abstract

Content Delivery Networks (CDN) are designed to efficiently deliver media content from content providers to a large community of geographically distributed users. The media type varies from the Web content to audio and video streaming. CDN providers are interested in placing CDN servers at suitable locations on the Internet to serve their users with the minimum cost. In this paper, we formulated the CDN server placement problem on the public Internet as a mixed integer programming model. The numerical results show that the model can be solved within tens of seconds in an AS-level topology of about 1000 ASs using a greedy decomposition method. In addition, simulation-based performance evaluation for various placement models in simulated Internet AS-level topologies shows that our model outperforms the random model and the K-median model. Sensitivity analysis is conducted empirically to study the impact of the deviation of user bandwidth demand and the link available bandwidth upon the system performance.

Index Terms

Server Placement, Streaming, Content Delivery Network

I. INTRODUCTION

Content Delivery Networks (CDN) have recently become a popular form of content distribution on the Internet. Originally, CDN were mainly targeted at the distribution of the Web content; however, nowadays they are also used extensively for the distribution of streaming audio and video content. CDN servers replicate content and are placed close to end-users to reduce the response time for user requests. User requests are usually redirected to an appropriate server with a copy of the desired content. In order to serve more end users, servers should be placed strategically on the Internet.

Today's Internet consists of thousands of administrative domains, also referred to as Autonomous Systems (ASs). ASs are interconnected via dedicated links and/or public network access points. CDN providers typically place their servers in the Internet hosting centers inside an AS and purchase access bandwidth to the Internet. Normally owned by one Internet Service

Provider (ISP), an AS can be carefully engineered and operated so that little or no internal congestion occurs. If one server is placed in one AS with sufficient access bandwidth for the users in the AS, these users can be served with sufficient bandwidth. However, it is not economic for the CDN provider to place one server in each AS.

In [1] Ratnasamy et al. argued that many Internet applications do not require *exact* topological information in order to achieve significant practical benefits. Though it is not feasible to obtain the router-level Internet topology due to the dynamic nature of the Internet, the AS-level topology is relatively easy to acquire using BGP routing tables [2]. In [3] Barford et al. reported that user demand clustering helps to generate even smaller scale AS-level topology without much accuracy loss. Note that users are not uniformly distributed in the ASs. The CDN provider can estimate its user distribution in this AS-level Internet map for an optimal server placement. If a server has already been placed in an AS, clients in other ASs can also be served by this server by utilizing the available bandwidth between ASs. Though the available bandwidth information is difficult to obtain, a number of research effort has been devoted to its acquisition. If a CDN provider cannot obtain the bandwidth utilization information from ISPs, some measurement techniques ([4], [5]) can be applied to obtain such an AS-level map with available bandwidth as the link capacity between ASs.

Quite a lot of research has been conducted on the server placement problem during the past several years. [6], [7], [8], and [9] focused on Web application; delay performance is the major concern in their models. For streaming CDNs, the bandwidth guarantee is more crucial than latency. As long as the minimum throughput to deliver streaming contents to end users can be guaranteed, the delay and delay jitter can be bounded due to the pre-buffering mechanism in streaming media players. Furthermore, in streaming applications some startup delay is tolerable. The integration of both delay and bandwidth as constraints complicates the mathematical model too much and hence it is not tractable. In this paper, we developed streaming server placement models with only the bandwidth constraint.

The communication channels between the original streaming servers, CDN servers and end-users are on the public Internet and, hence, do not incur any cost to CDN providers; however, the operation and maintenance of these distributed servers and the Internet access bandwidth required at each CDN server's network interface represent a major cost for CDN providers. There is an additional cost to keep the servers' content consistent when the content of the original server changes. A larger number of CDN servers results in a higher consistency cost. Therefore, we proposed models to determine the minimum number of servers needed to serve the user's demands in relation to the maximal usage of the bandwidth on the public Internet. Further, because the content can be pre-distributed into different CDN servers from the original server when the network is not busy, we assumed that it would cost nothing to keep content consistent in our model. With this assumption, the CDN servers can be treated as the same as the original server itself.

The rest of the paper is organized as follows. First, in Section II we present our mixed integer programming (MIP) model together with two other placement models for comparison and we propose a greedy decomposition method which, because the model is split into two sub-problems, can achieve a more efficient solution. We outline the simulation model for performance evaluation in Section III. Then, various AS-level network topologies used in performance evaluation are introduced and the parameter settings are explained in details in Section IV-A. In Section IV we evaluate the effectiveness of our greedy

decomposition method using numerical examples. Furthermore, we evaluate the performance using simulations conducted on various placement models in various simulated Internet AS-level topologies. In addition, a sensitivity analysis, conducted empirically to study impact of the deviation of the user bandwidth demand and the available bandwidth upon the system performance, is discussed. Finally, the concluding remarks are presented in Section V.

II. SERVER PLACEMENT MODELS

A. Problem Formulation

The streaming CDN server placement problem can be formulated as an MIP model. The streaming server placement involves multiple related decisions: (i) how many servers should be placed, (ii) where these servers should be, (iii) how much access bandwidth should be purchased and (iv) how user requests should be redirected to an appropriate server. With the assumption that the CDN provider relies on the public Internet to deliver the content to the end users, the only cost to construct a streaming CDN is the server setup cost. This includes the server cost and the network access bandwidth cost. Therefore, the primary goal when constructing the placement model is to minimize the setup cost. Furthermore, the minimization of traffic flows on the network is a secondary objective. To deliver a unit bandwidth from the server to the client, a longer route consumes more network bandwidth. Besides, this connection has a higher chance of performance degradation due to network congestion on each link along the path. In addition, from the ISP's point of view, the flow minimization is also preferable for a more efficient network utilization. In summary, minimization of the setup cost and the total traffic flow are two objectives with different priorities to dimension streaming CDNs. The server setup cost results in a real investment while the aim of minimizing traffic flows is to achieve a better Quality-of-Service (QoS) performance.

Let the graph $G = (V, E)$ be the network under study, where V is the set of nodes in the network, and $E \subset V \times V$ is the set of links. Note that each node in G refers to one AS. Let r_i be the average bandwidth demand from the users in AS $i \in V$. Due to various reasons, such as the policy of the network service provider, the server may not be able to be placed wherever necessary. We assume that the possible server location set in G is noted as L ($L \subseteq V$). For AS $k \in L$, a CDN provider can place server k in the Internet hosting center of AS k . We also associate b_{ij} with each link $(i, j) \in E$, where b_{ij} is the effective available bandwidth which can be utilized to deliver the flow from AS i to AS j .

We introduce decision variables x_i defined as selection variables: if a server is placed in AS i , $x_i = 1$ and $x_i = 0$, otherwise. We also define decision variables f_{ij} as the traffic flow from AS i to AS j ($f_{ii} = 0$), and z_k as the purchased Internet access bandwidth for server k . Due to practical considerations, z_k is bounded by physical constraints, such as the maximum throughput of the network interface, denoted here by a_k . Let d_i^0 be the cost of server i itself and the cost function for the access bandwidth is assumed to be the linear function $d_i * z_i$, where d_i is the unit access bandwidth cost. Therefore, the total setup cost can be defined as $\sum_{i=1}^{|L|} (d_i^0 * x_i + d_i * z_i)$. With these above notations, we have the following prioritized mixed integer programming model, noted as SR-Optimal, where the AS-level routing is assumed to follow source routing. The secondary objective of the model is to minimize the total flow in the network $\sum_{\{(i,j) \in E\}} f_{ij}$, which requires the cooperation between ASs.

$$\min \sum_{i=1}^{|L|} (d_i^0 * x_i + d_i * z_i) \quad (1)$$

$$\min \sum_{\{(i,j) \in E\}} f_{ij} \quad (2)$$

subject to

$$\forall k \in L, z_k + \sum_{\{i:(i,k) \in E\}} f_{ik} - \sum_{\{j:(k,j) \in E\}} f_{kj} = r_k \quad (3)$$

$$\forall k \in V - L, \sum_{\{i:(i,k) \in E\}} f_{ik} - \sum_{\{j:(k,j) \in E\}} f_{kj} = r_k \quad (4)$$

$$\forall k \in S, 0 \leq z_k \leq a_k * x_k \quad (5)$$

$$\forall (i, j) \in E, 0 \leq f_{ij} \leq b_{ij} \quad (6)$$

where

$$x_i = \begin{cases} 1, & \text{if a server is placed on node } i \in L \\ 0, & \text{otherwise} \end{cases}$$

Eqn. 1 and Eqn. 2 are the two objective functions of SR-Optimal, the goals of which are to minimize the total setup cost and traffic flow with the priority given to the cost. Basically, Eqn. 3 and Eqn. 4 exploit the flow conservation constraints that incoming flows equal to outgoing flows. Eqn. 5 indicates that if no server is placed on AS k , z_k is constrained to be 0; if there is a server placed on AS k , $x_k = 1$, z_k is constrained by a_k . Eqn. 6 indicates that traffic flow f_{ij} on the link $(i, j) \in E$ is bounded by the effective capacity b_{ij} . For simplicity, in the later discussion, we used the simple effective available bandwidth based on the mean value.

With the priority structure, the optimal solution of SR-Optimal can be obtained by solving the combined objective function with the introduction of a very large M shown as in Eqn. 7. SR-Optimal can be considered as a capacitated facility layout model which is known to be NP-hard [10]. However, with the prioritization structure of the problem, the SR-Optimal model can be decomposed into two sub-problems, SR-Placement and SR-Routing. The aim of SR-Placement is to find the minimum number of servers and to decide where to place these servers with a minimum setup cost, while the aim of SR-Routing is to decide how to redirect user requests to an appropriate server with minimum traffic flows. SR-Placement is obtained from SR-Optimal by ignoring temporarily the objective function Eqn. 2. The optimal solution of SR-Placement provides the minimum number of servers as well as the their locations using the values of x_i and the purchased access bandwidth using z_i . Suppose that the server locations are fixed according to SR-Placement; then, the server selection variables x_i become the known parameters of SR-Routing. Therefore, the objective function of SR-Routing is revised as Eqn. 8. It can be seen that SR-Routing is a conventional minimum cost network flow model and there exists polynomial time algorithms for solving it [11].

$$\min M * \sum_{i=1}^{|L|} (d_i^0 * x_i + d_i * z_i) + \sum_{\{(i,j) \in E\}} f_{ij} \quad (7)$$

$$\min M * \sum_{i=1}^{|L|} d_i * z_i + \sum_{\{(i,j) \in E\}} f_{ij} \quad (8)$$

There might exist multiple server location sets which lead to the same minimum number of servers. If the server locations obtained from SR-Placement are fixed as the input parameters of SR-Routing, there might be even smaller flow distribution with another set of server locations. Therefore, it can be seen that using the above decomposition approach is a greedy method of solving SR-Optimal. This is denoted as SR-Greedy.

For a general set of cost per access bandwidth $\{d_i\}$, the total cost $\sum_{i=1}^{|L|} (d_i^0 * x_i + d_i * z_i)$ obtained in SR-Optimal and SR-Greedy, are different. We further studied a special case that d_i is a constant. This indicates that we assume that the per unit access bandwidth cost is the same in different ASs. With this assumption, $\sum_{i=1}^{|L|} d_i * z_i = d * \sum_{i=1}^{|L|} z_i$. Note that $\sum_{i=1}^{|L|} z_i = \sum_{i=1}^{|V|} r_i = \text{Constant}$ based on the flow conservation property. Therefore, the term $\sum_{i=1}^{|L|} d_i * z_i$ can be removed from Eqn. 7 and Eqn. 8 without changing the optimal solution. The above two objective functions can be further simplified in Eqn. 9 and Eqn. 10.

$$\min M * \sum_{i=1}^{|L|} (d_i^0 * x_i) + \sum_{\{(i,j) \in E\}} f_{ij} \quad (9)$$

$$\min \sum_{\{(i,j) \in E\}} f_{ij} \quad (10)$$

The method of SR-Greedy is to find the optimal solution of SR-Optimal by minimize total server cost $\sum_{i=1}^{|L|} (d_i^0 * x_i)$ first and then minimize the total delivery flow $\sum_{\{(i,j) \in E\}} f_{ij}$. The total delivery flow in SR-Greedy is sub-optimal compared to that in SR-Optimal because the flow minimization is conducted after the server cost is minimized. However, the server cost $\sum_{i=1}^{|L|} (d_i^0 * x_i)$ in SR-Greedy and SR-Optimal is the same. This can be illustrated by contradiction as follows. Without loss of generality, d_i^0 ($i = 1, \dots, |V|$) are assumed to be integers for simplicity¹, and $M > \sum_{\{(i,j) \in E\}} b_{ij}$. Suppose the optimal solution of SR-Greedy and SR-Optimal are $\{x'_i, f'_{ij}\}$ and $\{x_i^*, f_{ij}^*\}$; however, $\sum_{i=1}^{|L|} (d_i^0 * x'_i) \neq \sum_{i=1}^{|L|} (d_i^0 * x_i^*)$.

By the optimality of SR-Greedy,

$$\sum_{i=1}^{|L|} (d_i^0 * x'_i) \leq \sum_{i=1}^{|L|} (d_i^0 * x_i).$$

Therefore,

$$\sum_{i=1}^{|L|} (d_i^0 * x'_i) \leq \sum_{i=1}^{|L|} (d_i^0 * x_i^*). \quad (11)$$

¹This integer assumption of d_i^0 can be extended to the case of positive real number similarly, such that the server cost in SR-Greedy and SR-Optimal can still be the same by selecting an appropriate large M.

Using the assumption $\sum_{i=1}^{|L|} (d_i^0 * x'_i) \neq \sum_{i=1}^{|L|} (d_i^0 * x_i^*)$ and Eqn. 11, $\sum_{i=1}^{|L|} (d_i^0 * x'_i) < \sum_{i=1}^{|L|} (d_i^0 * x_i^*)$.
Because d_i^0 ($i = 1, \dots, |V|$), x'_i and x_i^* are integers,

$$\sum_{i=1}^{|L|} (d_i^0 * x'_i) \leq \sum_{i=1}^{|L|} (d_i^0 * x_i^*) - 1. \quad (12)$$

By the optimality of SR-Optimal,

$$M * \sum_{i=1}^{|L|} (d_i^0 * x_i^*) + \sum_{\{(i,j) \in E\}} f_{ij}^* \leq M * \sum_{i=1}^{|L|} (d_i^0 * x_i) + \sum_{\{(i,j) \in E\}} f_{ij}.$$

Therefore,

$$M * \sum_{i=1}^{|L|} (d_i^0 * x_i^*) + \sum_{\{(i,j) \in E\}} f_{ij}^* \leq M * \sum_{i=1}^{|L|} (d_i^0 * x'_i) + \sum_{\{(i,j) \in E\}} f'_{ij}. \quad (13)$$

Using Eqn. 12 and Eqn. 13

$$M * \left(\sum_{i=1}^{|L|} (d_i^0 * x'_i) + 1 \right) + \sum_{\{(i,j) \in E\}} f_{ij}^* \leq M * \sum_{i=1}^{|L|} (d_i^0 * x'_i) + \sum_{\{(i,j) \in E\}} f'_{ij}.$$

$$M + \sum_{\{(i,j) \in E\}} f_{ij}^* \leq \sum_{\{(i,j) \in E\}} f'_{ij}. \quad (14)$$

Because $0 \leq \sum_{\{(i,j) \in E\}} f_{ij} \leq \sum_{\{(i,j) \in E\}} b_{ij} < M$, Eqn. 14 does not hold. Therefore, $\sum_{i=1}^{|L|} (d_i^0 * x'_i) = \sum_{i=1}^{|L|} (d_i^0 * x_i^*)$. \square

B. Other Models for Comparison

1) *Random K Server Placement*: In this method, K servers are chosen at random with uniform probability from all the nodes in the topology. This is considered as an ‘‘upper bound’’ placement method in a sense that an efficient server placement method should always be better than the random placement. In the actual performance study, we generated 10 random placements and selected the particular random server set, which achieved the best system performance.

2) *K-Median Placement*: In [12], Francis et al. presented the K -median problem, which is NP-hard. Consider a graph $G = (V, E)$ with weights. The weights on the nodes represent the bandwidth demand and the weights on the edges depict the distance between two nodes. Serving a request incurs network cost equal to the bandwidth of the shortest path between the client node and a server. The problem is to place K servers on the nodes in order to minimize the total network cost, given that at the most one server can be placed on one node. $d_{ij} = d(i, j)$ is the shortest distance between AS i and AS j in terms of hop numbers. In the previous section, r_j is the bandwidth demand on AS j . Therefore, treating weighted distances as a ‘‘cost’’, we have the following integer programming formulation of the K -median placement:

$$\min \sum_{i=1}^{|L|} \sum_{j=1}^{|V|} r_j d_{ij} y_{ij} \quad (15)$$

subject to:

$$\forall j \in R, \sum_{i=1}^{|L|} y_{ij} = 1 \quad (16)$$

$$y_{ij} \leq x_i \quad (17)$$

$$\sum_{i=1}^{|L|} x_i \leq K \quad (18)$$

where

$$x_i = \begin{cases} 1, & \text{if a server is placed on node } i \in L \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} 1, & \text{if node } j \text{ is assigned to a server at node } i \\ 0, & \text{otherwise} \end{cases}$$

The requirements on y_{ij} can be further written as simple nonnegativity constraints ($y_{ij} \geq 0$). This is sufficient, at least for the case of nonnegative weighted distances [12], because once x_i is specified, it is never optimum for any y_{ij} to take on a value other than 0 or 1.

The objective function in Eqn. 15, consists in minimizing the total weighted cost to serve a user's bandwidth request. Eqn. 16 indicates that each node $i \in V$ can only be assigned to one server. Eqn. 17 ensures that users can only be assigned to servers that have been actually placed. Eqn. 18 indicates that only K servers can be placed at the most.

C. Server Access Bandwidth Allocation

Based on the solutions of SR-Optimal and SR-Greedy, the server locations can be identified using x_i and the access bandwidth for server i can be purchased using z_i . We assigned the actual access bandwidth for server i to be z_i/μ , such that the server utilization level is μ . We compared our models (SR-Optimal and SR-Greedy) with the random placement and the K-median model. To conduct a fair comparison, we fixed the same number of servers of the random placement and the K-median model to be the minimum number of servers obtained using SR-Optimal and SR-Greedy. In addition, we assigned the same total access bandwidth for all the four models. Each server in the random and K-median model is assigned the same amount of access bandwidth.

D. User-Server Assignment

The solution of SR-Optimal or SR-Greedy provides the link flow distribution (f_{ij}). The flow decomposition theorem [11] shows that any unicast flow can be decomposed into flows along source-destination paths. The request redirection can be conducted accordingly using these source-destination paths. Note that these source-destination paths are determined by the minimum total traffic flows; therefore, they are not necessarily the shortest paths. Thus, the successful deployment of the model requires the network to support source routing. However, the popularity of policy-based routing between ASs might help enable source routing at the AS level [13]. The flow distribution provides guidelines for ISPs to deploy their routing

policies.

Since the requests in one AS might be served by multiple servers obtained from the optimization models, we propose a probabilistic request redirection scheme as follows. The average bandwidth demand on AS k is r_k and the end-users in AS k can be served by multiple servers $k_i (i = 1, \dots, q)$ with the bandwidth partition $b_k^i (i = 1, \dots, q)$ such that $\sum_{i=1}^q b_k^i = r_k$. When a request arrives, it is redirected to a server with the probability $p_k^i = \frac{b_k^i}{r_k} (i = 1, \dots, q)$. In case that server k is selected, the connection follows the path obtained by SR-Routing. In the long term, the flow partition follows the bandwidth partition obtained from SR-Optimal or SR-Greedy.

III. SIMULATION MODEL

After streaming CDN servers are placed in the network, it is interesting to study the actual system performance as perceived by end-users. In addition, the solutions to our proposed server placement models are based on long-term stochastic parameters, such as the average client's bandwidth demand, the average available bandwidth on the links between ASs, etc. The stochastic nature of the problem incurs uncertainty. The stochastic placement models create more modelling and optimization opportunities while the solvable problem size is very small [14]. Our two location models are two deterministic approximation models for the stochastic network situation, where the approximation consists of replacing the client's bandwidth demand and the link available bandwidth with their mean values. In reality, the client arrival process exhibits some stochastic properties, the dynamic client behaviors might influence the system performance. Further, the server are assumed to be placed on the public Internet; therefore, the cross traffic changes the available bandwidth on links as time goes on. This deterministic approximation can yield results far from the optimum [10]. We conducted a simulation-based study to investigate the stochastic behavior of the system. The simulation model is based on SimLib2.2 [15].

In addition, the average user bandwidth demand and available bandwidth are estimated with errors. Based on the deviated parameters, the actual performance of the optimal server placement might not necessarily be better than those of other schemes. Sensitivity analysis is normally conducted to study the influence of the deviated parameters upon the optimality of the optimization model. However, the traditional sensitivity analysis generally assumes that only one parameter is changed while others are kept unchanged. In our case, it is possible that multiple parameters vary at the same time; therefore, we employed a sampling-based evaluation to study the sensitivity performance of different server placement models.

A. Overview

The simulation model simulates the system behavior at the connection level, as shown in Fig. 1. The model consists of a streaming request generation unit, a cross traffic generation unit, a traffic multiplexer, a connection holding unit and a message sink. The internal structure of the request generator and the cross traffic generator are shown in Fig. 2 and Fig. 3, respectively. Their message flow is described as follows:

- There are two types of connection request messages: streaming requests and cross traffic requests. Streaming connections can be blocked if there are currently not enough network resources to serve new connection's bandwidth requirement.

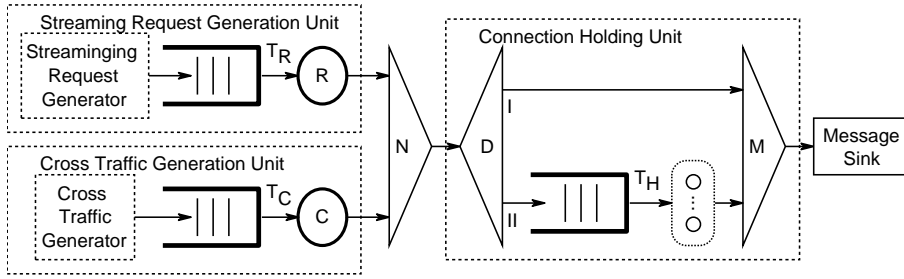


Fig. 1. The Simulation Model.

The cross traffic connections are always accepted because there is actually no call admission control mechanism for these connections.

- Each node k is associated with a streaming request generator. The streaming request generator generates a streaming connection request message. Messages are aggregated together via a multiplexer and arrive at the input port of a queue. The inter-arrival time T_k of the streaming request arrival process on node k between two messages is negative-exponentially distributed. The queue delivers the messages to the service unit, or stores them for later retrieval if the unit is busy. Message storage occurs according to the FIFO principle.

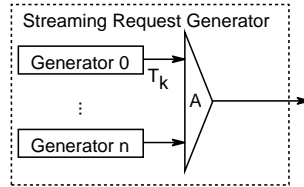


Fig. 2. The streaming request generator model.

- The service times T_R of the service unit can be used to model the connection set-up time. At the end of a service time, the service unit passes the processed message on to the connection multiplexer and requests another message from its queue. The request messages are processed differently based on the decision as to whether the connections are accepted or not. The streaming request unit can access a global network topology information with the link residual capacities by maintaining a pointer to the network topology data structure. Based on the decisions whether the connections are accepted or not, the network residual capacities are updated.

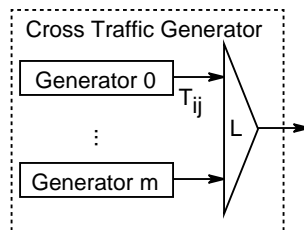


Fig. 3. The cross traffic generator model.

- Each link (i, j) is associated with a cross traffic generator. The generated messages also arrive at a queue. The inter-arrival

time T_{ij} of the cross traffic messages on link (i, j) is negative-exponentially distributed. The queue delivers the messages to the service unit, or stores them for later retrieval if the unit is busy. Message storage occurs according to the FIFO principle.

- The service times T_C of the service unit in the cross traffic routing unit are set to zero because there is no call admission control for them. At the end of a service time the service unit passes the processed message on to the connection multiplexer and requests another message from its queue. The cross traffic generation unit also updates the global network information database after a cross traffic connection is set up.
- The messages from the streaming request generation unit and the cross traffic generation unit arrive at the connection multiplexer and are passed on to the connection holding unit. If the connections are blocked because the network does not have enough resources, the messages traverse path I without any delay; if the connections are accepted, the messages go along path II and jump into the service unit with an infinite number of servers. The cross traffic messages always go along path II. The connection holding time information is generated when the request is generated and it is carried by the message and the service time T_H is determined by the connection holding time.
- When the messages arrive at the message sink, nothing happens for those blocked connections. For those accepted connections including the streaming connections and the cross traffic connections, the arrival of the messages indicate the termination of the connections. Therefore, the sink releases the network resource occupied by these connections and updates the network residual capacities accordingly. The network information database is maintained by the global simulation model. With the assumption that there is no delay in updating the network database, the traffic sink can maintain a pointer to the a global network topology information database for a direct access.

B. Data Collection

In order to obtain the statistics of the performance metrics, the simulator implements the system inspection function using a timer. Fig. 4 depicts how system inspection is conducted. After the simulator passes the transient phase, it starts the system inspection by triggering the inspection timer periodically. The timeout of the inspection timer indicates the time instance of one batch data collection. At the end of each system inspection, the simulator checks the network information database and calculates the performance metrics of each on-going connection in the network.

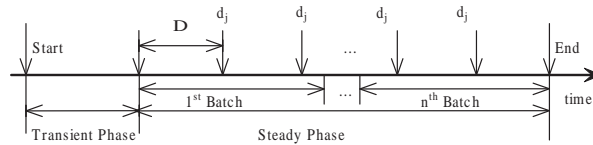


Fig. 4. The data collection procedure of system inspection.

$$\bar{X} = \frac{1}{n} \bullet \sum_{i=1}^n \bar{X}_i \quad (19)$$

The simulation experiments were run in multiple batches. In each batch, the system performance metrics are obtained in a batch average, i.e., \bar{X}_i for the i_{th} batch. The overall performance metric \bar{X} is obtained by averaging all the \bar{X}_i using Eqn. 19. The confidence interval is calculated using the set of \bar{X}_i . The error bars indicate the 95% confidence intervals.

Note that the simulator can further deploy the early connection termination mechanism to ensure that the network utilization is pushed back to a level at which the remaining connections can achieve their QoS requirements.

The QoS performance of the streaming service is mainly determined by whether the throughput of one connection can be maintained. This is reflected by the performance of the loss ratio perceived by end-users in the lifetime of connections. Because our simulation is conducted at the connection level, the packet level loss ratio could not be captured. It can, however, be estimated at the connection level as follows.

Considering link k , the number of total flows along the link is n_k , including normal streaming flows and cross traffic flows. The loss ratio for flow j over link k at the connection level can be approximated in [16] by: $d_{kj} = \frac{(\sum_{i=1}^{n_k} f_i - C_k)^+}{\sum_{i=1}^{n_k} f_i}$, where $(a)^+ = \max(0, a)$. With the assumption of link independence, we define the overall loss ratio for flow j as $d_j = 1 - \prod_k (1 - d_{kj})$.

In addition, the instantaneous loss ratio cannot reflect the overall performance of a streaming connection because its QoS is insensitive to temporary congestion due to the buffering mechanism. We also define a streaming connection as congested if its loss ratio is bigger than a threshold in a set of consecutive system inspections. Without an explicit specification, the default values used are as follows. The loss ratio threshold is set as 0.2. The system checking period is 2 second and the connection congestion threshold is set to 10 second. Therefore, if the loss ratio of a connection in its lifetime is larger than 0.2 in five consecutive system inspections, this connection is considered to be congested. Once a streaming connection is identified as one congested connection, the status does not change during the rest of its lifetime.

IV. PERFORMANCE EVALUATION

The performance evaluation involved two parts. One is the effectiveness of the greedy decomposition method and the other is the actual QoS performance perceived by the end users in different server placement models. First, we introduce the network models used in the performance evaluation.

A. Network Models

Modelling the Internet has become an active research area [17], [18], [19]. In this study, we consider two types of network topologies, grid network, random network. In a flat random topology, ASs are distributed at random locations in a plane and an edge is added between a pair of nodes with probability p . Though the random topology does not reflect the real structure of the Internet, it is commonly used in networking studies due to its simplicity.

1) *Parameter Settings*: The parameter configuration involved two parts: the settings for the network topology generation and the network traffic pattern. We assume that there are two types of homogenous traffic in the network: streaming connections and cross traffic connections. They differ in terms of the bandwidth consumption and the connection holding time. In particular, each streaming connection requires more bandwidth and is sustained longer compared to that of a cross traffic connection.

Each node is associated with one streaming arrival process and each link $e(i, j)$ is associated with one cross traffic arrival process. The parameters are denoted in Table I.

TABLE I
PARAMETER DEFINITIONS

Parameters	Definition
T_k	average inter-arrival time of streaming connections in AS k
T_s	average holding time of streaming connections
λ_k	arrival rate of streaming connections in AS k
r_k	bandwidth demand in AS k
bw_s	average bandwidth demand per streaming connection in AS k
T_{ij}	average inter-arrival time of cross traffic connections on link (i, j)
T_c	average holding time of cross traffic connections
β_{ij}	arrival rate of cross traffic connections on link (i, j)
c_{ij}	capacity of link (i, j)
bw_c	average bandwidth demand per cross traffic connection
b_{ij}	average available bandwidth on link (i, j)

The traffic parameter configuration mainly concerns the settings of the mean holding time of streaming connections and cross traffic connections. The average inter-arrival time T_k and T_{ij} , which are used to configure the streaming request generator and the cross traffic generator, can be derived as follows.

$$\lambda_k * T_s * bw_s = r_k,$$

$$T_k = 1/\lambda_k.$$

Therefore,

$$T_k = \frac{T_s * bw_s}{r_k},$$

$$\beta_{ij} * T_c * bw_c + b_{ij} = c_{ij},$$

$$T_{ij} = 1/\beta_{ij}.$$

Therefore,

$$T_{ij} = \frac{T_c * bw_c}{c_{ij} - b_{ij}}.$$

2) *Grid Topology*: In order to understand the simulation and the intricacy of the system, we first consider the 4x4 simple grid network shown in Fig. 5. The server location sets obtained from different models are depicted in Fig. 6 with the average minimum distance (d_{min}) in terms of hops for each node to one of the servers.

The link capacity c_{ij} between AS i and AS j is 2250 units and the average available bandwidth a_{ij} is set to 625 units. The average user bandwidth demand (r_k) in AS k is chosen to be 625 units. The server's Internet access bandwidth is constrained

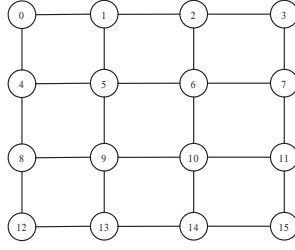


Fig. 5. Grid network (4x4).

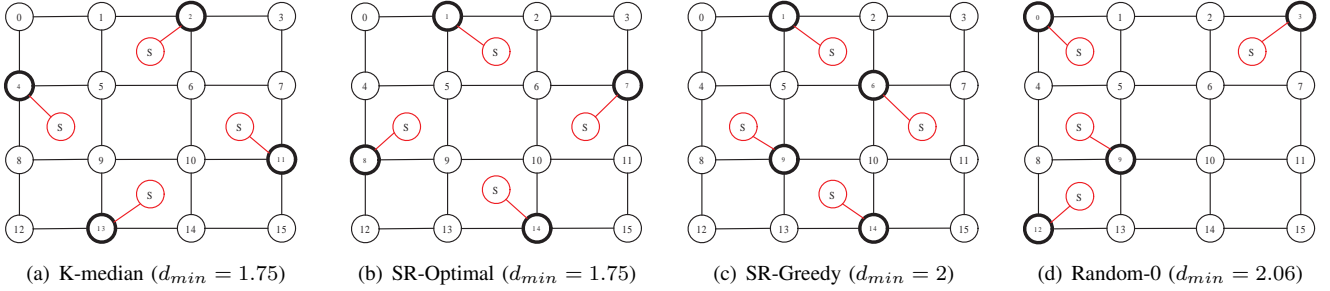


Fig. 6. Different server placements.

by 5000 units; however, the actual assigned access bandwidth is determined by the model. For each request, stream bandwidth bw_k is uniformly distributed in $\{5, 10, 15, 20\}$ units. For each cross traffic connection, the bandwidth bw_{ij} is uniformly distributed in $\{1, 2, 3, 4\}$ units.

3) *Random Topology*: The actual parameters for the random topology are configured as follows. The link capacity c_{ij} between AS i and AS j is uniformly distributed in $[5, 20]$ Mbps. The user bandwidth demand (r_k) in AS k followed a uniform distribution in $[5, 20]$ Mbps. The server's Internet access bandwidth is constrained by $a_k = 100$ Mbps.

In AS k , stream bandwidth for each request bw_k is uniformly distributed in $\{100\text{kbps}, 200\text{kbps}, 300\text{kbps}, 400\text{kbps}\}$. The holding time of streaming connections has a negative-exponential distribution with the mean $T_s = 600$ sec, if not specified.

On each link (i, j) , the bandwidth for each cross traffic connection bw_{ij} is uniformly distributed in $\{20\text{kbps}, 40\text{kbps}, 60\text{kbps}, 80\text{kbps}\}$. The cross traffic connection's holding time has a negative-exponential distribution with the mean $T_c = 60$ sec.

B. Computation Complexity

In this section we demonstrate the effectiveness of the greedy decomposition method in terms of computation time. All the cases are solved on a Pentium III 933 PC with 256M RAM. The models (SR-Optimal, SR-Greedy, Random, K-median) are implemented using C++. Commercial software, CPLEX, is used to solve these models [20]. For each problem, we allowed the software to run until an optimal solution is obtained.

We solved the model for 20 sets of network parameters for the case that the server cost $d_i^0 = 1000$ and $d_i = 1$. In all the example cases, SR-Optimal and SR-Greedy are found to have the same total cost $\sum_{i=1}^n (d_i^0 * x_i + d_i * z_i)$. This result is not surprising and it is explained in Section II-A. In particular, the number of servers obtained by SR-Greedy for various random topologies are depicted in Table II. The mean value and the standard variance of CPU time used by SR-Greedy are plotted in

TABLE II
SUMMARY OF NUMBER OF SERVERS OF SR-GREEDY FOR VARIOUS RANDOM TOPOLOGIES

Random Topology($p=0.05$)	Average number of servers	Standard deviation of number of servers
50	14.4	1.85
60	14.4	1.09
70	14.1	1.67
80	14.7	1.53
90	14.9	1.37
100	14.8	1.06
200	26	0
300	38	0
400	51	0
500	63	0
600	75	0
700	87	0
800	100	0
900	113	0
1000	125	0

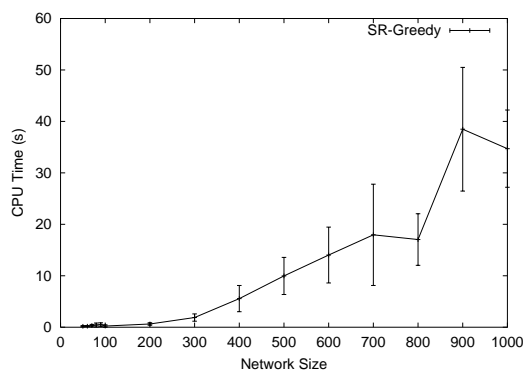


Fig. 7. The CPU time of SR-Greedy for random topologies with $p = 0.05$.

Fig. 7. The figure shows that with an increase in AS numbers, the optimization time increases as well but it is within dozens of seconds when the network size is less than 1000 ASs. The ratio of the CPU time of SR-Optimal and SR-Greedy is plotted in Fig. 8. The figure shows that with an increase in AS numbers, the optimization time required by SR-Greedy becomes much less than that of SR-Optimal. Note that when the network size is larger than 80 nodes, it cannot be guaranteed that SR-Optimal can be solved exactly. In some cases, the computer runs out of memory before the optimal solution of SR-Optimal can be obtained. However, up to 1000 nodes, all cases can be solved within dozens of seconds using the greedy decomposition. In Fig. 9 the average and standard deviation of total traffic flow for SR-Optimal and SR-Greedy are plotted. The gap seems to increase at a quite slow rate with an increase in the network size. Eventually, the ratio between the gap and SR-Optimal is around 12%, even when the network size is larger.

C. Simulation Results

1) *Grid Network*: The grid network is configured as in Section IV-A2. There are two types of packet dropping, in the network and in the server access link. In Fig. 10 and Fig. 11, the random placement exhibits a much higher loss ratio both in

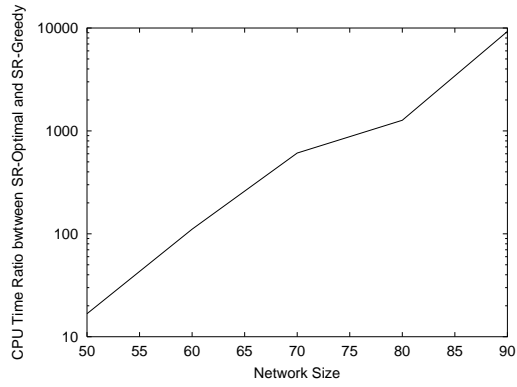


Fig. 8. The ratio of the CPU time of SR-Optimal and SR-Greedy.

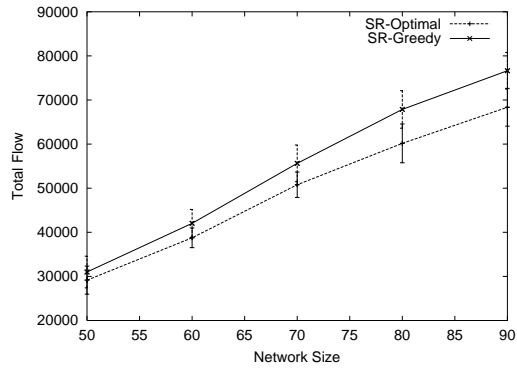


Fig. 9. The total flow of SR-Optimal and SR-Greedy.

the network and in the server access links. Though the average minimum distance ($d_{min} = 2.06$) for this random placement is only slightly larger than that of SR-Greedy ($d_{min} = 2$), its loss ratio is much higher than that of SR-Greedy with different network traffic dynamics and server access loading.

In Fig. 12, the congestion situation of streaming connections is depicted. Fig. 12(a) shows that congestion is more severe when the streaming requests are more bursty. (The average user bandwidth request is kept the same in different simulation

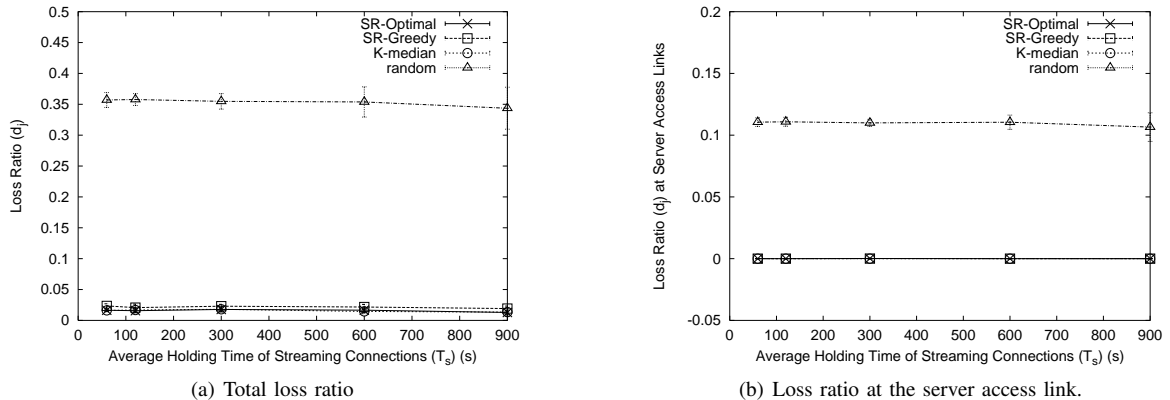
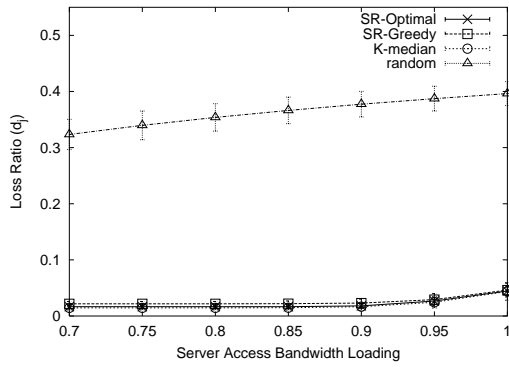
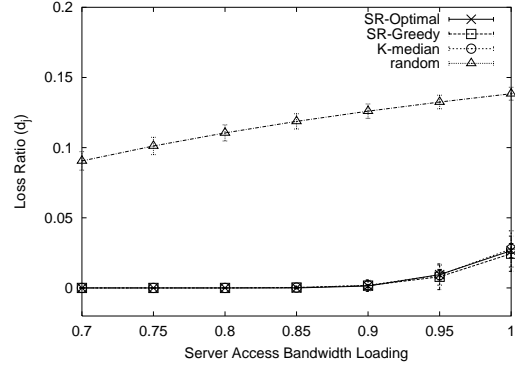


Fig. 10. Loss ratio with the server access bandwidth loading at 0.8.

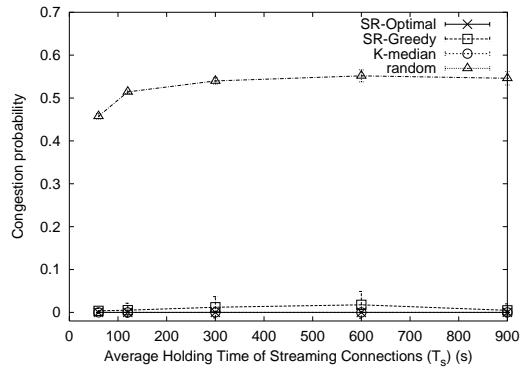


(a) Total loss ratio

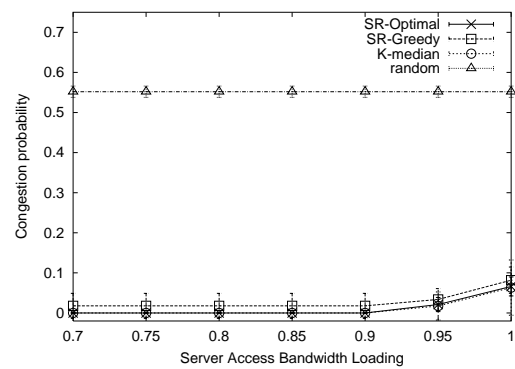


(b) Loss ratio at the server access link.

Fig. 11. Loss ratio with the average holding time of streaming connections at 600 sec.



(a) Server access bandwidth loading at 0.8.

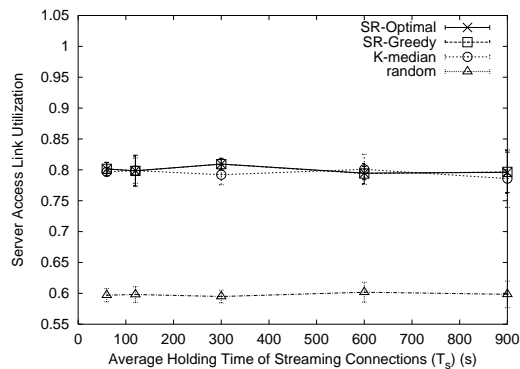


(b) Average holding time of streaming connections at 600 sec.

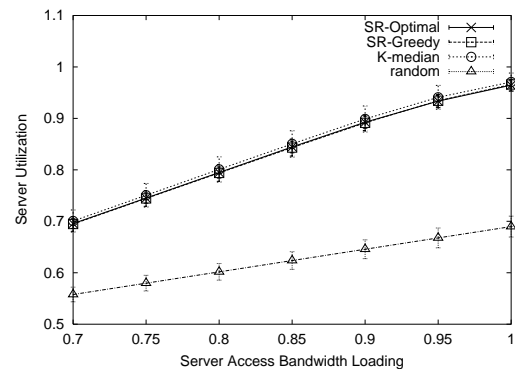
Fig. 12. Congestion probability of streaming connections

experiments; therefore, with a smaller average holding time of streaming connections, streaming requests are more bursty.)

Fig. 12(b) shows that when the server access loading increases, more streaming connections are congested at the server access links.



(a) Server access link loading at 0.8.



(b) Average holding time of streaming connections at 600 sec.

Fig. 13. Server access link utilization

In Fig. 13, it can be observed that the average server access link utilization in a random placement is much smaller than that

of other placements. This explains why the loss ratio in the random placement is much larger although the same total access bandwidth is assigned the same for the four schemes. When we inspected the utilization of individual servers, some servers are overloaded and some servers are not utilized at a reasonable level in the random placement. Therefore, the load balancing is very important issue in the streaming server placement.

2) *Random Network*: We studied the performance of a random network with the node number 50 and $p = 0.05$.

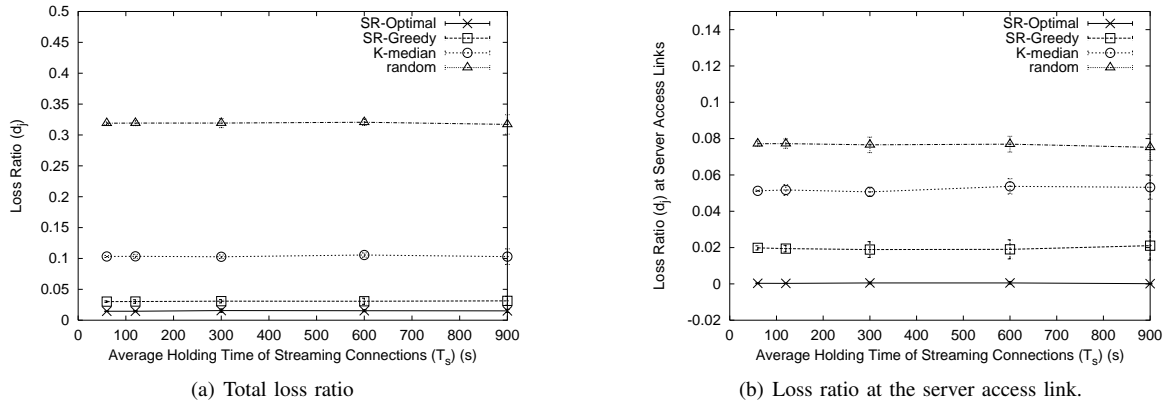


Fig. 14. Loss ratio with the server access bandwidth loading at 0.8.

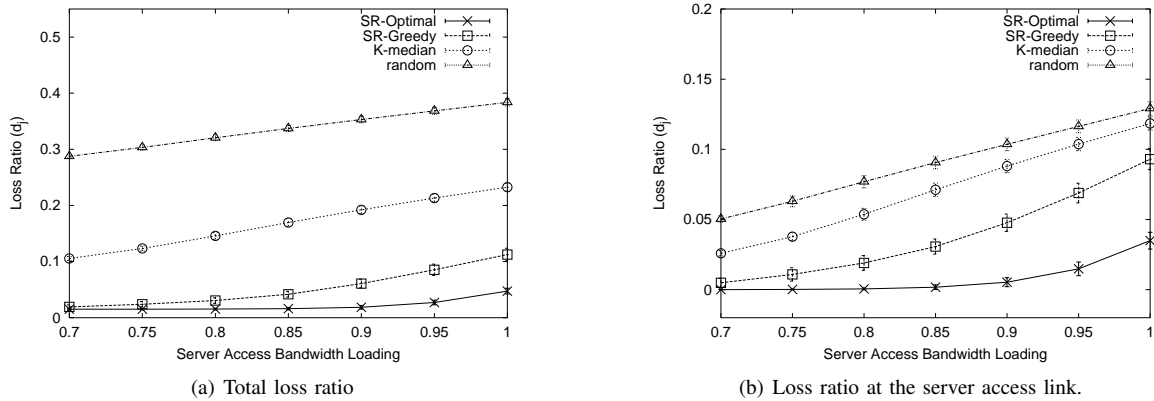
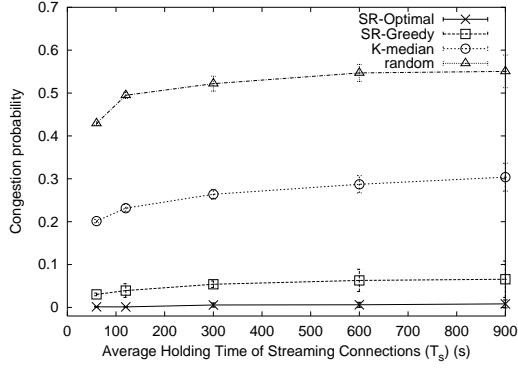


Fig. 15. Loss ratio with the average holding time of streaming connections at 600 sec.

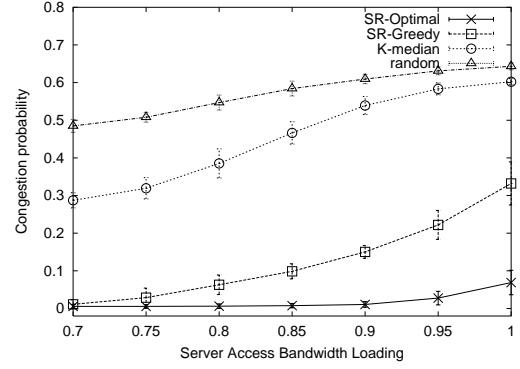
As shown in Fig. 14 and Fig. 15, the random placement exhibits much higher loss ratio both in the network and in the server access links than in other placements. In addition, SR-Optimal has the smallest loss ratio and SR-Greedy does not perform as well as SR-Optimal. K-median is a distance-minimization-oriented placement model and its loss ratio cannot compete with that of bandwidth-optimization-oriented placement models, such as SR-Optimal and SR-Greedy.

As in the case of the grid network, in Fig. 16, we closely inspected the congestion situation of streaming connections. Fig. 16(a) shows that the congestion is more severe when streaming requests are more bursty. Fig. 16(b) shows that when the server access loading increases, more streaming connections are congested at the server access links.

In Fig. 17, it can be observed that the average server access link utilization in a random placement is significantly smaller

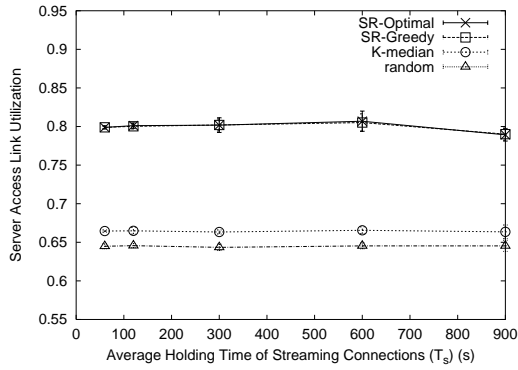


(a) Server access bandwidth loading at 0.8.

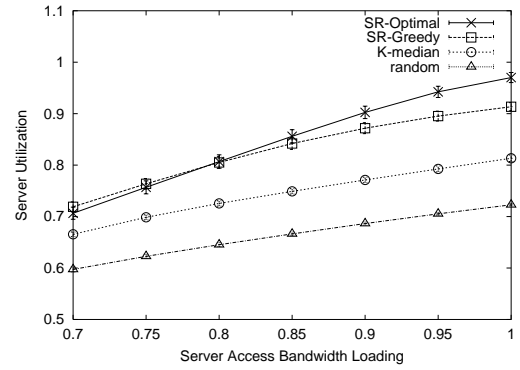


(b) Average holding time of streaming connections at 600 sec.

Fig. 16. Congestion probability of streaming connections



(a) Server access link loading at 0.8.



(b) Average holding time of streaming connections at 600 sec.

Fig. 17. Server access link utilization

than that of other placements. An inspection of the utilization of individual servers revealed that some servers are overloaded, and that some servers has very low utilization in the random placement. K-median, also, could not maintain a good load-balancing performance. This is not surprising because the aim of K-median is to minimize the average minimum distance in terms of hop numbers for each node to one of the servers without any considerations of the link and server capacities.

3) *Sensitivity performance of the bandwidth demand r_k* : In this experiments of this section, we varied the request bandwidth r'_k on AS k from the original value r_k but kept other parameters unchanged. The deviation factor is denoted as δ ($\delta > 0$). There are two cases under study: the double-sided deviation and the single-sided deviation. In the double-sided deviation scenario, the r'_k is uniformly generated in $r_k * [1 - \delta, 1 + \delta]$ while in the single-sided deviation scenario, r'_k is uniformly generated in $r_k * [1, 1 + \delta]$. For each set of r'_k , we studied the system performance of the four placement models.

As shown in Fig. 18 and Fig. 19, we configured the server access bandwidth loading at $\mu = 0.8$ to study the sensitivity impact of r_k upon the system. For the double deviation case shown in Fig. 18(a), the loss ratios of the four schemes increase slowly as the deviation increases. For the single deviation case, as in Fig. 18(b), the loss ratios of SR-Optimal and SR-Greedy increase more quickly than those of random and K-median when the deviation increases, and they are still smaller than those of random and K-median in the deviation range that is studied.

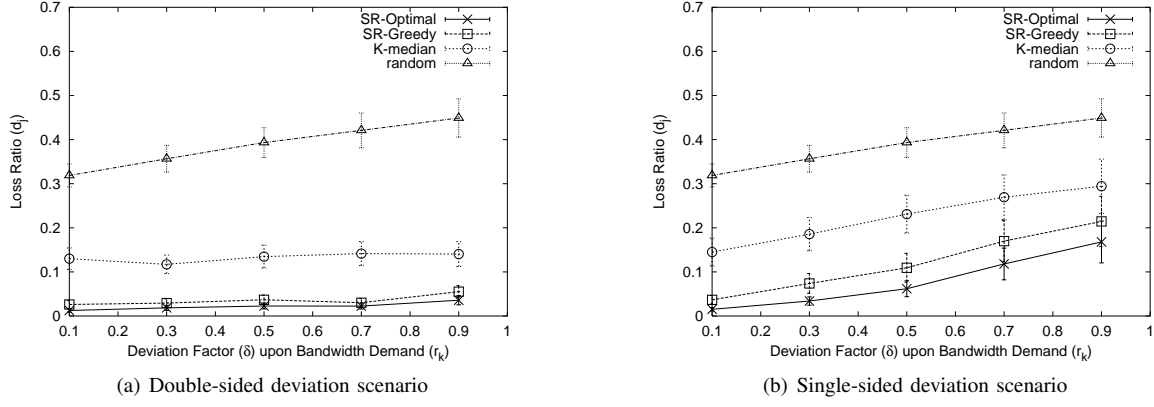


Fig. 18. Sensitivity impact of r_k upon loss ratio with the server access link loading at 0.8

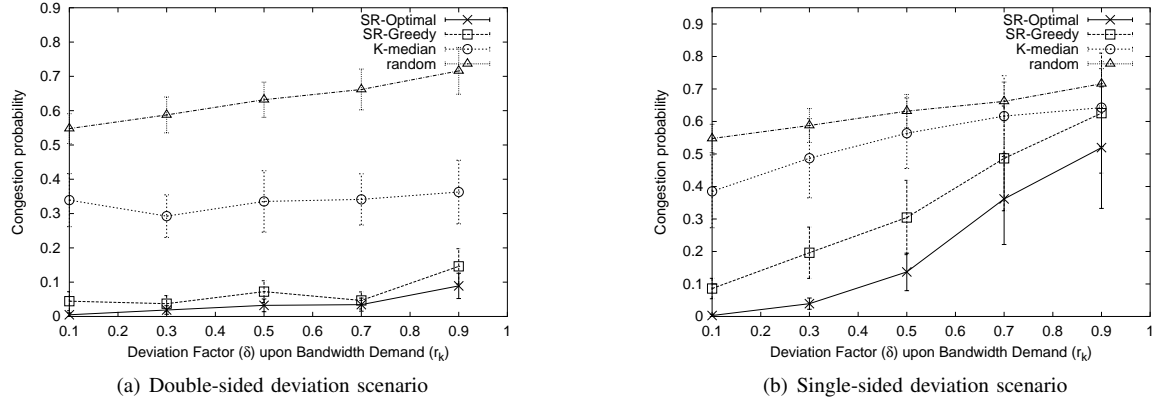


Fig. 19. Sensitivity impact of r_k upon congestion probabilities of streaming connections with the server access link loading at 0.8

For the double deviation case shown in Fig. 19(a), the congestion probabilities of the streaming connections for the four placements is not sensitive to the double-sided deviation of r_k . This is because one server generally serves clients from multiple ASs. Bandwidth demands from some ASs might increase while those from other ASs might decrease at the same time. Due to the aggregation effect, the bandwidth demand to one server might be maintained at a constant level. For the single deviation case shown in Fig. 19(b), the congestion probabilities of SR-Optimal and SR-Greedy increase more quickly than those of random and K-median as the deviation increases, and they are still bigger than those of random and K-median in the deviation range under study.

In Fig. 20 and Fig. 21, the cases in which the deviation ratio r_k is 0.9, are shown. For the double deviation case, see Fig. 20(a), the loss ratios of the four schemes increase slowly as the server access link loading increases. For the single deviation case, see Fig. 20(b), the loss ratios of SR-Optimal and SR-Greedy increase more quickly than that of random and K-median as the deviation increases, and they are still smaller than those of random and K-median in the deviation range.

In Fig. 21 the sensitivity performance of the congestion probabilities of the streaming connections with a deviation ratio (δ) of r_k at 0.9 are depicted. For the double deviation case shown in Fig. 21(a), the congestion probabilities of the four schemes increase slowly as the server access link loading increases. For the single deviation case in Fig. 21(b), the congestion

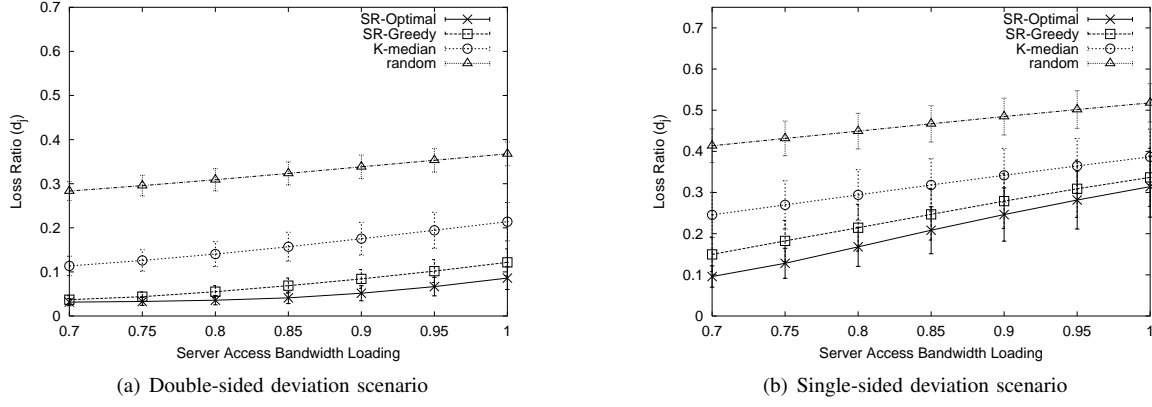


Fig. 20. Sensitivity impact of r_k upon loss ratio with the deviation ratio (δ) of r_k at 0.9

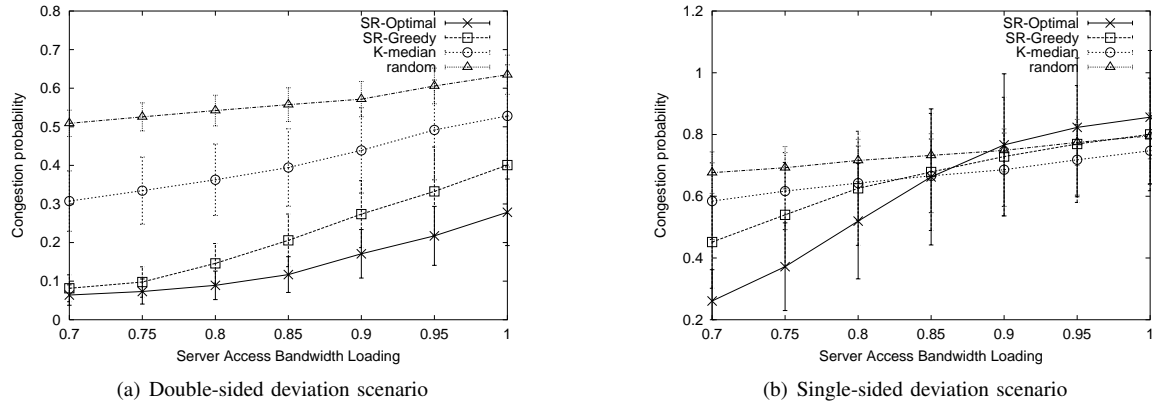


Fig. 21. Sensitivity impact of r_k upon congestion probabilities of streaming connections with the deviation ratio (δ) of r_k at 0.9

probabilities of SR-Optimal and SR-Greedy increase more quickly than those of random and K-median. When the loading is larger than 0.85, the three optimization based placements have worse congestion probabilities than the random placement does. This indicates the necessity for careful server access bandwidth allocation. In Fig. 20 and Fig. 21, it can also be seen that the loss ratio and the congestion probability are two related but different performance metrics. As illustrated in Fig. 20(b), SR-Optimal has the smallest loss ratio of the four models, and as shown in Fig. 21(b), SR-Optimal has a larger congestion probability than K-Median does when the server access loading is bigger than 0.85.

4) *Sensitivity performance of the link available bandwidth b'_{ij}* : We varied the link available bandwidth b'_{ij} on edge (i, j) from the original value b_{ij} but retained the other parameters. As in other simulation experiments, we studied two deviation cases. In the double-sided deviation scenario, the b'_{ij} is uniformly generated in $b_{ij} * [1 - \delta, 1 + \delta]$ while in the single-sided deviation scenario, b'_{ij} is generated uniformly in $b_{ij} * [1 - \delta, 1]$.

As shown in Fig. 22 and Fig. 23, we configured the server access bandwidth loading at $\mu = 0.8$ to study the sensitivity impact of b_{ij} on the system. As Fig. 22 shows, the loss ratios of four placements increase when the server access link loading increases, and the loss ratios of SR-Optimal and SR-Greedy are smaller than those of random and K-median in both the single and double deviation cases. This indicates that when the server loading increases, more and more streaming connections are

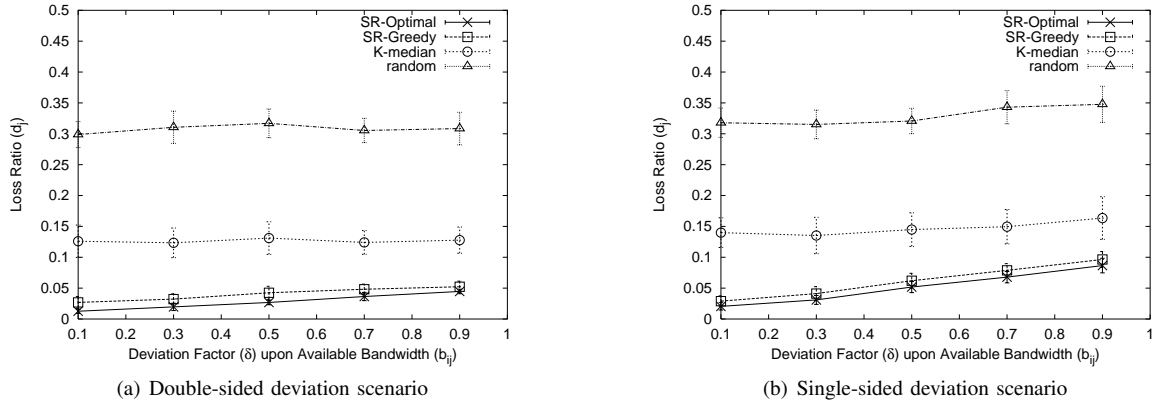


Fig. 22. Sensitivity impact of $b_{i,j}$ upon loss ratio with the server access link loading at 0.8

liable to congest at the server access links.

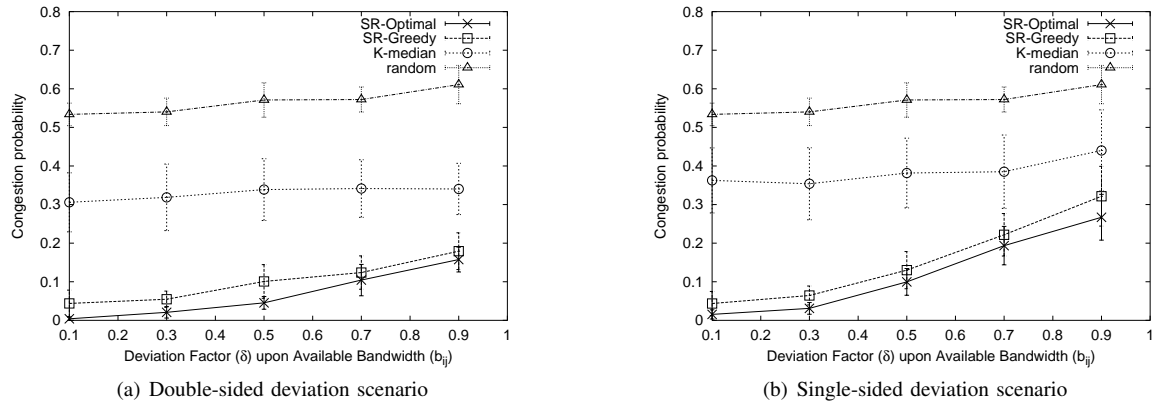


Fig. 23. Sensitivity impact of $b_{i,j}$ upon congestion probabilities of streaming connections with the server access link loading at 0.8

The congestion probabilities of four models are plotted in Fig. 23. The server link loading is set at 0.8. The congestion probabilities of four placements increase when the server access link loading increases, and the congestion probabilities of SR-Optimal and SR-Greedy remain higher than those of random and K-median in both the single and double deviation cases.

In Fig. 24 and Fig. 25, the case in which the deviation ratio of $b_{i,j}$ is 0.9, is shown. For both the double deviation case and the single deviation case, the loss ratios of the four schemes increase slowly as the server access link loading increases. This indicates that the loss ratio is not sensitive to the deviation of $b_{i,j}$.

Shown in Fig. 25 are the plots of the congestion probabilities of the four models when the deviation ratio of $b_{i,j}$ is 0.9. The congestion probabilities of the four placements increase when the deviation ratio of $b_{i,j}$ increases, and the congestion probabilities of SR-Optimal and SR-Greedy remain higher than those of the random and K-median in both the single and double deviation cases.

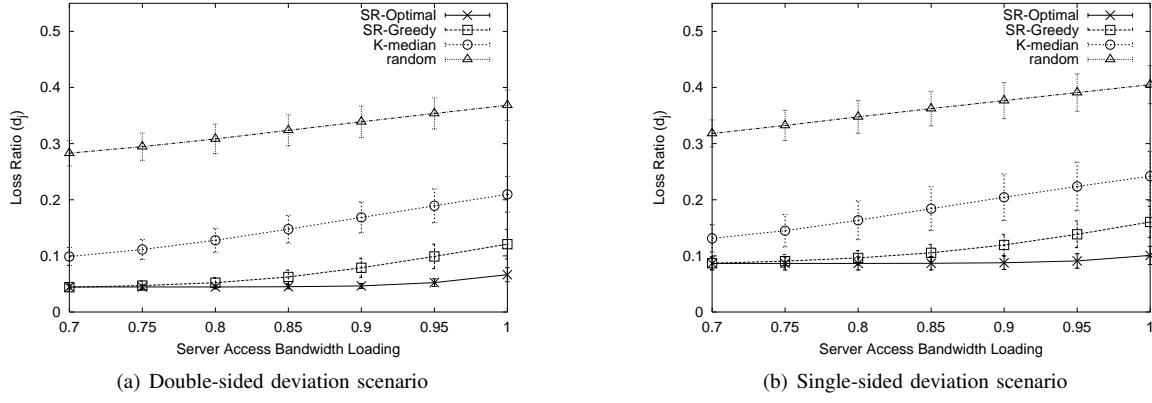


Fig. 24. Sensitivity impact of b_{ij} upon loss ratio with the deviation ratio (δ) of b_{ij} at 0.9

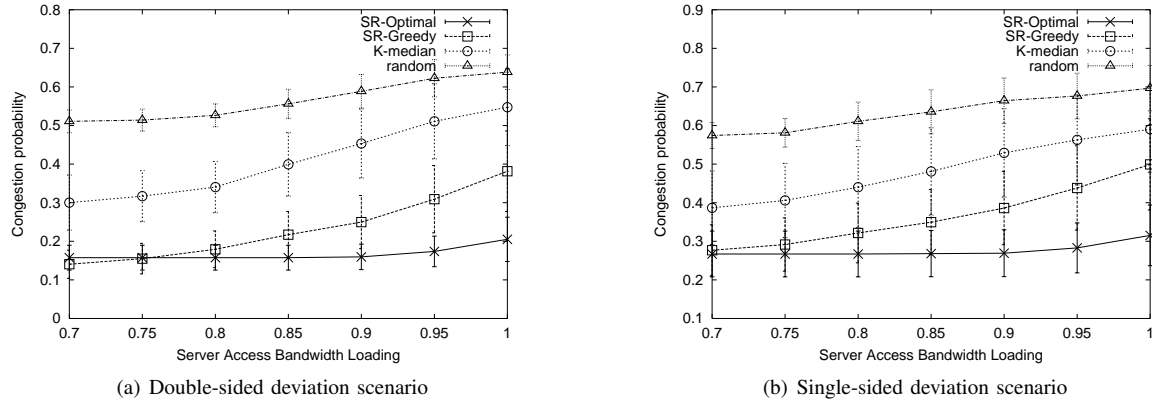


Fig. 25. Sensitivity impact of b_{ij} upon congestion probabilities of streaming connections with the deviation ratio (δ) of b_{ij} at 0.9

V. CONCLUSIONS

The future Internet will support the delivery of rich streaming media contents. The streaming applications result in severe scalability problems in both network and streaming servers. The deployment of CDN servers at the network edge can increase the reliability and performance of the streaming service. In this paper, we discussed the CDN server placement problem as an MIP model and proposed a greedy decomposition method (SR-Greedy) to solve the model. The simulation results demonstrated the effectiveness of the optimization model in the grid and the random Internet AS-level topologies. The numerical results showed that SR-Greedy can be solved within tens of seconds for a topology within 1000 ASs. The SR-Optimal and SR-Greedy outperformed the random model and the K-median model in the simulation. In general, our models are insensitive to the double-sided deviation of the user bandwidth demand available bandwidth between ASs; however, the single-sided estimation error of user bandwidth demand has a large impact on the system performance. In particular, with the single-sided deviation factor of bandwidth demand ($\delta = 0.9$), SR-Optimal and SR-Greedy maintain a better performance than the K-Median and random models do until the server access loading is larger than 0.85. In addition, it is relatively easier to decrease the server access loading by purchasing more access bandwidth. CDN providers can monitor the loading of CDN servers and maintain

it at less than a critical loading threshold. Through our simulation, this threshold is found to be 0.85.

The server placement model proposed in this paper requires the cooperation between ASs and we are now working on the placement model with the additional shortest path constraint. Because the grid and random topologies cannot reflect the actual Internet topology, we are continuing our model evaluation over more realistic Internet topologies such as GT-ITM [17], BRITE [19] and the inferred AS-level topology from the BGP table [2].

REFERENCES

- [1] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection," in *Proceedings of INFOCOM*, New York, NY USA, June 23-27 2002.
- [2] H. Chang, R. Govindan, S. Jamin, S. Shenker, and W. Willinger, "Towards capturing representative AS-level Internet topologies," in *Proceedings of ACM Sigmetrics*, June 2002.
- [3] P. Barford, J.-Y. Cai, and J. Gast, "Cache placement methods based on client demand clustering," in *Proceedings of INFOCOM*, New York, NY USA, June 23-27 2002.
- [4] M. Jain and C. Dovrolis, "End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput," *IEEE/ACM Trans. Netw.*, vol. 11, no. 4, pp. 537–549, 2003.
- [5] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE JSAC Special Issue in Internet and WWW Measurement, Mapping, and Modeling*, vol. 21, no. 6, pp. 879–893, Aug. 2003.
- [6] Y. Chen, R. Katz, and J. Kubiawicz, "Dynamic replica placement for scalable content delivery," in *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [7] S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang, "On the placement of Internet instrumentation," in *Proceedings of INFOCOM*, Tel Aviv, Israel, Mar. 2000.
- [8] L. Qiu, V. Padmanabhan, and G. Voelker, "On the placement of Web server replicas," in *Proceedings of INFOCOM*, Anchorage, Alaska, April 22-26 2001.
- [9] S. Shi and J. Turner, "Placing servers in overlay networks," in *Proc. of International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, San Diego, July 2002.
- [10] P. Mirchandani and R. Francis, *Discrete location theory*. A Wiley-Interscience publication, 1990.
- [11] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
- [12] R. Francis, J. L. McGinnis, and J. White, *Facility layout and location : an analytical approach*, 2nd ed. Englewood Cliffs, N.J. ; Upper Saddle River, N.J.: Prentice Hall, 1992.
- [13] D. Verma, *Policy-based networking : architecture and algorithms*, 1st ed. New Riders, 2001.
- [14] M. Jamil, A. Baveja, and R. Batta, "The stochastic queue center problem," *Computers & Operations Research*, vol. 26, no. 14, pp. 1423–36, Dec. 1999.
- [15] IKR, "University of Stuttgart, IKR Simlib library 2.2." [Online]. Available: <http://www.ind.uni-stuttgart.de/INDSimLib/>
- [16] J. Roberts, U. Mocci, and J. Virtamo, Eds., *Broadband network traffic : performance evaluation and design of broadband multiservice networks : final report of action COST 242*. Springer, 1996.
- [17] K. Calvert and E. Zegura. (1997) Gt internetwork topology models (GT-ITM). [Online]. Available: <http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html>
- [18] Inet: Internet topology generator, university of michigan. [Online]. Available: <http://topology.eecs.umich.edu/inet/>
- [19] Boston university representative Internet topology generator. [Online]. Available: <http://www.cs.bu.edu/brite/>
- [20] *CPLEX manual*, March 2001, CPLEX Optimization, Incline Village, Nevada.

APPENDIX

Flow Decomposition Algorithm

Flows are usually defined on arcs: x_{ij} is the flow along (i, j) . Alternatively, they can be defined on paths and cycles. To see how to translate between the two descriptions, define an arc flow as a set of flows $\{x_{ij}\}$ that satisfies:

$$\sum_j x_{ij} - \sum_i x_{ij} = -e(i), \text{ for all } i \in V$$

$$0 \leq x_{ij} \leq u_{ij}, \text{ for all } (i, j) \in E$$

Note that this is not necessarily a feasible solution for given node supplies and demands. The quantity $e(i)$ is node i 's imbalance (excess if positive, deficit if negative) with these flows. If $e(i) = -b(i)$ for all i , then the flows are feasible.

Let Π be the set of all paths and Ω be the set of all cycles in the network. A path and cycle flow formulation has decision variables $f(P)$ = the flow on path P , $P \in \Pi$, and $f(W)$ = the flow on cycle W , $W \in \Omega$. For a path P , let

$$\delta_{ij}(P) = \begin{cases} 1, & \text{if arc } (i, j) \text{ is included in path } P \\ 0, & \text{otherwise} \end{cases}$$

and define $\delta_{ij}(W)$ similarly for a cycle W . Then the total flow on arc (i, j) is given by:

$$x_{ij} = \sum_{P \in \Pi} \delta_{ij}(P)f(P) + \sum_{W \in \Omega} \delta_{ij}(W)f(W)$$

- 1) Select a deficit node i with $e_i < 0$. $k = i$.
- 2) For node k , select (k, l) with $x_{kl} > 0$. $next(k) = l$; $k = l$.
- 3) Repeat 2 until (a) $e_k > 0$ or (b) $k = i$.
- 4) If (a) $\Delta f = \min\{-e_i, x_{ij} | (i, j) \in P, e_k\}$. If $\delta_{ij}^P = 1$, $e_i = e_i + \Delta f$, $x_{ij} = x_{ij} - \Delta f$, $e_k = e_k - \Delta f$.
If $e_i < 0$, go to (2).
- 5) If (b) (cycle) $\Delta f = \min\{x_{ij} | (i, j) \in P\}$. If $\delta_{ij}^W = 1$, $x_{ij} = x_{ij} - \Delta f$.
- 6) If $e_i < 0$, go to (2); else find (i, j) s.t. $x_{ij} > 0$.
If no $x_{ij} > 0$, stop; else $k = i$, go to (2).