Minimizing Packet Loss by Optimizing OSPF Weights Using Online Simulation

Hema Tahilramani Kaur, Tao Ye, Shivkumar Kalyanaraman, Kenneth S. Vastola
Electrical Computer and Systems Engineering Department,
Rensselaer Polytechnic Institute, Troy, NY-12180.
Email:{hema,yet3,shivkuma,vastola}@networks.ecse.rpi.edu
Phone: +1 518 276 8289, +1 518 276 8979

Abstract*—In this paper, we present a scheme for minimizing the packet loss in an OSPF network by optimizing link weights using Online Simulation. For this work we have chosen packet loss rate in the network as the optimization metric as it is a good indicator of congestion and impacts the performance of the underlying applications. We have formulated packet loss rate in the network in terms of the link parameters, such as bandwidth and buffer space, and the parameters of the traffic demands. A GI/M/1/K queuing model has been used to compute the packet drop probability on a given link. The problem of optimizing OSPF weights is known to be NP-hard even for the case of a linear objective function [4]. We use On-line Simulator (OLS) [14] to search for a good link weight setting and as a tool for automatic network management. OLS uses fast, scalable recursive random search (RRS) algorithm to search the parameter space. The search algorithm has been compared with the local search heuristic of [3] in terms of the number of function evaluations needed to obtain a "good" OSPF link weight setting. Our results demonstrate that the RRS takes 50-90% fewer function evaluations to find a "good" setting. The amount of improvement depends on the network topology, traffic conditions and optimization metric. We have simulated the proposed OSPF optimization scheme using ns and our results demonstrate improvements of the order of 30-60% in the total packet drop rate for the traffic and topologies considered.

***Index Terms*— Traffic Engineering, OSPF, Optimization, Network Management, Online Simulation**

I. Introduction

Traffic engineering (TE) is defined as the task of mapping traffic flows onto an existing physical topology to meet the performance objectives of network operators. The problem of minimizing packet loss can be broadly classified as a traffic engineering problem. Prior work in the area of TE in OSPF networks has focused on optimizing the OSPF link weights for a given traffic demand estimate [3], [4]. The OSPF routing with these link weights leads to desired routes.

OSPF routes traffic on shortest paths based on the advertised link weights. As a result, the links along the shortest paths may become congested while other links on longer remain idle. OSPF also allows for Equal Cost Multi Path(ECMP) where the traffic is distributed equally among various next hops of the equal cost paths between a source and a destination [10]. This is useful in distributing the load to several shortest paths. However, the splitting of load by ECMP is not optimal as shown in [4]. Various methods have been proposed in literature to balance the traffic across an OSPF network. One of the earlier approaches was to adapt link weights to reflect the local traffic conditions on a link or to avoid congestion ([5], [8]). This is called adaptive routing or traffic-sensitive routing. However, adapting link weights to local traffic conditions leads to frequent route changes and is unstable (see [1], [13] for stability analysis of adaptive routing). Additionally, adaptive routing is based on the local traffic conditions and therefore can not optimize traffic allocation from the network's perspective. These drawbacks are alleviated by assuming the knowledge of entire traffic demand of the network. This is the common assumption in all the TE work in OSPF networks. In practice, estimates of traffic demands can be obtained by using the tools described in [6], [7].

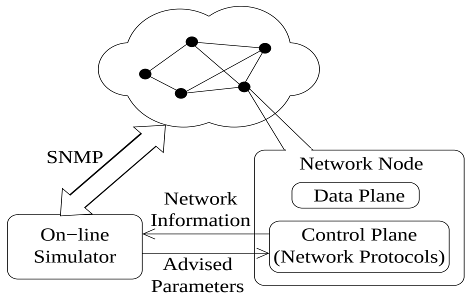


Fig. 1. On-line simulation architecture for automatic network management

We use the Online Simulation (OLS) framework [14] as the network management tool for finding and deploying "good" OSPF link weights. In the OLS framework, the optimization of network protocols is modelled as a general "black box" problem where the objective function is unknown but can be evaluated through simulations. The advantage of this approach is that it makes the OLS a very *flexible* system whose use is *not restricted in one specific protocol or one performance objective*. However, the formulation and evaluation of optimization objective is important. Figure 1 shows the general operation of OLS and its interaction with the network.

The packet loss rate for a given link weight setting can be estimated using packet-level or flow level simulation. However, for fast evaluation an analytic approach is presented to calculate the packet drop rate by using a GI/M/1/K queuing model. General inter-arrival (GI) is more general arrival model as compared to the commonly used poisson process. It allows the arrival process to be more bursty than a poisson process while being mathematically tractable. However, it does not capture the correlation structure. More accurate traffic models, that capture the long range dependence, may be used but they are computationally intractable or very complex. Analysis is considerably faster than the simulation approach and crucial if optimal link weights are obtained using a search scheme. We have simulated the OSPF optimization scheme in ns. Our results show that the total packet loss in the network can be significantly reduced by appropriately setting link weights.

Our contributions in this paper include

- Formulation of packet loss as the optimization objective using a GI/M/1/K queueing model
- Demonstration of Online Simulation framework as the network management tool in OSPF networks
- Application of RRS for finding a "good" link weight setting fast
- Demonstration of improvement in packet loss by using the OLS scheme

The rest of this paper is organized as follows. Section II derives the link packet drop rate from the offered load and formulates the optimization problem. Section III describes the online simulation framework approach for OSPF optimization. Section IV presents the simulation results and finally, Section V presents the conclusions and future work.

II. The Objective Function

Our goal is to minimize the packet loss rate in a network for a given mean and variance of the aggregate demands between each source and destination routers. Let us consider a network represented by a directed graph $\mathcal{G}$=($\mathcal{N}$,$\mathcal{L}$), where $\mathcal{N}$ and $\mathcal{L}$ represent respectively the set of routers and links in the network. Each link $l \in \mathcal{L}$ has bandwidth denoted by B_l and a buffer space of K_l packets. We assume that the packets arriving when the buffer space at a link is full are dropped. We also assume that there is no other active queue management algorithm running at the routers. In addition to the knowledge of bandwidth and buffers at all the links, we assume that an estimate of the mean and variance of the aggregate demand from each source s to destination t is known (using a tool suc as [6]). Let $\mathcal{D}$, $\mathcal{V}$ denote the mean and variance matrix of the estimated aggregate demand.

In the following sub-sections, we will first show how to derive the drop probability for one link based on the offered load. Then we will formulate the optimal general routing problem which aims to optimize the overall packet drop rate for the network. Note that the OSPF optimization problem is just the optimal general routing subject to the shortest path constraint.

A. *Packet Drop Probability on a Single Link*

Let P denote the packet drop probability on a link, λ, σ^2 denote the mean, variance of the offered load to this link in packets per second, and B, K denote its bandwidth and buffer space respectively. In order to find a closed-form expression for the packet drop probability P, let us assume an exponentially distributed packet size with mean $\bar{X}$. However, we consider a general arrival process. We compute the packet drop probability at the link using a GI/M/1/K queuing model. The drop probability of a finite GI/M/1/K has been approximated by an infinite buffer GI/M/1 queue [9] using the following equation.

$$P(N_K = K) = \frac{P(N_\infty = K)}{P(N_\infty \leq K)} \qquad (1)$$

N_K denotes the number of packets in the finite buffered queue, whereas, N_∞ denotes number of packets in the infinite buffer GI/M/1 queue. The queue length distribution of GI/M/1 queue is given by [2]:

$$P(N_\infty = j) = A\omega^{j-1} \qquad (j \geq 0) \qquad (2)$$

where A is the normalization constant and ω is a constant depending on the arrival process and service rate. ω can be obtained by solving the following equation:

$$\omega = \gamma\left((1-\omega)\mu\right) \qquad (3)$$

where $\gamma(s)$ is the Laplace transform of the arrival process and μ is the service rate which is given by $\frac{B}{\bar{X}}$. In order to solve (3) for ω, we need to assume a inter-arrival time distribution for the arrival process. Let us consider the Generalized Exponential (GE) distribution for modelling the arrival process to first two moments. We discuss below the reason for choice of GE distribution.

The pdf of GE distribution is given by

$$g(x) = (1-p)\delta(x) + pae^{-ax} \qquad (4)$$

where $\delta(x)$ is the delta function, p and a two constant parameters. As can be seen from (4), a GE process is characterized by two parameters, p and a. GE distribution is a special case of H_2 distribution and can be used to model general inter-arrival processes that are more bursty than Poisson process. For a Poisson process the variance is equal to the square of mean. Hence, GE distribution may be used to model the first two moments of processes with variance greater than the square of mean. If the arrival process is represented by a GE distribution, then, with probability p the inter-arrival time is exponentially distributed with mean a and with probability $1-p$, the inter-arrival time is zero. Hence, this distribution represents a batch arrival process with geometrically distributed batch size and exponentially distributed inter-batch arrival times. For a link with λ, σ as its mean and variance of the offered load, we can have the parameters of the GE distribution representing the arrival process:

$$p = \frac{2\lambda^2}{\sigma^2 + \lambda^2} \text{ and } a = p\lambda \qquad (5)$$

The merging of N independent GE(p_i,a_i) processes is a bulk-arrival Poisson process with mean arrival rate a equal to $\sum_{i=1}^{N} a_i$ and p equal to $a/\sum \frac{a_i}{p_i}$. Similarly, splitting of a GE(p,a) process into N streams according to a Bernoulli filter $r_1, r_2, ...r_N$, the parameters of the i^{th} process are

$$p_i = \frac{p}{p(1-r_i) + r_i} \text{ and } a_i = r_i a. \qquad (6)$$

Reader may refer to [12], Section 1.4 for more details.

The packet arrival process of a single TCP flow is bursty in nature with a "bulk" of packets arriving every round-trip time. The model that we have considered implies that we have "bulk" arrivals (in form of bursts of packets from competing TCP sources) of varying sizes arriving into a queue. Our model does not capture the feedback effect of packet drops on TCP flows because we have considered the aggregate traffic arriving at an OSPF router as our demand estimate.

Taking the Laplace transform of (4), we get,

$$G(s) = 1 - p + \frac{pa}{s+a} \tag{7}$$

Then substitute it into (3) and solve it for ω for the GE arrival process gives

$$\omega = \rho + (1-p) \tag{8}$$

where,

$$\rho = \frac{a}{\mu} = \frac{a\bar{X}}{B}. \tag{9}$$

Finally, using (1), (2), (3) and (7), we get the packet drop probability

$$P = \frac{(p-\rho)(\rho+1-p)^K}{1-(\rho+1-p)^{K+1}} \tag{10}$$

In summary, (10) represents the closed form expression of packet drop probability, P, on a single link as a function of mean, variance λ, σ^2 of the arrival process, mean packet size $\bar{X}$, link bandwidth B and buffer space K.

B. The Optimal General Routing

Using link packet drop probabilities obtained from (10), we can formulate the optimal general routing problem as:

$$\Phi = \sum_{l \in \mathcal{L}} \lambda_l P_l \tag{11}$$

where λ_l is the arrival rate for link l and P_l is its drop rate calculated by (10). This is a constrained optimization problem with the flow constraints at each router j for each demand $\mathcal{D}(s,t)$ between source s and destination t. If $f_l^{(s,t)}$ denotes the fraction of the demand $\mathcal{D}(s,t)$ on link l, then the flow balance constraints are given by

$$\sum_{i:(i,j)\in\mathcal{L}} f_{(i,j)}^{(s,t)} - \sum_{i:(j,i)\in\mathcal{L}} f_{(j,i)}^{(s,t)} = \begin{cases} -\mathcal{D}(s,t) & \text{if } j = s \\ \mathcal{D}(s,t) & \text{if } j = t \\ 0 & \text{Otherwise} \end{cases} \tag{12}$$

The mean packet arrival rate to a link l, λ_l, is given by

$$\lambda_l = \sum_{(s,t)\in\mathcal{N}\times\mathcal{N}} f_l^{(s,t)} \tag{13}$$

The parameter $p^{(s,t)}$ for the GE process used to fit the demand $\mathcal{D}(s,t)$ is given according to (5):

$$p^{(s,t)} = \frac{2\mathcal{D}(s,t)^2}{\mathcal{D}(s,t)^2 + \mathcal{V}(s,t)} \tag{14}$$

Let $r_l^{(s,t)}$ denote the probability with which the demand $\mathcal{D}(s,t)$ is sent on link l. Then $r_l^{(s,t)}$ is given by

$$r_l^{(s,t)} = \frac{f_l^{(s,t)}}{\mathcal{D}(s,t)} \tag{15}$$

Let $p_l^{(s,t)}$ denote the parameter p of the GE process after splitting the demand $\mathcal{D}(s,t)$ with probability $r_l^{(s,t)}$. Then $p_l^{(s,t)}$ denotes the parameter p of the GE process representing the flow $f_l^{(s,t)}$. The parameter $p_l^{(s,t)}$ is given according to (6):

$$p_l^{(s,t)} = \frac{p^{(s,t)}}{p^{(s,t)}(1-r_l^{(s,t)}) + r_l^{(s,t)}} \tag{16}$$

The total offered load on link l is given by λ_l (13), the parameter p of the associated GE distribution may be obtained by merging the flows $f_l^{(s,t)}$ going through l. If p_l denotes the parameter p of the GE process associated with the aggregate traffic on link l, then p_l is given by

$$p_l = \lambda_l \Big(\sum_{(s,t)\in\mathcal{N}\times\mathcal{N}} f_l^{(s,t)} p_l^{(s,t)} \Big)^{-1} \tag{17}$$

If ρ_l is equal to $\frac{\lambda_l p_l \bar{X}}{B_l}$, then, using (10), the probability of packet dropped at link l is given by

$$P_l = \frac{(p_l-\rho_l)(\rho_l+1-p_l)^{K_l}}{1-(\rho_l+1-p_l)^{K_l+1}} \tag{18}$$

The optimal general routing problem is given by (11), subject to the constraints given by (13), (14), (15), (16), (17), (18). It may be noted that we are casting the traffic according to the routing in order to obtain the mean and variance of the total offered traffic to each $l \in \mathcal{L}$. Equilibrium parameters of the arrival process to a link can be obtained by assuming an initial value of drop probability P_l^0. The actual arrival process parameters, taking into account the packet loss, may be obtained by splitting the aggregate arrival process with probability (P_l)(or P_l^0 in the first iteration). This may be iterated till the convergence of flows is achieved to the desired accuracy. In the results shown in this paper, we have not iterated to obtain the equilibrium traffic parameters. Essentially, we are using the upper bound on the packet drop probability as value of P_l and a higher λ_l in (11).

III. Optimization of OSPF weights Using On-line Simulation

The general optimal routing problem, where the objective function is completely defined by (11)-(18), may possibly be solved for $f_l^{(s,t)} \forall l \in \mathcal{L}$ by using some non-linear programming techniques. However, under constraints of OSPF routing, the relation between the link weights and optimization metric can no longer be analytically defined. Hence, the optimal routing in OSPF becomes a "black box" optimization problem which may be defined as:

$$\min \Phi(\mathbf{w}) \tag{19}$$

where $\mathbf{w}$ is the vector of network link weights and $\Phi(\cdot)$ the objective function, which is unknown. Basically, in order to obtain the value of Φ for a given OSPF weight setting, we run modified Floyd Warshall's algorithm (modified to obtain equal cost paths also) to obtain the routing. Then the traffic is cast to obtain parameters of the aggregate packet arrival process and drop probability for every link $l \in \mathcal{L}$ using (13), (14), (15), (16), (17) and (18). Finally the value of Φ may be calculated by (11).

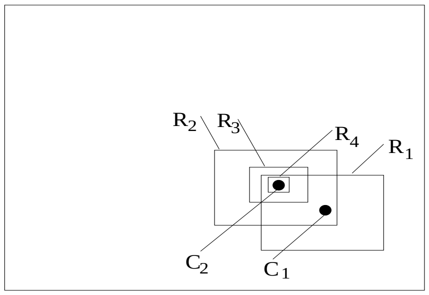


Fig. 2. Shrink and re-align procedure of Recursive Random Search

Finding optimal OSPF link weights is an NP-hard problem even for a linear objective function [4]. OLS uses RRS algorithm to obtain an optimal or near optimal link weight setting for the optimization problem given by (19).

In the context of network optimization, a highly efficient search algorithm is needed to find "good" OSPF link weight setting since the network is a dynamic system and network conditions may change significantly from time to time. Furthermore the search algorithm should be *scalable to high-dimensional problems* since there may be hundreds of parameters in a network. Another issue that needs to be considered is that network simulation only provides an approximate estimation of network performance. This means that the objective function is superimposed with small random noises due to inaccuracies in network modelling, simulation, etc. To address these issues, OLS uses a recursive random search scheme (see [15] for details and performance study of RRS). The RRS is based on the high-efficiency feature of random sampling at initial steps. The basic idea of RRS is to use the first part of high-efficiency samples to identify the promising areas. The recursive random sampling processes is used in these areas which are shrunk and re-aligned to obtain local optima.

An sample search using RRS is illustrated in Figure 2. First a number of random samples, say n, are taken from the parameter space D, and the best point is taken as the center C_1 of the promising region R_1 which is further explored. The point C_1 falls in $A_D(r)$, $r = 1 - (1-p)^{1/n}$ with probability p.The size of R_1 is taken to be the size of $A_D(r)$ so as to cover at least one local optimum in $A_D(r)$ with high probability. Then another l random samples are taken from R_1. Here l should be much less than n since the search is in a promising area and expects to find better points quickly. If a better point is found within these l samples, the center of the sample space is moved to this point and the size is kept unchanged. As shown in Figure 2, the center is moved to C_2, the region R_2 is used as the next sample space. If a better point is not found in l samples, the size of sample space is reduced by half and the center is kept unchanged. As shown in Figure 2, R_3 is used as the next sample space after l unsuccessful samples in R_2, and the center C_2 is left unchanged. This shrink-and-re-align procedure is repeated until the size of the region is reduced below a threshold, then the above search process is restarted.

Figure 3 shows the functional block diagram of the overall setup of this simulation. The OLS monitors the traffic to provide the estimates of mean and variance of the traffic demand for performance evaluation of link weights. A GE model parameters are used to fit the first two moments of link traffic. Dijkstra's algorithm (modified to include ECMP) is used to find the OSPF paths and traffic is cast using split and merging of GE flows. The aggregate traffic is used to compute the packet loss for a given OSPF link weight setting. RRS is then be used to search for better link weight setting for the network which is evaluated using the above procedure. When a certain stopping criteria is met, for example, the time limit is reached, the best-so-far link weight setting found by RRS may be deployed in the real network if it results in substantial improvement in the performance otherwise the link weights are left unchanged. A long search time suggests a near-optimal link weight setting with high probability.

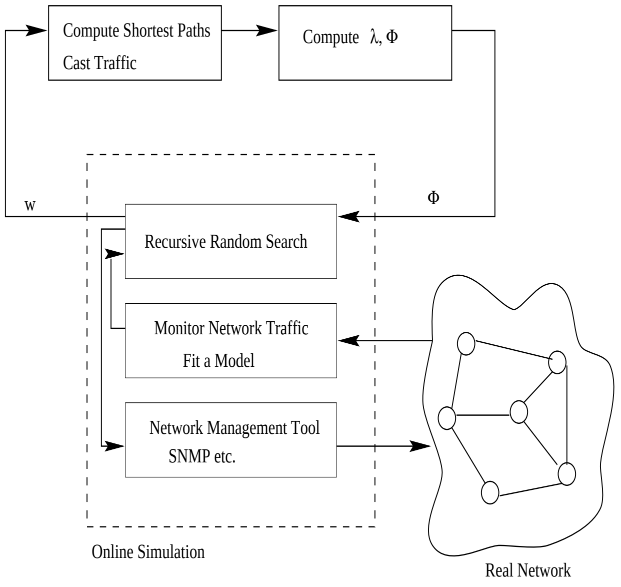


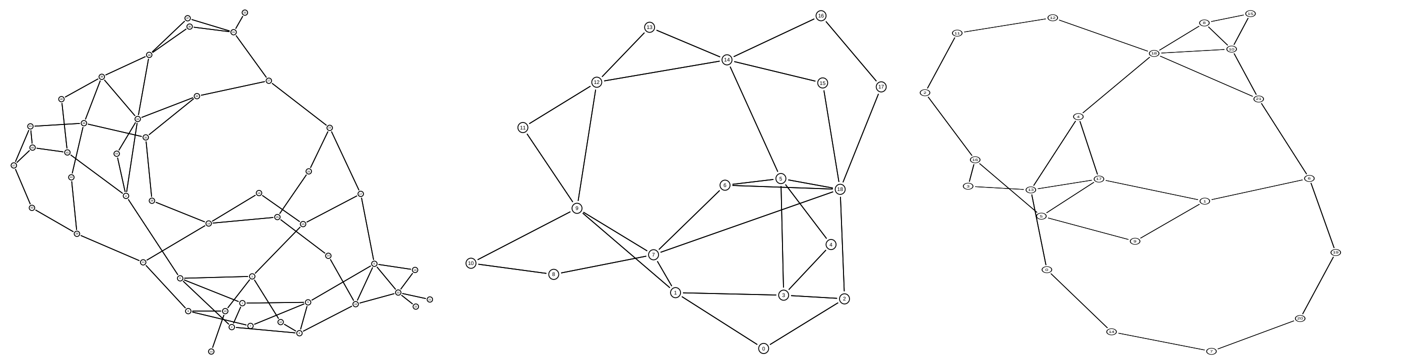
Fig. 3. Overall OSPF optimization setup using on-line simulation architecture

IV. Simulation Results

In this section we present two sets of simulation results. One is to demonstrate that the recursive random search scheme obtains better OSPF link weight settings with fewer function evaluations than the algorithm proposed in [3]. Another set of results demonstrate the improvement in end-to-end performance (in terms of the drop rate) by dynamic optimization of OSPF weights.

We have considered three network topologies, shown in Figure 4, to demonstrate our results. Two are well-known ARPANET topology and MCI topology. We couldn't include AT&T topology used in [3] for comparison as it is not publicly available (the authors could not disclose the topology). The ARPANET topology consists of 48 routers and 140 simplex links Each link in the network is assumed to consist of two simplex link whose weights may be set independently. MCI topology consists of 19 routers and 62 simplex links. We have also considered a randomly generated topology with 22 routers and 60 simplex links.

Random amount of traffic was sent from every node to every other node in the network. This random traffic was generated using the method outlined in [3]. For each node u, two random



(a) ARPANET Topology (b) MCI Topology (c) Randomly Generated Topology

Fig. 4. Figure showing the network topologies used in simulation

numbers are generated $O_u, D_u \in [0,1]$. For each pair of nodes (u,v) another random number $C_{(u,v)} \in [0,1]$ was generated. If Δ denotes the largest Euclidian distance between any pair of nodes and if α denotes a constant, the average demand between u and v is given by

$$\mathcal{D}(u,v) = \alpha O_u D_v C_{(u,v)} e^{\frac{-\delta(u,v)}{2\Delta}}$$

where, $\delta(u,v)$ denotes the Euclidian distance between the nodes u and v. This method of generating random traffic (the term $e^{\frac{-\delta(u,v)}{2\Delta}}$) ensures more traffic for source destination pairs that are closer to each other. Since a product of three random variables is taken to generate the demands, there is actually a large variation in the traffic demands. The ratio of square of mean to the variance was assumed to be a uniformly distributed random variable in $[0,1]$. The mean and variance of the traffic demands are generated using the above procedure. All the links in the network have 1Mbps bandwidth with a buffer size of 50 packets. The packet size was chosen to be exponentially distributed with mean packet size of 200 bytes.

In the simulation results presented in this paper, we do not verify the traffic modelling assumptions as this is not a focus of this paper. The performance results shown in IV-A are the average results from ten simulation runs. Average of multiple simulation runs is presented as we compare the performance of two stochastic search algorithms.

A. Comparison of Search Schemes

In this section, we present the results of comparison of recursive random search scheme with the local search scheme proposed in [3]. In optimization literature, the comparison between algorithms is usually done in terms of the number of function evaluations instead of the absolute time taken to find a "good" parameter setting. This is because the computation time is considerably dependent on many other factors, such as, implementation efficiency, testing platform, compiler, etc.. Assuming that the main computation time is for function evaluations, the number of function evaluations is a more appropriate performance metric under the assumption that the computation time per function evaluation is approximately the same for both schemes. Note this assumption is not exactly true in the context of our problem, where one function evaluation represents one optimization metric computation for a specific set of link weights. In [3], authors have used incremental shortest path computations to improve the speed of search as very few link weights change from one iteration to the next which is reported to have 15% improvements on an average. In spite of this, we still use the number of function evaluations as our algorithm performance metric for the reasons mentioned above and the consideration that our algorithm is designed to be a general "black-box" search algorithm where no problem-specific is available. It should be noted that even after taking the 15% improvement for the local search scheme of [4] into consideration, our algorithm is significantly faster (please see the results).

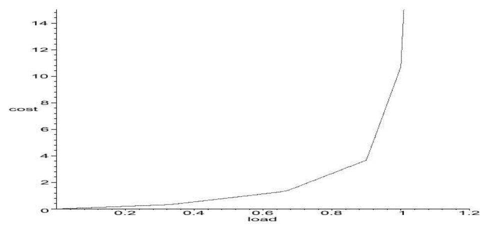
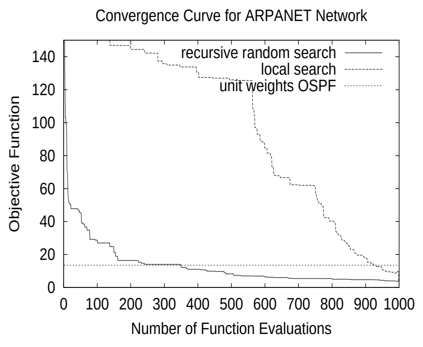


Fig. 5. Figure showing the link cost as a function of offered load

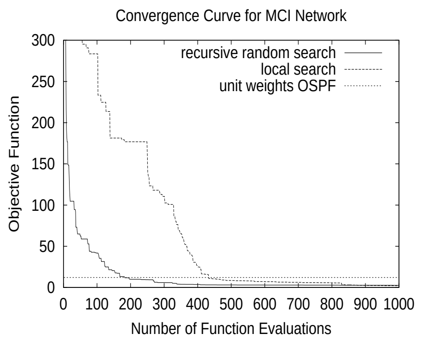
Loosely, we refer to the number of function evaluations required to obtain a "good" parameter setting as the speed of convergence. A "good" parameter setting has been defined as the OSPF link weight setting that give metric value lower than that by setting all link weights equal to unity (called unit OSPF). This definition is just for the purpose of comparison. A "good" parameter setting may have been defined alternatively as the link weight setting to achieve performance metric equal to, say, 80% of the unit OSPF.

1) Heuristic Piecewise Linear Metric: In order to compare the speed of convergence of our search scheme with the local search scheme proposed in [3], we use the same metric used in [3], which is piecewise linear with the link offered load. Figure 5 shows the cost for one link as a function of offered load. The optimization objective is to minimize the sum of link costs, summed over all $l \in \mathcal{L}$.

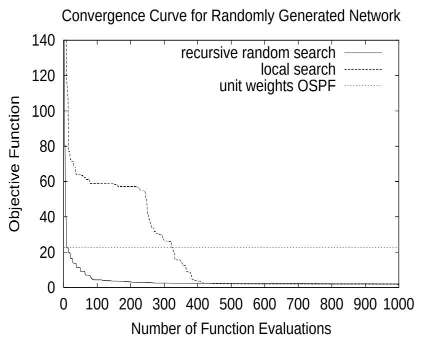
Figure 6 shows the optimization convergence curves for the ARPANET, MCI and Randomly generated network topologies respectively. For the sake of comparison, these graphs also show the optimization metric value when all the links' weights



(a)



(b)



(c)

Fig. 6. Figure showing the convergence curves of piecewise linear metric for (a) ARPANET (b) MCI (c) Randomly generated network topology

are set to unity. It can be seen that the recursive random search scheme outperforms the local search scheme in terms of the number of function evaluations needed to find a "good" parameter setting for all three network topologies. These results have been tabulated in Table I.

Scheme	ARPANET	MCI	Random
Local Search	932	433	322
RRS	350	183	9
Improvement	62.4%	57.7%	97.2%

TABLE I

TABLE COMPARING THE NUMBER OF FUNCTION EVALUATIONS NEEDED TO OBTAIN A "GOOD" PARAMETER SETTING FOR PIECEWISE LINEAR METRIC

2) Packet Drop Rate Metric: In this section we present the comparative results for the packet drop metric defined in (18). Figure 7 shows the comparison results of the optimization convergence speed. The results clearly show that the recursive random algorithm significantly outperforms the local search algorithm. Table II shows that for the packet drop rate metric, our recursive random search scheme took 70% or fewer function evaluations to obtain a "good" OSPF link weight setting.

Scheme	ARPANET	MCI	Random
Local Search	882	469	372
RRS	210	125	54
Improvement	76.1%	73.3%	85.5%

TABLE II

TABLE COMPARING THE NUMBER OF FUNCTION EVALUATIONS NEEDED TO OBTAIN A "GOOD" PARAMETER SETTING PACKET DROP RATE METRIC

B. Optimizing OSPF for improving packet drop rate

Now we describe the simulation showing how the network performance can be improved by our OSPF optimization scheme. Figure 3 shows the overall simulation set-up.

We used the network simulator ns [11] to simulate the real network running OSPF. The traffic in the network was generated in the same way as outlined in the beginning of this section. However, every 200 seconds the traffic pattern (the mean and variance of demand matrix) was changed in order to create a dynamic scenario. The traffic generator is implemented over UDP to generate bursty traffic with the GE inter-arrival distribution described in (4). In our simulation, we assume OLS has a complete knowledge of necessary network information, such as, traffic demands, network topology, etc.. Whenever a change of traffic pattern happens, the OLS runs the recursive random search for a certain iterations to obtain a better parameter setting. If the optimized setting is much better than the original, it will be deployed at 100 seconds after the traffic change. The 100-seconds time difference is used because we want to observe the performance difference between before optimization and after optimization. Note that here we assume the running time of the search algorithm is faster than the traffic change period, i.e., the search algorithm has finished running at 100 seconds after the traffic change.

The actual packet drop rates are collected during the simulation for all the traffic sinks in the network and then summed together to get the total packet drop rate. Figure 8 shows total packet drop rate in the network as a function of time. Table III summarizes the maximum improvement in packet drop rates for different topologies. Note that more or less improvements may result depending on the topology and traffic conditions.

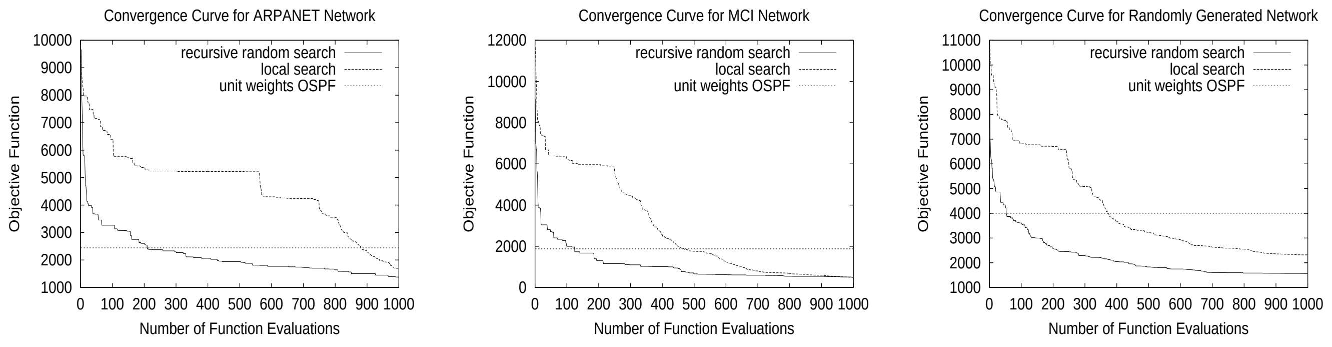
	ARPANET	MCI	Random
Max. Improvement	31.8%	60.2%	35.7%

TABLE III

TABLE SUMMARIZING THE MAXIMUM PERCENTAGE IMPROVEMENT IN THE PACKET DROP RATES OBTAINED FOR DIFFERENT TOPOLOGIES FOR THE RESULTS SHOWN IN FIGURE 8

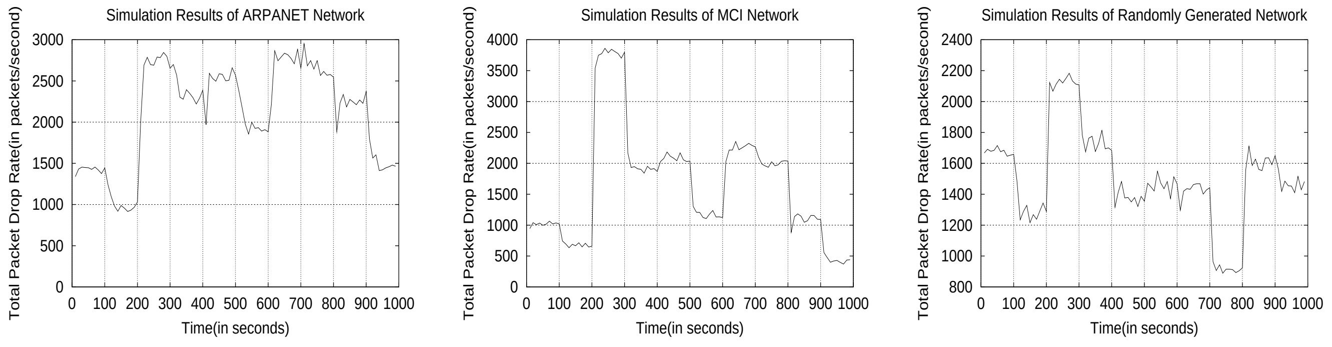
V. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a scheme for minimizing packet loss in the network by optimizing OSPF weights using Online simulation framework. The optimization problem was formulated. A general inter-arrival GI/M/1/K queuing model was used to compute the packet loss rate in the network. A GE process was used to find closed-form expression which is general enough to fit a bursty arrival process to two moments. OLS uses a fast, scalable recursive random search (RRS) scheme to



(a) (b) (c)

Fig. 7. Figure showing the convergence curve of total packet drop rate for (a) ARPANET (b) MCI (c) Randomly generated network topology



(a) (b) (c)

Fig. 8. Figure showing total packet drop rate as a function of time for the (a) ARPANET (b) MCI (c) Randomly generated network topology. Traffic pattern was changed at times 0, 200, 400..., the optimized OSPF weights were deployed at times 100, 300,...

search the parameter space. The search algorithm has been compared with the local search heuristic of [3] in terms of the number of function evaluations needed to obtain a "good" OSPF link weight setting. Our results demonstrate that the RRS takes 50-90% fewer function evaluations to find a "good" setting. The amount of improvement depends on the network topology, traffic conditions and optimization metric. The simulation results demonstrated improvements of the order of 30-60% in the total drop rate in the network for the different topologies considered.

Future work includes demonstration of the proposed scheme in a real test network. Validating the GE model for traffic, packet loss probability approximation and developing models for Random Early Drop (RED) and more complex buffering strategies are topics for future work. Investigating issues associated with traffic monitoring and modelling and its impact on the performance of dynamic optimization will be another goal for future work.

References

[1] Dimitri P. Bertsekas, "Dynamic Models of Shortest Path Routing Algorithms for Communication Networks with Multiple Destinations," *Proceedings of 1979 IEEE Conference on Decision and Control*, Ft. Lauderdale, FL, pp. 127-133, Dec. 1979.

[2] Robert B. Cooper, "Introduction to Queueing Theory," Second Ed. New York : North Holland, 1981.

[3] Bernard Fortz and Mikkel Thorup, "Internet Traffic Engineering by Optimizing OSPF Weights, in *Proceedings of the INFOCOM 2000*, pp. 519-528, 2000.

[4] Bernard Fortz and Mikkel Thorup, "Increasing Internet Capacity Using Local Search," Preprint, *http://smg.ulb.ac.be/Preprints/Fortz00_21.html*, 2000.

[5] David W. Glazer and Carl Tropper, "A New Metric for Dynamic Routing Algorithms," *IEEE Transactions on Communications*, Vol. 38 No.3 March 1990.

[6] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford and F. True, "Deriving traffic demands for operational IP networks: methodology and experience," *IEEE/ACM Transaction on Networking*, pp. 265-278, June 2001.

[7] A. Feldmann, A. Greenberg, C. Lund, N. Reingold and J. Rexford, "Netscope: traffic engineering for IP networks," *IEEE Network Magazine*, special issue on Internet traffic engineering, pp. 11-19, March/April 2000.

[8] Atul Khanna and John Zinky, "The Revised ARPANET Routing Metric," *Proceedings of the ACM SIGCOMM*, pp. 45-56, 1989.

[9] Ramesh Nagarajan, James F. Kurose, and Don Towsley, "Approximation techniques for computing packet loss in finite-buffered voice multiplexers," *IEEE J.Select.Areas Commun.* , 9(3):368–377, April 1991.

[10] J. Moy, "OSPF Version 2," RFC 2178, April 1998.

[11] (1997) NS-2(*network simulator*) http://www-mash.cs.berkeley.edu/ns.

[12] Harry G. Perros Queueing Networks With Blocking, Exact and Approximate Solutions, *Oxford University Press*, 1994.

[13] Zheng Wang, Jon Crowcroft, "Analysis of Shortest-Path Routing Algorithms in a Dynamic Network Environment," *Computer Communication Review*, Vol. 22, no. 2, pp.63-71, 1992.

[14] T. Ye, D. Harrison, B. Mo, B. Sikdar, H. T. Kaur, S. Kalyanaraman, B. Szymanski and K. S. Vastola, "Network Management and Control Using Collaborative On-line Simulation," *In Proceedings of IEEE ICC,* Helsinki, Finland, June 2001.

[15] T. Ye, S. Kalyanaraman, "A Recursive Random Search Algorithm for Optimization Network Protocol Parameters," Technical report, ECSE Department, Rensselaer Polytechnic Institute, 2001.