A New Approach for MPLS Traffic Engineering

I. Widjaja, I. Saniee, A. Elwalid and D. Mitra

Bell Laboratories, Lucent Technologies

Murray Hill, NJ 07974, USA

Abstract— **We propose an approach, called design-based routing (DBR), whereby optimized paths computed offline are used to guide online label swicthed path setups. By means of simulation, we show performance improvement of DBR compared to standard SPF and CSPF under static and dynamic connections with different protection modes. The results emphasize the robustness of DBR even in the face of substantial uncertainty in traffic demand estimate.**

Fig. 1. Path setup process.

I. INTRODUCTION

A Traffic Engineering (TE) system for connection provisioning in a (G)MPLS-based network involves three components: (1) resource discovery and distribution, (2) path design/selection (working and protection), and (3) path setup. Although the first and third components are standardized, the second component is left as an implementation-dependent module. Existing implementations of path design widely use shortest path first (SPF) and constraint SPF (CSPF). We propose a hybrid online-offline approach for path selection and show via simulation that the approach outperforms the widely used schemes.

Our approach takes into account some information about the traffic demand between each pair of nodes. We investigate the impact of the accuracy of the traffic demand information on the performance of the proposed scheme under a variety of conditions. In particular, we consider *static connections* where established connections stay permanently, and *dynamic connections* where connections arrive and depart at random. We also consider survivable networks with different protection modes: (1) no protection, (2) dedicated path protection, and (3) shared path protection. To the best of our knowledge, path design based on imprecise or uncertain traffic demand information has not been quantified.

II. DBR MODEL AND COMPONENTS

Fig. 1 shows the basic model of the path setup process of the proposed scheme, hereforth referred to as Design-Based Routing (DBR). The DBR server automatically collects the network topology and link capacities via a link-state protocol, and collects and estimates the traffic demand. Based on this information, the DBR server computes the optimized paths offline and stores the results in a database for future use.
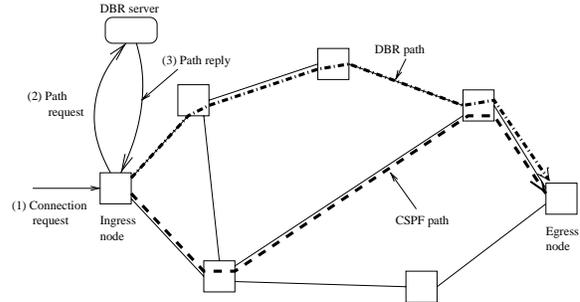
In this model, a connection request triggers a node (typically the ingress node) to issue a *Path Request* message to the server, which immediately replies with an explicit path(s) that is already pre-computed. We note that DBR can also be extended to the case where the path computation process is distributed to all ingress nodes.

III. PATH DESIGN WITH DBR

A. Definitions and Notations

We consider a network topology represented by a graph $G(V, E)$ where V denotes the set of vertices (nodes) and E denotes the set of edges (links). Each edge $e \in E$ has a capacity of C_e and a length of l_e. The length may be a function of actual distance or other salient parameters used in the optimization criteria. Our interest in this paper is to focus on a transport network, where the edges and the connections are typically bi-directional. Furthermore, we assume that each connection is a label switched path (LSP) with integral multiples of a fixed capacity.

Let K denote the set of all node pairs in the network. For static connections, the traffic demand d_k is expressed in terms of the total number of connections that need to be routed for node pair $k \in K$. For dynamic connections, the traffic demand δ_k is the offered load in Erlangs for node pair $k \in K$. Let P_k denote the set of all paths for node pair k, and let P denote the set of all paths for all node pairs; that is, $P = \bigcup_k P_k$. The output of path design determines the amount of demand (flow) that needs to be assigned to each path. Let the quantity x_p represents the amount of demand, in predefined fixed units, assigned to path $p \in P$.

Finally, let Q_e denote the set of all paths that traverse edge e. The subsequent sections present DBR formulations for the case with no protection.

B. DBR with Static Demand Information

The optimal path design in DBR minimizes the total bandwidth-length product. Furthermore, the optimization problem is subject to meeting the designated end-to-end demands while not exceeding the edge capacities. If all connections cannot be established in the existing network due to constraint by the edge capacities, the problem needs to be solved in two stages. In stage 1, the demands are appropriately re-scaled to fit into the existing network. We leave out the stage-1 formulation in this paper for brevity.

The stage-2 optimization problem is as follows:

Q1$(\lambda * \mathbf{length})$

$$min \sum_{e \in E} \lambda_e l_e \tag{1}$$

subject to

$$\sum_{p \in P_k} x_p \geq d_k, \quad \forall k \in K \tag{2}$$

$$\sum_{p \in Q_e} x_p \leq \lambda_e, \quad \forall e \in E \tag{3}$$

$$\lambda_e \leq C_e, \quad \forall e \in E \tag{4}$$

$$x_p \geq 0, \quad \forall p \in P \tag{5}$$

The output of the optimization problem Q1 provides the assignments of demands to paths for each node pair (i.e., x_p). It is readily seen that when the edge capacities are arbitrarily large, the effect of performing this optimization is simply to let each node pair use the shortest path. In this case, no concurrent optimization is necessary. Indeed, Q1 which is minimization of the total bandwidth-length product subject to (edge) capacity constraints can be thought of as the simplest non-trivial extension of the shortest path routing when there are capacity constraints on the edges.

The above formulation assumes exact end-to-end traffic demand information. However, DBR does not generally require accurate traffic demand information. To explore this issue further, we assume that $d_k \in \mathcal{N}(\bar{d}_k, \sigma_k)$ and reformulate Q1 when end-to-end demands are no longer fixed but are random variables. In this case, we propose to meet each end-to-end demand (d_k) with a given probability ($1 - \epsilon_k$). We can then formulate DBR with inaccurate demand estimate as follows:

Q2$(\lambda * \mathbf{length})$

$$min \sum_{e \in E} \lambda_e l_e \tag{6}$$

subject to

$$Pr\{\sum_{p \in P_k} x_p \geq d_k\} \geq 1 - \epsilon_k, \quad \forall k \in K \tag{7}$$

$$\sum_{p \in Q_e} x_p \leq \lambda_e, \quad \forall e \in E \tag{8}$$

$$\lambda_e \leq C_e, \quad \forall e \in E \tag{9}$$

$$x_p \geq 0, \quad \forall p \in P \tag{10}$$

As in $Q1$, the above stochastic optimization problem provides x_p, the flow assignment to each path p. There are several ways to solve Q2, of which the most direct solution is as follows. Define η_k such that

$$erf(\eta_k) = 1 - \epsilon_k \tag{11}$$

where

$$erf(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-y^2}{2}} dy \tag{12}$$

Then the constraint set (7) can be rewritten as

$$\sum_{p \in P_k} x_p \geq \bar{d}_k + \eta_k \sigma_k \tag{13}$$

Thus this procedure transforms $Q2$ to the deterministic optimization problem by substituting Eq. 2 in Q1 with Eq. 13.

C. DBR with Dynamic Connections

When connections are dynamic, there is always a chance of blocking even with generous dimensioning of the paths. In this case, the DBR objective is to minimize network resource usage subject to meeting the traffic demand up to a given level of blocking probability. Variants of this problem have been considered in [1] where three approaches were proposed: complete partitioning, complete sharing and virtual partitioning. For simplicity and clarity of the presentation, we only discuss complete partitioning in this paper. However, DBR is naturally extensible to all three approaches.

Assume that connection requests between node pair k arrive according to a Poisson process with offered load δ_k Erlangs, and a logical path with capacity $L_k (= \sum_{p \in P_k} x_p)$ units is assigned for node pair k. Then the end-to-end blocking probability for node pair k is given by the Erlang B formula

$$B(\delta_k, L_k) = \frac{\delta_k / L_k!}{\sum_{i=0}^{N_k} \delta_k / i!} \tag{14}$$

The optimization procedure for DBR with dynamic connections can be formulated as follows:

Q3$(\lambda * \mathbf{length})$

$$min \sum_{e \in E} \lambda_e l_e \tag{15}$$

subject to

$$B(\delta_k, L_k) \le \beta, \quad \forall k \in K \tag{16}$$

$$\sum_{p \in P_k} x_p \ge L_k, \quad \forall k \in K \tag{17}$$

$$\sum_{p \in Q_e} x_p \le \lambda_e, \quad \forall e \in E \tag{18}$$

$$\lambda_e \le C_e, \quad \forall e \in E \tag{19}$$

$$x_p \ge 0, \quad \forall p \in P \tag{20}$$

Note that the non-linear constraint (17) can be eliminated by first solving for L_k. In this case, the optimization problem can be simplified by using the following procedure:

Q4($\lambda * $length)
1. Compute $L_k^* = \text{argmin}_{L_k}\{B(\delta_k, L_k) \le \beta\}, \forall k \in K$.
2. Solve Q1 with $d_k \leftarrow L_k^*, \forall k \in K$ (using pre-processing P1 if necessary).

The formulation of DBR with protection (dedicated or shared) follows similar approaches presented in the preceding discussions. For details of these formulations and the computations of link capacities for the case of accurate static demand, see [2].

IV. PERFORMANCE INVESTIGATION

We assume a transport network that supports dynamic routing (e.g., OSPF-TE) to distribute link-state information and a signaling protocol (e.g., RSVP-TE) to setup and teardown connections on demands. We consider static and dynamic connection models. In the static model, connection requests arrive one-by-one. If the connection is successfully routed, it will stay indefinitely (i.e., the holding time is infinite). In the dynamic model, connection requests arrive at random and the holding time of a connection is finite. The dynamic model may be more suited when the transport network is optical and the clients are IP networks that use bandwidth on demand.

The transport network may be operated under different protection scenarios depending on service requirements. We consider three path protection scenarios:
- no protection,
- 1+1 or 1:1 dedicated path protection, and
- 1:1 shared path protection.

A. Network Model

In our simulation experiment, we consider the topology of a generic U.S. transport network, as shown in Fig. 2. The network has 30 nodes and 38 links with each link assumed to be bi-directional. For static model, the input to the simulator is driven by the traffic demand matrix $D = [d_{ij}]$, for each $i, j \in V$ (note that the end-to-end demand is previously defined between a pair of nodes $k = (i, j)$). Note that the demand is symmetric; that is $d_{ij} = d_{ji}$. For dynamic model, $D = [\delta_{ij}]$. In the subsequent simulation results, we consider the practical case where the network has been designed so that the link capacities are optimized (e.g., via Q1).
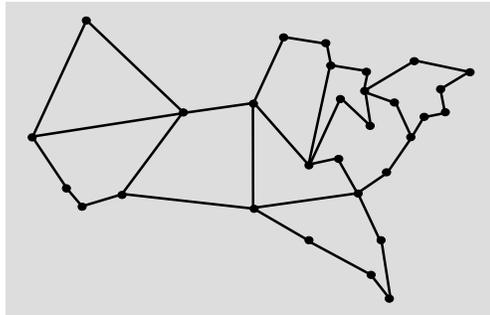

Fig. 2. Network topology.

B. No Protection

We investigate the case where connections are not protected. We first consider the static connection model where the traffic demand matrix is assumed to be accurate for DBR. The network is initially unloaded, and a total of $\sum_{i,j} d_{ij} = 3495$ connection requests are to be provisioned. To emulate the online operational model, we assume that connection requests arrive one by one in a given order.

Each request is provisioned until all requests have been processed. The performance measure of interest here is the total number of connections that are successfully established (routed). Because the order of requests determines the number of connections that can be successfully established, we perform a number of trials where each trail corresponds to a given permutation of requests. Fig. 3 compares the performance of three path computation approaches (SPF, CSPF, and DBR). In this example, DBR is able to route all connection requests. As expected, CSPF performs better than SPF. CSPF can route about 80% of all requests, while SPF can route about 75% of them.

Next, we consider the case where the traffic demand matrix used in DBR can be inaccurate. To model the inaccuracies, we independently perturb each original traffic demand, d_k, by a gaussian noise $\mathcal{N}(0, \sigma)$, properly truncated and rounded, so that the value of the perturbed traffic demand is a non-negative integer. The degree of inaccuracies in the traffic demand is determined by the coefficient of variation $c = \sigma_k/d_k$. The larger the value of the coefficient of variation, the worse the original traffic demand estimate is.

We compare the performance of DBR relative to another path computation approach by a measure called *competi-*
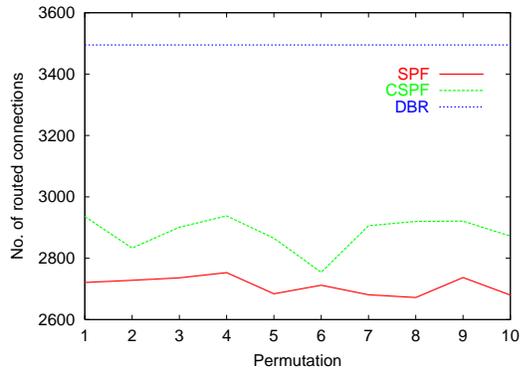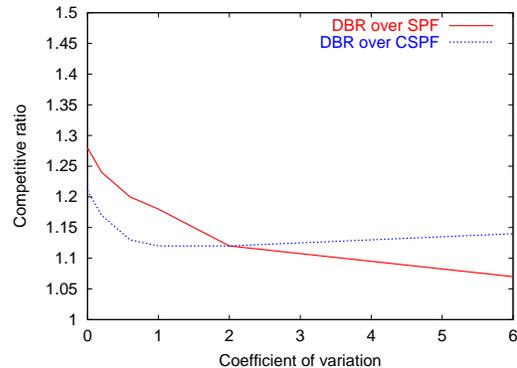
Fig. 3. Static case with perfect traffic demand.

tive ratio adopted from [3]. Let $N(X)$ denote the number of connections that are successfully routed via approach X for a given order of requests. Then, the competitive ratio is

$$CR(X) = \frac{E[N(DBR)]}{E[N(X)]} \qquad (21)$$

Fig. 4(a) plots the competitive ratio against the coefficient of variation for SPF and CSPF. Here, the expected value is obtained by averaging the results of 500 independent trials, where each trial corresponds to a particular order of requests. The point where $c = 0$ corresponds to the case where the traffic demand is perfectly known a priori. This ideal operating point reveals that DBR can route 20% more connections than CSPF, and about 25% more connections than SPF. As expected, the advantage of DBR diminishes as the traffic demand estimate becomes less accurate (i.e., c increases). However, notice that DBR still outperforms SPF and CSPF even when c becomes very large, indicating the robustness of DBR against errors in traffic demand estimates. Notice an interesting case where SPF outperforms CSPF as c is greater than 2. The reason is that the total number of connection requests increases when c becomes large. This in turn causes CSPF to greedily choose a longer path in an attempt to satisfy the current request without regard to any possible future requests.

It is well known trunk reservation may be applied to two-hop paths to reduce resource usage. The benefits of trunk reservation in the context of CSPF paths is illustrated in Fig. 4(b), where R indicates the trunk reservation level. If the number of available bandwidth units divided by the total capacity on a particular link along the CSPF path that is different from the SPF path is less than R, then a new connection request is rejected. $R = 0$ corresponds to the case where no trunk reservation is applied. As R increases, CSPF becomes less greedy. But when R becomes too large, CSPF becomes too conservative. Notice that $R = 5$ appears effective in combating against greed

(a)
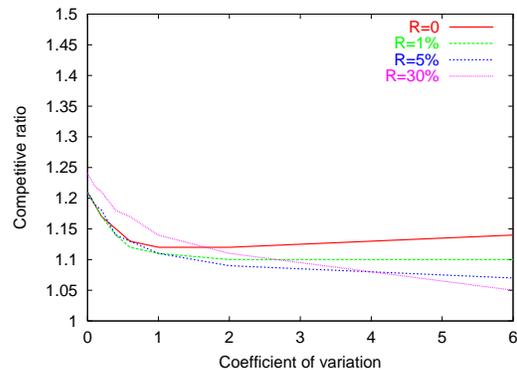
(b)

Fig. 4. Static case with inaccurate traffic demand: (a) without trunk reservation, (b) with trunk reservation.
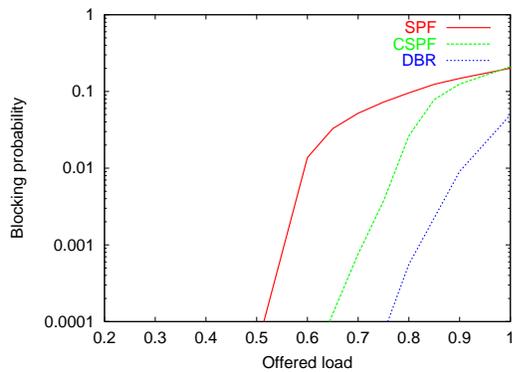
while remains aggressive under normal loading.

We now consider the dynamic model without protection. We assume that connection requests arrive according to a Poisson process, and that the holding time of a connection is exponentially distributed. Fig. 5(a) compares SPF, CSPF, and DBR with respect to the path blocking probability as the normalized total offered load is varied. As can be seen from the figure, DBR reduces the blocking probability by at least an order of magnitude less than others under moderate load. Fig. 5(b) shows the corresponding curves under heavy load. Observe that SPF outperforms CSPF when the network is congested.

C. Protection

We now turn our attention to the scenario where the tranport network employs path protection for each node pair. Path protection requires a path computation algorithm for two disjoint paths between each node pair. To this end, we adopt the disjoint-path computation algorithm due to [4] for SPF and CSPF.

With dedicated protection, the protection path is allocated with the same amount of bandwidth as the corresponding working path.

The performance results with dedicated protection for
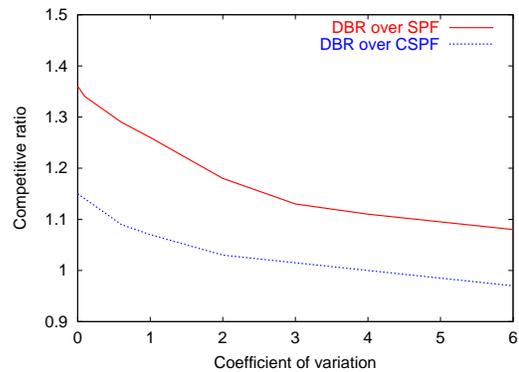
(a)
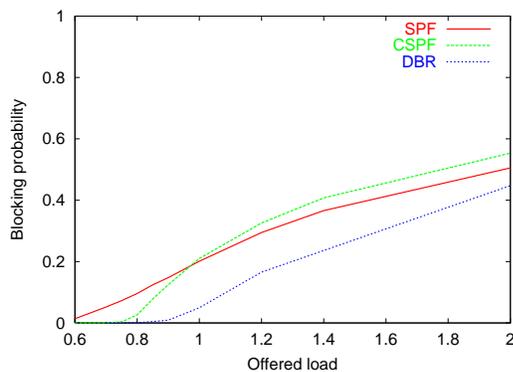
(b)

Fig. 5. Dynamic case: (a) Moderate load, (b) heavy load.

(a)

(b)

Fig. 6. Dedicated protection: (a) Static case with inaccurate traffic demand, (b) dynamic case.

static and dynamic models are displayed in Fig. 6. As expected, DBR again outperforms CSPF which in turn outperforms SPF. Looking into more detail, we note two interesting observations. First, comparing Fig. 6 with Fig. 4(a), observe that CSPF continues to outperform SPF even in congested state under dedicated protection. This is attributed to the fact that CSPF is less able to choose long disjoint paths than long single paths. Second, observe that CSPF slightly outperforms DBR when the coefficient of variation is very high. This is because CSPF has a richer set of two-disjoint paths, and can take advantage of this fact without incurring the penalty for being too greedy.
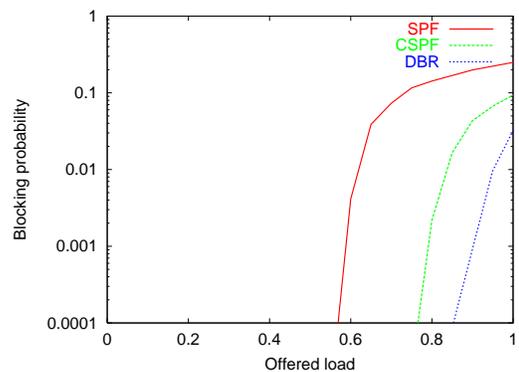
For shared protection, our simulation results show improved benefits due to DBR compared to dedicated protection.

V. CONCLUSION

We have presented an approach, called design-based routing (DBR), that utilizes a server which collects relevant data, computes paths offline, and responds to requests for routes. We showed that DBR is superior to widely deployed online path computation approaches such as SPF and CSPF under static or dynamic connections with or without protection. Our results indicate that load information, even in the presence of uncertainties and inaccu-

racies, can be utilized to improve the performance of an online path setup mechanism.

REFERENCES

[1] D. Mitra and I. Ziedins, "Virtual Partitioning by Dynamic Priorities: Fair and Efficient Resource-Sharing by Several Services," in *Proc. 1996 International Zurich Seminar on Digital Communications*, Lecture Notes in Computer Science, Broadband Communications, B. Plattner (editor), Springer, 1996, pp. 173-185.

[2] R. D. Davis, K. Kumaran, G. Liu, and I. Saniee, "SPIDER: A Simple and Flexible Tool for Design and Provisioning of Protected Lightpaths in Optical Networks," *Bell Labs Technical Journal*, vol.6, no.1, Jun. 2001.

[3] S. Plotkin, "Competitive Routing of Virtual Circuits in ATM Networks," *IEEE Selected Areas in Communications*, vol.13, no.6, pp. 1128-1136, Aug. 1995.

[4] J. W. Suurballe, "Disjoint Paths in a Network," *Networks*, pp. 125-145, 1974.