Oblivious Routing Scheme Using Load Balancing Over Shortest Paths

Marija Antić, Aleksandra Smiljanić
Faculty of Electrical Engineering, Belgrade, Serbia
m.antic@ieee.org, aleks@ieee.org

***Abstract*—In this paper, we propose the oblivious routing scheme based on shortest-path routing and load balancing. We present the LP model that finds the optimal routing. Then we compare the performance of the proposed scheme with the performance of the shortest-path routing for some regular and real-case network topologies. We show that the proposed routing strategy allows to achieve higher guaranteed traffic from/to a node in the network, compared to the case of the classical shortest-path routing.**

I. Introduction

The choice of the routing strategy has a large impact on the overall network performance. It is very important to choose a routing scheme that optimizes the utilization of the network resources. Bad routing decisions can put very high loads on some links, thereby seriously lowering the throughput. On the other hand, the appropriately chosen routing strategy can make the most of the given network resources, allowing it to achieve the maximum possible throughput.

If the network topology and the exact traffic demands are known, the problem of determining the optimal routing can be solved by a linear program (LP). However, it is practically impossible to measure and predict the actual traffic demands. Even if it was possible to do so, it would still take time to solve the optimization problem, and frequent changes in the routing strategy could lead to service disruptions.

The common approach to solving the routing problem is the *oblivious routing* design. Generally, a routing is considered to be oblivious if routing decisions for a particular flow are made independently of other traffic demands. In other words, the path or the set of possible paths are predetermined for every source-destination pair, and routing decisions are made by the originating nodes based only on the traffic destination. The problem of finding the optimal oblivious routing for the changing source-destination traffic demands has been studied by many authors. A non-polynomial solution for general networks, based on graph decomposition, was proposed in [8], and was later followed by the polynomial-time algorithm in [9], [10]. In [3], [4] linear programming was used to find the optimal routing strategy.

A different approach was introduced by Kodialam et al. in [6], [7]. It is much easier to estimate the total incoming/outgoing traffic for a particular node, than to predict the actual traffic distribution. In fact, we always know the upper limit for the node traffic - it cannot be higher than the sum of link capacities entering or leaving the node. Kodialam et al. introduce the two-phase routing scheme, based on load balancing. The concept of load balancing allows them to calculate the source-destination traffic demands based on the incoming/outgoing node traffic. Then they solve the general linear programming problem of routing coefficients optimization. In [6], a linear program for the capacity minimization was presented, and in [7], the comparisson of the performance of the proposed routing strategy versus the performance of the shortest-path routing was done for one network topology and four different traffic matrices.

In this paper, we propose the routing strategy that is similar to the one presented by Kodialam et al. We use load balancing and every flow is routed in two stages. The difference is that we use the shortest-path routing as the underlying routing scheme. This allows us to significantly simplify the LP model - we only optimize the node weights, since the routing coefficients are already determined. Our model, therefore, has only $O(N)$ variables, compared to $O(N^2M)$ in [6] (where N and M are the number of nodes and links in the network, respectively). The main advantage of the proposed strategy is the simplicity of its implementation. The shortest-path routing is most widely used, and the proposed scheme represents a modification that could easily be brought to life. We compare the performance of the proposed strategy with the performance of the classical shortest-path routing, and show that the proposed strategy can support significantly higher traffic demands.

II. Preliminaries

In the process of network designing and planing, it is important for the network operators to estimate the maximum number of users that can be served by a network node in every circumstance. When the user bit-rates are known, this problem is equivalent to determining the *maximum admissible node traffic* , i.e. the maximum incoming/outgoing node traffic that can be routed through the network, regardless of the actual traffic-pattern. In this paper we present the load balanced routing (LBR) scheme and compare it with the shortest-path routing (SPR). We calculate the maximum admissible node traffic for both LBR and SPR, and show that LBR allows larger values of the maximum admissible node traffic (and the larger number of users, consequently).

Assume that the network is represented by a directed graph $G = (V, E)$, where V is the set of nodes (vertices) and E is

This paper was partly supported by the Ministry of Science of Republic of Serbia grant for graduate students.

the set of links (edges). The number of nodes in the network will be denoted by N and the number of links by M.

Let i be a source node. The outgoing traffic generated by i equals $s_i = \sum_{j\in V} d_{ij}$, where d_{ij} is the intensity of a flow from i to j . Similarly, for j being the destination node and r_j its total incoming traffic, we have $r_j = \sum_{i\in V} d_{ij}$.

Matrix $\mathbf{TM} = [d_{ij}]_{N\times N}$ will be referred to as *traffic matrix*, and vectors $\mathbf{S} = [s_1, s_2, \ldots, s_N]$ and $\mathbf{R} = [r_1, r_2, \ldots, r_N]$ as *out-traffic vector* and *in-traffic vector*.

The elements d_{ij} of the traffic-matrix **TM** are hard to predict, especially as the peer-to-peer traffic is becoming dominant on the Internet. On the other hand, the in-traffic and out-traffic vectors can easily be estimated, based on the number of users attached to the node and their bit-rates. In our analysis we will focus on the backbone network. In this case, we can consider that the in-traffic and out-traffic vectors are equal ($\mathbf{S} = \mathbf{R}$). To simplify the analysis, we will assume that all the nodes have equal traffic demands, $s_i = s$, for $i = \overline{1, N}$. The same analysis can apply to the case with unequal node demands, if we represent every node by a set of fully connected basic nodes with equal demands.

The maximum admissible node traffic is determined by the network *congestion*. Congestion Q depends on the routing and the actual traffic-pattern, and is defined as the maximum link utilization in the network. Link utilization represents the ratio of the link load, $L(l)$, and link capacity, $C(l)$, i.e. $U(l) = L(l)/C(l)$. Thus, $Q = \max_{l\in E} U(l)$. The link with the highest utilization is the first one to get overloaded, as the traffic demands increase. Every flow in the network can be increased up to $1/Q$ times, without causing the link overload. Therefore, minimizing the congestion maximizes the allowed traffic values.

It has been realized in practice and some regular networks that load balancing decreases the congestion. We use this concept in our load balanced routing (LBR) design, to increase the maximum admissible node traffic. The traffic from i to j is not sent directly. Instead, it is split in portions that are directed to intermediate nodes $m \in V$ first - Fig. 1. For all pairs (i, j), the portion of a flow d_{ij} that is balanced across a node m equals k_m. Obviously, $\sum_{m=1}^{N} k_m = 1$. In the next step every intermediate node m forwards the received traffic to its final destination j. Traffic from i to m and from m to j is routed along the shortest paths. We will show that this routing strategy increases the maximum admissible node traffic, compared with the case of classical shortest-path routing.

III. Performance Analysis

We will compare the performance of the proposed load balanced routing (LBR) with the performance of the shortest-path routing (SPR). We use the maximum admissible node traffic as the measure of the routing performance on a particular network. In this section, we describe the LP model used to calculate the optimal node weights k_m, for which the maximum admissible node traffic in the case of LBR is achieved. Then we describe the algorithm used to calculate the maximum admissible node traffic for SPR.

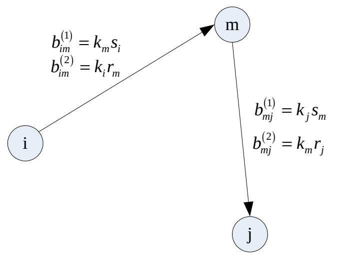

Fig. 1: Routing Scheme Illustration

A. Performance Analysis of the Load Balanced Routing (LBR)

We will determine the optimal values of node weights k_m to achieve the maximum admissible node traffic. As already mentioned, the maximum admissible node traffic is proportional to the inverse of congestion. In order to maximize the admissible traffic, we will actually solve the congestion minimization problem.

The traffic from i to j is not sent directly. It is first balanced across intermediate nodes $m \in V$. The traffic between nodes i and m consists of two components: load generated by i and balanced across m (denoted by $b_{im}^{(1)}$) and load for m balanced across i (denoted by $b_{im}^{(2)}$). Thus, $b_{im} = b_{im}^{(1)} + b_{im}^{(2)}$.

The traffic that a node i balances across m equals

$$b_{im}^{(1)} = \sum\nolimits_{j\in V} k_m d_{ij} = k_m s_i. \tag{1}$$

At the same time, the portion k_i of every flow d_{pm}, $p \in V$ is balanced across i. The traffic that i has to forward to m equals

$$b_{im}^{(2)} = \sum\nolimits_{p\in V} k_i d_{pm} = k_i r_m. \tag{2}$$

Let the variable $F_{im}^{(l)}$ take the value 1 if the link l belongs to the shortest path between the nodes i and m, and 0 otherwise. This variable is calculated based on the OSPF routing tables. For the link load $L(l)$, we have

$$L(l) = \sum\nolimits_{(i,m)} F_{im}^{(l)} \, (b_{im}^{(1)} + b_{im}^{(2)}). \tag{3}$$

Link utilization is given by the formula

$$U(l) = L(l)/C(l) = \sum\nolimits_{(i,m)} F_{im}^{(l)} \, (b_{im}^{(1)} + b_{im}^{(2)})/C(l). \tag{4}$$

The general-case formulation of the linear program that minimizes congestion has the form:

$$\begin{aligned}
&\min Q\\
&\text{(C1)}\ \sum\nolimits_{i=1}^{N} k_i = 1\\
&\text{(C2)}\ \forall l \in E:\ \frac{\sum_{(i,m)} F_{im}^{(l)}\,(k_i r_m + k_m s_i)}{C(l)} \le Q\\
&\text{(C3)}\ Q \le 1\\
&\text{(C4)}\ \forall n \in V:\ \sum_{l\in \mathrm{IN}(n)} L(l) - \sum_{l\in \mathrm{OUT}(n)} L(l) = r_n - s_n.
\end{aligned} \tag{5}$$

Sets $\textsc{in}(n)$ and $\textsc{out}(n)$ represent the set of incoming and outgoing links of a node n, respectively. This problem has $N+1$ variables - $k_1, k_2, \ldots, k_N$ and Q. The number of constraints is $M + N + 2$ (not counting non-negativity constraints).

We will prove that if for every node the incoming equals the outgoing traffic, the flow conservation constraints (C4) are superfluous.

Lemma 1: If $r_n = s_n$ the constraint (C4) is redundant.

Proof: When $r_n = s_n$, after substituting $L(l)$ from Eq. 3, constraint (C4) becomes

$$\sum_{\substack{(i,m)\\ l\in \mathrm{IN}(n)}} F_{im}^{(l)}(k_i s_m + k_m s_i) = \sum_{\substack{(i,m)\\ l\in \mathrm{OUT}(n)}} F_{im}^{(l)}(k_i s_m + k_m s_i). \tag{6}$$

For $i \neq n, m \neq n$, node n can either be on the shortest path between i and m or not. If it is not on this path, then there is no link $l \in \mathrm{IN}(n)$ or $l \in \mathrm{OUT}(n)$ such that $F_{im}^{(l)} = 1$, and the term $k_i s_m + k_m s_i$ does not appear in the equation. If n is on the shortest path, then there is exactly one link entering and one leaving this node with $F_{im}^{(l)} = 1$, so the term $k_i s_m + k_m s_i$ appears on both sides of the equation, and can be eliminated. This leaves only the terms associated with $i = n$ or $m = n$.

Let us now consider the case when $i = n$, i.e. n is the source node. For any m, there can be no link $l \in \mathrm{IN}(n)$ such that $F_{im}^{(l)} = F_{nm}^{(l)} = 1$ (otherwise the path from n to m would contain a loop, and would not be the shortest path). Similarly, if $m = n$, i.e. n is the destination node, there can be no link $l \in \mathrm{OUT}(n)$, such that $F_{im}^{(l)} = F_{in}^{(l)} = 1$.

Thus, we can rewrite the Eq. 6 in form:

$$\sum_{\substack{i\\ l\in \mathrm{IN}(n)}} F_{in}^{(l)}(k_i s_n + k_n s_i) = \sum_{\substack{m\\ l\in \mathrm{OUT}(n)}} F_{nm}^{(l)}(k_n s_m + k_m s_n). \tag{7}$$

If the network is connected, there is a path between every pair of nodes. For any i there is exactly one link $l \in \mathrm{IN}(n)$ such that $F_{in}^{(l)} = 1$, so for every i the term $k_i s_n + k_n s_i$ appears exactly once on the left hand side of the Eq. 7. Similarly, for any m there is exactly one link $l \in \mathrm{OUT}(n)$, such that $F_{nm}^{(l)} = 1$. For every m, the term $k_n s_m + k_m s_n$ appears exactly once on the right hand side. Equation 7 always holds, because it holds $\sum_{i\in V} k_n s_i = \sum_{m\in V} k_n s_m$ and $\sum_{i\in V} k_i s_n = \sum_{m\in V} k_m s_n = s_n$. Thus, the initial constraint is always fulfilled and is therefore not necessary to add it to the model. ■

As said before, we consider that all the nodes have equal traffic demands $s_i = s$. If we normalize $s = 1$, we have the following form of the linear program:

$$\begin{aligned} &\min Q \\ &\text{(C1)}\ \ \textstyle\sum_{i=1}^{N} k_i = 1 \\ &\text{(C2)}\ \ \forall l \in E : \frac{\sum_{(i,m)} F_{im}^{(l)}\,(k_i + k_m)}{C(l)} \leq Q \\ &\text{(C3)}\ \ Q \leq 1 \end{aligned} \tag{8}$$

This model now has $N+1$ variables and $M+2$ constraints. The optimal (minimum) value Q implies the maximum admissible node traffic for LBR

$$s_{max}^{lbr} = 1/Q, \tag{9}$$

since all the traffic demands can be increased up to $1/Q$ times without causing the overload of the most highly utilized link.

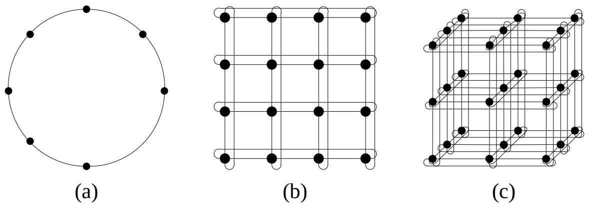

Fig. 2: Regular topologies. (a) Ring. (b) Manhattan. (c) Torus.

B. *Performance of the Shortest Path Routing (SPR)*

We will determine the maximum admissible node traffic for SPR. For every link, we find the maximum matching of the sources and the destinations using the link. Based on the size of the matching, we calculate the highest node traffic allowed on that link. The lowest value of the allowed node traffic among all the links in the network represents the maximum admissible node traffic. In this subsection we describe the algorithm in more detail.

Link load depends on the number of source-destination pairs that communicate across the link. Denote the set of all node pairs that use the link l as $P(l) = \left\{(i,j)|F_{ij}^{(l)} = 1\right\}$. Node i sends the traffic $t_i(l) = \sum_{j|(i,j)\in P(l)} d_{ij}$ across l. Traffic-matrix is admissible if $\sum_i t_i(l) \leq C(l)$ for all $l \in E$.

Observe link $l \in E$. Define the set of sources sending their traffic across l, $I(l) = \{i|\exists j, (i,j) \in P(l)\}$, and the set of destinations receiving the traffic across l, $J(l) = \{j|\exists i, (i,j) \in P(l)\}$. We suppose that all the nodes $i \in V$ have equal, symmetrical traffic demands, $s_i = r_i = 1$. Traffic-matrices that put the highest load on the observed link are those with $t_i(l) = s_i = 1$, for all $i \in I(l)$.

Let us form a bipartite graph with $I(l)$ and $J(l)$ as the sets of nodes. Let there be an edge in the bipartite graph between every pair of nodes $(i,j) \in P(l)$ (note the difference between these edges and the links in the network, they are not related). According to the Birkhoff-von Neumann theorem and [12], in the worst case every node $i \in I(l)$ sends the traffic $s = 1$, and every node $j \in J(l)$ can receive at most $r = 1$. The maximum load of the link l in this case equals the size of the maximum matching for the bipartite graph, $p(l)$. The link utilization is $U(l) = p(l)/C(l)$, and implies the maximum node traffic allowed on link l to be $a(l) = C(l)/p(l)$. We assume that all the nodes in the network have equal traffic demands, so this restriction applies to all of them, and not only to the nodes included in the maximum matching.

The maximum admissible node traffic for the whole network is $s_{max}^{spr} = \min_{l\in E} a(l)$. We calculate it using the algorithm:

- Step 1: (Initialization) Set $s_{max}^{spr} = \infty$ and $X = E$.
- Step 2: Take $l \in X$. Determine $I(l)$ and $J(l)$. Form a bipartite graph, with $I(l)$ and $J(l)$ as the sets of nodes, and edges between pairs $(i,j) \in P(l)$.
- Step 3: Find the maximum matching for the bipartite graph. Let $p(l)$ denote the number of matched pairs.
- Step 4: Calculate $a(l) = C(l)/p(l)$.

- Step 5: If $a(l) < s^{spr}_{max}$, let $s^{spr}_{max} = a(l)$.
- Step 6: $X = X \setminus \{l\}$. If $X \neq \emptyset$, go to Step 2; else stop.

IV. Results

We compare the maximum admissible node traffic for LBR and SPR. In particular, we determine the performance gain

$$G = s^{lbr}_{max} / s^{spr}_{max}, \tag{10}$$

for some regular topologies and examples of real networks. The value s^{lbr}_{max} is determined using the linear program (8), while s^{spr}_{max} is determined as described in Section III-B.

A. Regular Topology Networks

In this section we analyze the performance of the proposed routing strategy for some regular network topologies: ring, Manhattan, torus and full-mesh - Fig. 2. In some of these topologies there are multiple shortest paths between two nodes. We analyze two cases of the underlying shortest-path routing: the case when the traffic is routed along the axes of a grid in a predetermined order, and the case used in practice, when the path is chosen based on the next hop node ID number.

Let us observe LBR in the case of the underlying shortest-path routing along the axes. We suppose that all the nodes have equal traffic demands $s_i = s = 1$. Due to the symmetry, all links are equally loaded, and the optimal coefficients k_m are equal, $k_m = 1/N$. Based on Eq. 1 and Eq. 2, the flow between two nodes in the network equals

$$b = b^{(1)}_{ij} + b^{(2)}_{ij} = 2s/N = 2/N. \tag{11}$$

1) Ring: The average path length of a flow in the ring network in the case of routing along the axes equals $l_{av} = \sum_{i=1}^{\lfloor N/2 \rfloor} (2i/N) \approx N/4$. Since the number of directed links is $2N$, and there are N^2 node pairs, the expected link load in LBR case is $L^{lbr}(l) = l_{av} b N^2 / 2N \approx N/4$, (Eq. 11). Therefore, the maximum admissible node traffic $s^{lbr}_{max} = 4C/N$ (Eq. 9). For SPR, the worst case is when only the pairs of the most distant nodes communicate, and exactly $\lfloor N/2 \rfloor$ flows are routed across every link. This implies the maximum allowed node traffic $s^{spr}_{max} = 2C/N$. Therefore, the gain is $G = 2$.

The gain is plotted in Fig. 3a. The dashed line represents the above derived result, and the full line the results in the case when node enumeration affects the choice of the path.

2) Manhattan: Manhattan has the form of two-dimensional grid, with rows and columns circularly connected - Fig. 2b. In a symmetrical Manhattan, both dimensions of the grid are equal (D), while in a non-symmetrical case they differ.

Let us analyze the case of routing along the axes in a symmetrical Manhattan, with $N = D^2$ nodes. The average length of the path in each of the directions is $l_{av} \approx D/4$. Since the total number of horizontal (vertical) links is $2N$, the expected load of a link for LBR equals $L^{lbr}(l) = l_{av} b N^2 / 2N \approx D/4$, (Eq. 11). This implies $s^{lbr}_{max} = 4C/D$. In the worst-case scenario for SPR, only pairs of the most distant nodes communicate. For the presumed routing strategy, the maximum number of node pairs that can use one link in the network equals $D/2$. Thus, $s^{spr}_{max} = 2C/D$, and $G = 2$.

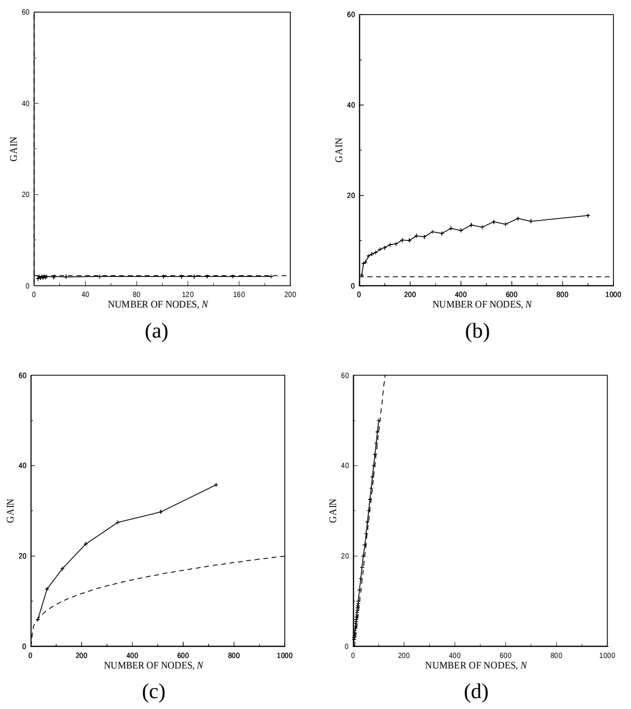

Fig. 3: Gain for regular network topologies. (a) Ring. (b) Symmetrical Manhattan. (c) Torus. (d) Full-mesh.

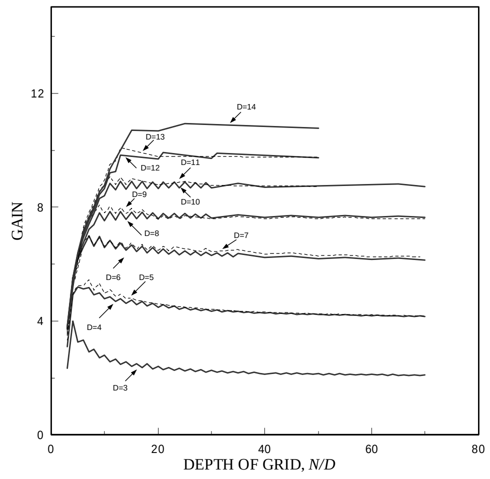

Fig. 4: Gain for a non-symmetrical Manhattan network. One side of the grid is D.

The gain for the symmetrical Manhattan is plotted in Fig. 3b. The dashed line represents the results in case of the routing along the axes, and the full line in case of the routing based on the node enumeration. We can observe that the gain in the second case is significantly higher than 2 and increases with the dimension of the grid. Fig. 4. represents the results for the non-symmetrical Manhattan, assuming the underlying shortest-path routing based on node enumeration. The gain depends predominantly on the shorter dimension of the grid; for larger values of the longer dimension, it remains constant.

TABLE I: Results for the six Rocketfuel network topologies

NETWORK	NODES	LINKS	GAIN	MODEL SIZE		TIME [seconds]			
				variables	constraints	load data	presolve	solver	total
Telstra (Australia) 1221	104	302	2.141	105	304	3.984	0.032	0.062	4.078
Sprintlink (US) 1239	315	1944	7.548	316	1923	51.015	0.282	4.750	56.047
Ebone (Europe) 1755	87	322	2.012	88	316	3.360	0.015	0.063	3.438
Tiscali (Europe) 3257	161	656	7.217	162	629	12.640	0.063	0.922	13.625
Exodus (US) 3967	79	294	6.057	80	282	2.734	0.016	0.062	2.812
Abovenet (US) 6461	138	744	4.576	139	720	9.843	0.063	1.047	10.953

3) Torus: Torus has the form of a three-dimensional circularly-connected grid - Fig. 2c. In a symmetrical case, all three sides of the grid are D. The average path length in each of the directions is $D/4$. As in the Manhattan network, the expected link load for LBR is $L^{lbr}(l) \approx D/4$, and $s^{lbr}_{max} \approx 4C/D$. For SPR, the maximum number of most distant node pairs that can communicate across one link equals $D^2/2$. This implies $s^{spr}_{max} = 2C/D^2$, and $G = 2D = 2\sqrt[3]{N}$.

The results are shown in Fig. 3c. Again, the full line represents the gain in case when the choice of the path depends on the enumeration of the nodes. In this case, the gain increases even faster with the number of nodes in the network.

4) Full-mesh: Each node in a full-mesh is connected with all the other nodes and has the degree of $N - 1$. The load of a link in the LBR case is $L^{lbr}(l) = 2/N$ (Eq. 11), and $s^{lbr}_{max} = CN/2$. In the case of SPR, only one flow is routed across each link, so $s^{spr}_{max} = C$. Therefore, $G = N/2$. The calculated gain is presented in Fig. 3d.

We can observe that load balancing produces gain even for the simplest topology such as ring. The gain increases with the degree of a node in the network, and is proportional to the number of nodes for the full-mesh topology.

B. Real Case Networks

Router-level ISP network topologies are usually considered confidential and are not publicly available. In Rocketfuel project [11], the authors made an effort to estimate the topologies of major US, European and Australian ISPs, based on routing information. We use their data for six backbone-level topologies. Since the original data provides only link weights, we assume that the capacity of a link is inversely-proportional to its weight.

We suppose that all nodes generate equal traffic $s_i = 1$. Results of the analysis are listed in Table 1, together with times needed to find the optimal values k_m for the LBR case.

The time that LP_Solve solver needs to find the optimal solution to a linear program is predominantly determined by the time needed to load data. This is actually the time that the algorithm takes to calculate link loads based on values $F^{(l)}_{ij}$ and add the constraints to the model. LP_Solve seems to have a problem with longer data loading times [13], so the optimization time could be improved by using another optimization tool.

The routing scheme with load balancing provides gain in guaranteed node traffic for all topologies, when compared with shortest-path routing. For AS 1221 (Telstra) and AS 1775 (Ebone), this gain is two. Australian network has the topology of a ring between major cities, and sparks elsewhere, so this only confirms the theoretical results. European network topology seems to be highly meshed, but the link weights indicate that the traffic is predominantly directed across a ring connecting major cities, which can explain the results. For the remaining four well meshed topologies, the significant gain in range 4.5-7.5 is achieved.

V. Conclusion

In this paper we showed that the guaranteed node traffic can be significantly increased when the load-balancing routing scheme is deployed. The gain that can be achieved for many topologies is significant, and grows with the degree of a node in the network. It is shown that gain is achieved for all major ISP topologies. The implementation of this routing scheme would make it possible to satisfy greater demands without changing the network topology. Since it is based on the OSPF algorithm, and a simple linear programming algorithm, the required network upgrade is of an acceptable complexity.

References

[1] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications,* Prentice Hall, 1993.

[2] J. Matoušek, B. Gärtner, *Understanding and Using Linear Programming*, Springer Verlag, 2007.

[3] D. Applegate and E. Cohen, ”Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs,” *Proc. of SIGCOMM '03*, 2003.

[4] Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Räcke, ”Optimal oblivious routing in polynomial time,” *Proc. of the 35th ACM Symposium on the Theory of Computing,* 2003.

[5] *LpSolve, Reference Guide* [Online]. Available:http://lpsolve.sourceforge.net, 2005.

[6] M. Kodialam, T. V. Lakshman, and S. Sengupta, ”Efficient and Robust Routing of Highly Variable Traffic,” *3rd Wksp. Hot Topics in Networks*, 2004.

[7] M. Kodialam, T. V. Lakshman, and S. Sengupta, ”Traffic-Oblivious Routing for Guaranteed Bandwidth Performance,” *Communications Magazine*, 45(4):46-51, Apr. 2007.

[8] H. Räcke, ”Minimizing congestion in general networks,” *FOCS 43*, 2002.

[9] C. Harrelson, K.Hildrum, and S. Rao, ”A Polynomial-time Tree Decomposition to Minimize Congestion,” in *Proc. of SPAA'03*, 2003.

[10] M. Bienkowski, M. Korzeniowski, and H. Räcke, ”A Practical Algorithm for Constructing Oblivious Routing Schemes”, *Proc. of SPAA'03*, 2003.

[11] N. Spring, R. Mahajan, and D. Wetherall, ”Measuring ISP topologies with Rocketfuel,” *Proc. of the ACM SIGCOMM02*, 2002.

[12] B. Towles and W. J. Dally, ”Worst-case Traffic for Oblivious Routing Functions,” *Computer Architecture Letters*, 1, Feb. 2002.

[13] S. R. Thorncraft, H. R. Outhred, and D. J. Clements, ”Evaluation of Open-Source LP Optimization Codes in Solving Electricity Spot Market Optimization Problems,” *19th Mini-Euro Conference on Operation Research Models and Methods in the Energy Sector,* Sept. 2006.