A Hop-by-hop Energy Efficient Distributed Routing Scheme

Chenying Hou
Institute of Computing Technology, CAS
University of Chinese Academy of Sciences
Beijing, China
houchenying@ict.ac.cn

Fa Zhang
Institute of Computing Technology
Beijing, China
zhangfa@ict.ac.cn

Antonio Fernández Anta
Institute IMDEA Networks
Madrid, Spain
antonio.fernandez@imdea.org

Lin Wang
Institute of Computing Technology, CAS
University of Chinese Academy of Sciences
Beijing, China
wanglin@ict.ac.cn

Zhiyong Liu
Institute of Computing Technology
Chinese State Key Lab for Computer Architecture, ICT,CAS
Beijing, China
zyliu@ict.ac.cn

ABSTRACT

Energy inefficiencies in current networks provide both challenges and opportunities for energy saving. Recently, there are many works focusing on minimizing energy cost from the routing perspective. However, most existing work view them as optimization problems and solve them in a centralized manner such as with a solver or using approximations.

In this paper, we focus on a network-wide bi-objective optimization problem, which simultaneously minimizes the total energy consumption and the total traffic delay using speed scaling. We propose a hop-by-hop dynamic distributed routing scheme for the implementation of this network-wide optimization problem. Our scheme is more practical to realize in large distributed networks compared with current centralized energy minimization methods. We can also achieve near global optimal in a distributed manner, while mostly used shortest path routing protocols such as OSPF cannot make it. Our routing scheme is totally distributed and maintains loop-free during every instant of routing. Simulations conducted in real world data sets show that the distributed loop-free routing scheme converges to the near Pareto optimal values. Also, our method outperforms the widely applied shortest path routing strategy with 30% energy saving.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Miscellaneous

General Terms

Theory, Performance

Keywords

distributed routing, energy efficient, bi-objective optimization

1. INTRODUCTION

With the widespread use of the Internet, energy conservation has become a major concern in networks over the past years. Current networks are not designed with energy optimization as an objective or a constraint. They are often over-provisioned and designed for peak traffic. These design strategies give us huge potential for energy saving in current networks. Studies show that usually the network bandwidth is rather underutilized and sometimes the network link utilization is even less than 5 percent of the time [13]. Speed scaling and powering down are explored as two promising energy saving mechanisms on network-device level. They are currently used in the industry [2], [13].

Recently, there are works focusing on energy saving at the network-wide level such as designing energy saving network routing protocols based on energy-aware network devices. Gupta et al. [9] firstly proposed the importance of considering energy saving from a network protocols' view. Andrews et al. [2], [1] have studied the global energy consumption minimization routing problem considering the speed scaling model and the powering down model under static traffic. Zhang et al. [19] developed GreenTE, an energy aware traffic engineering scheme by putting links into sleep mode while obeying the performance constraints. Similarly, Vasic et al. [16] proposed an online energy aware traffic engineering scheme based on source routing and they proposed energy-critical path as routing selection basis to achieve more energy saving [15]. Avallone et al. [4] designed a joint admission control and routing scheme in order to minimize active network devices while meeting the performance requirement. Some other network energy-efficient heuristic algorithms were proposed in [7], [14]. However, most existing works use centralized strategies or source routing for routing path

assignment, they usually go to a solver to get results or develop related heuristics strategies. These methods are not easy to implement especially in large distributed networks because of the impracticality of obtaining global information and high computation cost. They are also not flexible for packet to adjust path selection during routing according to the dynamic condition of links.

With the scale of networks keeping growing, compared with centralized strategies, a hop-by-hop distributed scheme is a more scalable and flexible choice. But how to achieve the same or near the same global energy optimization in a distributed way brings in some challenges. First of all, without knowing the whole network topology and traffic demands, each node makes their own decision of choosing working mode/speed and selecting which neighbor to forward a packet. Lacking global information, it is important to make sure that the individual operation will benefit the global energy saving performance rather than impair it. Another critical issue is that a distributed scheme should work correctly which means it is loop-free and can converge well without oscillations. Additionally, only considering energy saving will sacrifice the network performance such as bringing in congestion and intolerable delay. If increasing traffic aggregate into the link with low energy cost, it will cause congestion and degrade the network performance. So how to deal with the tradeoff between performance such as delay and energy efficiency is also necessary. The problem of multi-metric QoS routing have been widely investigated in the past years(e.g [18], [6] to cite a few). However, to the best of our knowledge, they did not consider energy as a metric and did not try fully distributed hop-by-hop strategy which can approach global optimal.

In this paper, we develop a distributed hop-by-hop energy-efficient routing scheme that simultaneously minimizes the global energy cost and the total traffic delay. We propose to leverage speed scaling as the hardware level support for energy saving. It dynamically adjust network devices' working speed according to the traffic load. Using speed scaling, we study how to direct traffic demands to flow through energy saving paths while optimizing its delay in a distributed manner. In order to maintain loop-freeness we propose a distributed multi-path finding algorithm as the preprocess of the routing. It provides nodes with next-hop-choice information without causing loops during routing. Our multi-path routing guarantees better load balancing compared with existing single loop-free path routing methods. Also, it has little communication among nodes and is easy to implement. By applying and extending the optimization theory in [8], our distributed scheme can quickly converge to the near global optimal with respect to energy saving and traffic delay.

The main contributions of this paper are the following:

First, we formalize the minimizing-energy consumption and delay problem in wired networks in a multi-criterion optimization framework that simultaneously optimizes the two objectives. Combining the two objectives by scalarization, we can get the Pareto optimal curve which is the optimal tradeoff between energy saving and packet delay.

Second, based on the formalization of the problem, we develop a distributed loop-free scheme to route traffic demands. Experiment results show that our algorithm can converge quickly and achieve near global optimal value.

Third, we consider dynamic traffic with bursts and develop a distributed algorithm to adapt to the dynamic change of the traffic. We show through experiments that our algorithm works rather well under dynamic traffic in comparison with theoretical optimal value.

The rest of the paper is organized as follows. In Section 2, we describe the network model and formulate the problem as a bi-objective optimization. In Section 3 we will introduce the loop-free distributed routing approach, including the distributed loop-free path finding algorithm and the optimal distributed routing. Section 4 presents the numerical results and Section 5 concludes the paper.

2. MODEL AND PROBLEM FORMULATION

We model a network topology as an undirected graph with bi-directional links $G = (N, E)$, N is the set of all nodes and E is the set of all links in the network topology. We assume that the traffic is injected into the network from source node i to destination node j with the expected rate r_j^i. Our goal is to route traffic on the network aiming at optimizing network performance measured as network energy consumption and traffic delay.

We assume that fractional routing is permitted which means that a routing demand can be splitted on multiple paths [2].(Since the packet reordering problem has been widely studied [17], we will not discuss it here.) Here, we use the same definition of variables as in [8]. Let t_j^i represent the total traffic from node i destined to node j. t_j^i is the sum of r_j^i and the traffic flowing to j through node i from other nodes. Let N^i denote the set of neighbors of node i. The routing variable ϕ_{jk}^i represents the proportion of traffic t_j^i allocated to edge (i, k), where node k is one of the neighbors of node i. From flow conservation, we have

$$t_j^i = r_j^i + \sum_{k \in N^i} t_j^k \times \phi_{ji}^k. \tag{1}$$

Let x_{ik} be the total expected traffic passing through link (i, k).Then,

$$x_{ik} = \sum_{j \in N} t_j^i \times \phi_{jk}^i. \tag{2}$$

Let CA_{ik} represents the capacity of link (i, k), the flow of each link cannot exceed the capacity of that link. That is,

$$0 \leq x_{ik} \leq CA_{ik}. \tag{3}$$

The routing variable ϕ_{jk}^i is nonnegative and satisfies the following constraints.

1. $\phi_{jk}^i = 0$ if $(i, k) \notin E$ or if $i = j$.
2. $\sum_{k \in N^i} \phi_{jk}^i = 1$.

The routing algorithm chooses paths for packets based on the weight of links. We define the weight of each link by a new cost function simultaneously considering energy cost and delay. We aim at minimizing both two metrics at the network-wide level which is a bi-objective optimization.

The energy function $f_{ik}(x_{ik})$ represents the energy cost of link (i, k) when transmitting x_{ik} units of traffic through it.(It actually contains energy cost of the link and related devices when transmitting packets through them.) The delay function $d_{ik}(x_{ik})$ is the delay of transmitting x_{ik} units of traffic on the link (i, k). Both two functions are computed from the expected traffic. By summing up the energy cost

of each link, we define F_T as the total energy cost of the network as $F_T = \sum_{(i,k)\in E} f_{ik}(x_{ik})$. And the total traffic delay D_T in the network is formulated as $D_T = \sum_{(i,k)\in E} d_{ik}(x_{ik})$.

we combine two objectives together in a scalarization way using weighted sum method. Weighted sum method provides the sufficient condition to generate optimal points in multi-objective optimization. We call this optimal points Pareto Optimal [12]. We give each objective function a weight to represent the preference on the objective. We define the final single objective C_T to be the total cost of the network as

$$C_T = w_1 \times F_T + w_2 \times D_T. \quad (4)$$

The sum of the weights of each objective equals to 1 as in (5), and both w_1 and w_2 are nonnegative.

$$w_1 + w_2 = 1. \quad (5)$$

A (w_1, w_2) pair corresponds to a Pareto Optimal point. With many (w_1, w_2) pairs, we can get a set of Pareto Optimal points.

We formalize this bi-objective optimization in the following way. Minimizing C_T is to find the Pareto Optimal points for minimizing F_T and D_T.

$$(P) \quad \text{Min } C_T$$

subject to

$$x_{ik} = \sum_{j\in N} t_j^i \times \phi_{jk}^i \ \ \forall i,k$$

$$t_j^i = r_j^i + \sum_{k\in N^i} t_j^k \times \phi_{ji}^k \ \ \forall i,j \ \text{(flow conservation)}$$

$$\sum_{k\in N^i} \phi_{jk}^i = 1 \ \ \forall i,j \ \text{(routing variable restriction)}$$

$$0 \le \phi_{jk}^i \le 1 \ \ \forall i,k,j \ \text{(capacity restriction)}$$

$$x_{ik} \le CA_{ik} \ \ \forall i,k \ \text{(routing variable restriction)}$$

When the traffic matrix is fixed, this optimization problem (6) can be solved by the optimization solver in a centralized way. However, centralized method needs the whole network topology and traffic matrix as the input and takes time to compute routes for all traffic demands and it may highly rely on central management and scheduling. To implement this complex computation is pretty hard especially in large distributed networks, it is therefore meaningful to realize such a bi-objective routing problem in a distributed manner. In next section, we will present a distributed routing scheme to solve this problem.

3. DISTRIBUTED LOOP-FREE ROUTING SCHEME

Our distributed loop-free routing aims to solve the bi-objective optimization (P). In order to achieve global optimal in a distributed way, we apply and extend the optimal routing conditions proposed by Gallager [8], which only be used to optimize network delay under quasi-static network traffic. Different from his work, we also develop a distributed multi-path finding strategy to avoid loop at every instance during routing. Additionally, we consider routing strategies under both static and dynamic traffic conditions.

Our loop-free routing scheme is totally distributed, which means each node only knows the local topology and local traffic information. Each node makes routing decision individually according to its routing table. A routing table records routing information for each destination including next hops and the traffic allocation proportion to each next hop. The main goal of our routing scheme is to construct a routing table at each node and update it according to link weight's changes. In this paper, the link weight is defined considering both energy cost and traffic delay.

Constructing the routing table involves two important issues. First is how to choose next hops at each node for traffic demands. Second is how to distribute traffic to its next hops. We propose a multi-path finding algorithm to generate next-hop information for each node and we guarantee that routing decision among these next hops will not cause looping. We also develop traffic allocation strategy based on optimal routing condition considering both static and dynamic traffic and finally approach the global optimal.

3.1 Distributed Multi-path Finding Algorithm

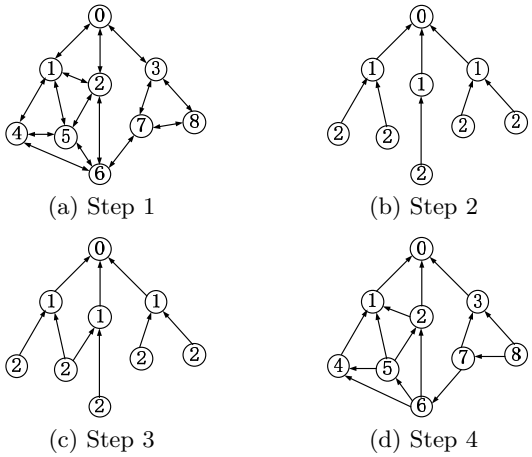
In the distributed routing process, a node must choose next hops for the packet passing through it. We need to find loop-free paths since it is obvious that paths with loops will cause the distributed routing algorithm not converge. Additionally, the energy cost of the route will increase if loop occurs because of the accumulative features of energy function. For a particular destination t, all multiple routing paths from every nodes in the network to t form a DAG (directed acyclic graph). To find next hops at each node for each destination which maintain loop-freeness is to generate a DAG for each destination including all nodes in the network topology.

Here we propose a "height-based" algorithm to generate a connected graph $GEN\_G(t)$ from $G(N, E)$ for each destination node t. Our method is inspired by the nature of water flow which always flows from a higher place to a lower one. The principle of our algorithm is that we assign each node a height, then if network traffic can flow from higher nodes to lower nodes based on the height, loops will never exist.

The method to generate $GEN\_G(t)$ is as follows: first, we generate a shortest path tree based on original graph G with root t as the destination. We use shortest path tree because it contains all nodes in the graph which can give all nodes a height. However, only considering the shortest path routing will not be energy-efficient [2]. Some links not included in the shortest path tree may have good energy saving performance. In order to include such kind of links, we try to construct a DAG having as many alternative links as possible. Our rule to further build the DAG is two steps. First, for the node pairs which are neighbors in the original graph and have different height in the shortest path tree, we reserve the link in the original graph from higher node to the lower node. Then, for the nodes pairs which are neighbors in the original graph and have equal height in the shortest path tree, we reserve the link from larger node to the smaller node according to the alphabetical order of nodes' indexes.

The shortest path trees for all destination can be built in a distributed way by using the distributed Bellman-Ford algorithm. We also observe that reserving links can be accomplished in a distributed way at each node. So the whole process of loop-free path finding method can be realized in a totally distributed manner. Fig. 1 presents an example for generating $GEN\_G(0)$. In Fig. 1(a) and Fig. 1(d) each

node is labeled with nodes' index and Fig. 1(b) and Fig. 1(c) each node is labeled with nodes' height in shortest path tree with node 0 as the root. Fig. 1(a) is a original graph, Fig. 1(b) is the shortest path tree generated from G, Fig. 1(c) presents reserving links by nodes' height and Fig. 1(d) is reserving links by alphabetical order of nodes' indexes.



(a) Step 1 (b) Step 2

(c) Step 3 (d) Step 4

Figure 1: Generating *GEN_G(0)*

we can proved that each *GEN_G(t)* is a *DAG* and it is also one of the largest *DAG* of G. Assuming there are loops in *GEN_G(t)* or there exists larger *DAG* will lead to contradiction, so the above statement can be proved. Due to the limited space, we do not present the detailed proof here.

3.2 Optimal Routing Condition

In this part, we will describe the sufficient and necessary conditions of minimizing C_T in network-wide. Gallager [8] proposed theorem of sufficient and necessary conditions to minimize total traffic delay in the network. We find that Gallager's theorem can be extended to apply to other cost functions satisfying certain characteristics and it is suitable for our problem. Different from Gallager's work, we consider simultaneously optimizing energy cost and delay. In this part, we will describe how to apply Gallager's optimal routing theorem to our bi-objective problem.

First, we will define some variables. According to the formulation of the bi-objective problem, we use $c_{ik}(x_{ik})$ to represent the weight of link (i,k). We define $c'_{ik}(x_{ik})$, the partial derivative of C_T with respect to x_{ik} as the marginal cost of link (i,k). For each node i, considering a destination j, the partial derivatives of the total cost C_T with respect to r_j^i is the marginal distance of node i destined for node j. We now give the formula of partial derivatives of the total cost C_T with respect to r_j^i and ϕ_{jk}^i as follows

$$\frac{\partial C_T}{\partial r_j^i} = \sum_{k \in N^i} \left[c'_{ik}(x_{ik}) + \frac{\partial C_T}{\partial r_j^k} \right] \tag{6}$$

$$\frac{\partial C_T}{\partial \phi_{jk}^i} = t_j^i \left[c'_{ik}(x_{ik}) + \frac{\partial C_T}{\partial r_j^k} \right]. \tag{7}$$

According to Gallager's theorem, we can get following sufficient and necessary condition for minimizing C_T.

THEOREM 1 ([8]). *The necessary condition for minimizing C_T with respect to ϕ for all $i \neq j$ and $(i,k) \in E$ is*

$$\frac{\partial C_T}{\partial \phi_{jk}^i} \begin{cases} = \lambda_{ij} & \phi_{jk}^i > 0 \\ \geq \lambda_{ij} & \phi_{jk}^i = 0. \end{cases} \tag{8}$$

THEOREM 2 ([8]). *The sufficient condition for a minimum of C_T with respect to ϕ for all $i \neq j$ and $(i,k) \in E$ is*

$$c'_{ik}(x_{ik}) + \frac{\partial C_T}{\partial r_j^k} \geq \frac{\partial C_T}{\partial r_j^i}. \tag{9}$$

The necessary and sufficient conditions provide a perfect load balancing leading to the global optimal. The perfect load balancing tends to increase the traffic variable associated with the neighbor node having small marginal distance and decrease those whose marginal distance is large. The final optimal condition will be arrived if for all nodes's neighbors, only the ones with smallest marginal distance value have traffic transmitted through them.

3.3 Distributed Routing Algorithm

The distributed routing algorithm is developed based on the necessary and sufficient conditions for optimal routing. Each node i in the network maintains a routing table RT_i. Each entry rt_j^i of the routing table RT_i records routing information for destination j including next hops to j, marginal distance from i to j and routing variables for next hops. Routing variables reflect the traffic allocation strategy among next hops. Calculating the routing variables is the key of our distributed routing algorithm.

The distributed routing algorithm runs iteratively to adjust traffic allocation and calculate routing variable according to optimal routing condition. The algorithm contains three main parts. First is the initialization. Second is the iterative updating routing variables. Third is the quiescence, which is the termination of iterative updating.

Let S_j^i represent the set of next hops through which router i forwards packets destined for j, and it is the neighbors of i in the *GEN_G(t)* generated in the loop-free path finding algorithm. Before routing, S_j^i has been fixed and recorded in the routing table. Let $D_j^i = \frac{\partial C_T}{\partial r_j^i}$ denote the marginal distance of node i to node j and $w_{ik} = c'_{ik}(x_{ik})$ denote the marginal cost of link (i,k). Let Φ_j^i denote the routing variable set of node i for destination j.

In the initialization part, for each destination j, each node i evenly distribute traffic by initializing routing variables ϕ_{jk}^i for all $k \in S_j^i$ with the same value and calculates its initial marginal distance D_j^i to each destination j. Then, node i records the above information into its routing table.

When updating, each node i iteratively recalculates the routing variable set Φ_j^i for every destination j and adjusts the traffic allocation towards the global optimal. One round of updating at node i for destination j is a mapping from old routing variable set Φ_j^i to a new routing variable set $(\Phi_j^i)^*$. According to the optimal routing condition (8, 9), each node i increases the routing variable ϕ_{jk}^i of the output link (i,k) whose value of $(D_j^k + w_{ik})$ is the smallest one in S_j^i, and decreases the routing variable ϕ_{jk}^i of the output link (i,k) with non-smallest $(D_j^k + w_{ik})$.

Then, based on the updated routing variable, each node i reallocates the traffic and recalculates D_j^i to each destination j. Then node i broadcasts D_j^i to all its neighbors. The

algorithm runs iteratively until the change of the routing variable set is less than the threshold after a new round of update, then the updating is terminated.

Now we will discuss the stop condition of the algorithm. We set a threshold θ to control the stop condition of the routing variable. If the variation of the routing variable is larger than θ, the updating algorithm will continue, otherwise the updating stops. When the node i stops updating routing variables, it notifies its neighbors and stops sending updating information to the neighbors. Then the neighbor nodes record the quiescent state of i and use the routing information of the quiescent state of i to do the updating.

Assuming that before updating, the network total cost is C_T and after one round of iteration, the total cost of the whole network is becoming $(C_T)^*$. When the step size is sufficiently small, we have $(C_T)^* - C_T \leq 0$. In the experiment, we determined the proper value of stepsize to ensure the descent property of C_T and make sure the algorithm will converge. In each round of iteration for one destination, each node communicate with its all neighbors once. The time complexity for updating information of one destination in one iteration is $O(D)$, in which D is the diameter of the network topology. Based on numerical results, our algorithm can converge to the near global optimal within 15 rounds iteration.

Algorithm 1 Routing under dynamic traffic flow

1: each destination j maintains a timer I to record the time passing in an epoch. I is initialized as T.
2: each node i initializes $r_j^i(0)$.
3: When the new epoch begin, every destination j sends up in the $GEN/_G(j)$ a "signal" for all nodes to start a new epoch and reset I as T.
4: When node i receives the "signal" in the epoch L, it drops $r_j^i(L)$ and uses the $r_j^i(L+1)$ for the new epoch $L+1$. Then, it starts to update routing variable set using current traffic flow information in the new epoch.
5: each node i is continuously monitoring the traffic flow (the arrival rate r_j^i) and after every fixed interval it recomputes a new r_j^i for destination j.

When the traffic flow is dynamic, according to the variation of the traffic, we divide the time into epochs with fixed length of T time units. During an epoch, the routing variable set Φ_j^i for all j will not be updated. Under dynamic traffic in the epoch, each node allocates traffic demand passing through it according to the current routing variable set. When a new epoch begins, every node starts to update the routing variable set using new traffic flow information until the routing variable set converges to the new stable state. Let $r_j^i(t)$ denotes the injected traffic rate from i to j at time t. Algorithm 1 describes the dynamic routing algorithm in detail.

4. PERFORMANCE EVALUATION

In this section, we evaluate the energy saving routing framework in this paper by simulation using real network data. Our simulation mainly has two goals: 1) To compare the performance of the distributed routing algorithm on $GEN\_G(t)$ we generated with the optimal solution and with the distributed routing algorithm on optimal $DAGs$. (The optimal $DAGs$ are generated by optimal solution) 2) To evaluate the performance of the routing framework under dynamic traffic.

We do the simulation on the AS3967 topology obtained from Rocketfuel dataset. There are 17 nodes and 56 links in our experimental topology. We assume that the network topology is fixed with no link failure. We use self-similar traffic (can be generated using Pareto distribution) to model dynamic traffic and use one snapshot of the dynamic traffic matrix as the static traffic matrix.

Based on the previous works' definition [3], [5], [8], we use $d_{ik}(x_{ik}) = \frac{x_{ik}}{CA_{ik}-x_{ik}}$ as our delay function and $f_{ik}(x_{ik}) = \frac{x_{ik}^2}{CA_{ik}}$ (we do not consider startup cost here) as our energy function. Both energy and traffic delay functions are convex (there are research on more complex energy and delay functions [11], we consider a simpler form as a start). In our experiment, the optimization solutions are obtained using the LINGO optimization solver [10].

We first generate the $GEN\_G(t)$ and then run the distributed routing algorithm on it. With the proper stepsize, the algorithm can converge within nearly 15 rounds iterations. We compare this result with the optimal solution obtained by LINGO and with the shortest path routing. Fig. 2 is the comparison among different routing methods. Opt represents the solution got by the LINGO, and Dis is our distributed routing scheme. SP denotes the shortest path algorithm (min-hop). We find that our distributed scheme achieves almost 30% energy saving compared with shortest path routing while keeps packet delay nearly the same with shortest path strategy.

Then, we extract the optimal $DAGs$ generated by LINGO, and conduct the distributed routing algorithm on it. We find it can converges to the exact global optimal. We therefore find that Dis can not obtain the exact optimal value because the $GEN\_G(t)$ we generated is not the same with the optimal one (generated by the LINGO). By comparing the $DAGs$ generated by the LINGO and all $GEN\_G(t)$ generated by our loop-free path finding algorithm, we find that they are different in some links. We call the links exist in our $GEN\_G(t)$ for all t but not in LINGO's $DAGs$ are "bad links". We find that the majority of the "bad links" in our experiment are reserved by alphabetical index order of nodes with the same height. However, it is very interesting that our generated $GEN\_G(t)$ don't affect the optimality of delay function but only impair the optimality of the total energy cost a little.

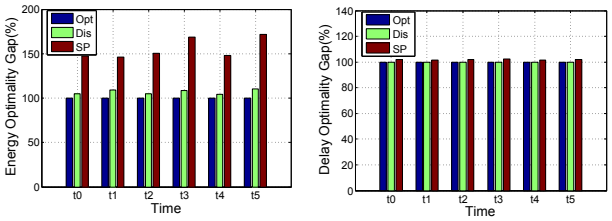


Figure 2: Comparison among methods on total delay and energy

We also conduct the experiment under dynamic traffic. We set different *epoch* lengths and compare the performance of dynamic routing with optimal solution. In Fig. 3, the optimality gap means the gap between the evaluated methods

and the LINGO's solution. $Dis(epoch = 1)$ and $Dis(epoch = 5)$ represent dynamic routing under different epoch length. $ShortestPath$ represents shortest path routing method. We can observe from the results that the routing strategy works well under dynamic traffic compared with shortest path strategy and it can always achieve near optimal value with certain epoch length.

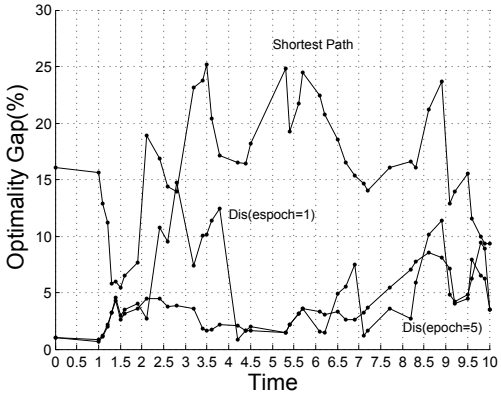


Figure 3: Distributed routing under dynamic traffic

5. CONCLUSIONS

In this paper, we consider a routing problem simultaneously optimizing energy consumption and delay in wired networks. We formulate the problem as a bi-objective optimization problem. We then propose a hop-by-hop distributed loop-free routing scheme to solve this bi-objective optimization problem. By obeying the optimal routing law, our scheme can approach the global optimal. Experiments have been carried out using real network data and the results show that, after several rounds of iteration, the distributed algorithm converges to the near Pareto optimal. Our technique can have *30*% energy saving compared with shortest path routing.

In our future works, we will work on the improvement of the loop-free multi-path finding algorithm and try to give theoretical measurements of the optimality of *DAGs*. Meanwhile, how to speed up the convergence is also an interesting topic. We can further try to explore the theoretical bound of the iteration number of updating algorithm in routing. We also plan to test this distributed strategy in networks of larger scale with topology changes. The routing algorithm under dynamic traffic also merits further research.

6. ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China grant 61020106002, 61161160566 and 61202059, and the Comunidad de Madrid grant S2009TIC-1692, Spanish MICINN grant TEC2011-29688-C02-01.

7. REFERENCES

[1] M. Andrews, A. F. Anta, L. Zhang, and W. Zhao. Routing and scheduling for energy and delay minimization in the powerdown model. In *INFOCOM*, pages 21–25, 2010.

[2] M. Andrews, A. F. Anta, L. Zhang, and W. Zhao. Routing for energy minimization in the speed scaling model. *IEEE/ACM Transaction on Networking*, 20(1):285–294, 2012.

[3] S. Antonakopoulos, S. Fortune, and L. Zhang. Power-aware routing with rate-adaptive network elements. In *GLOBECOM Workshops*, pages 1428–1432, 2010.

[4] S. Avallone and G. Ventre. Energy efficient online routing of flows with additive constraints. *Comput. Netw.*, 56(10):2368–2382, July 2012.

[5] D. P. Bertsekas and R. Gallager. *Data networks (2. ed.)*. Prentice Hall, 1992.

[6] S. Chen and K. Nahrstedt. An overview of quality-of-service routing for the next generation high-speed networks: Problems and solutions. *IEEE Network*, 12, 1998.

[7] W. Fisher, M. Suchara, and J. Rexford. Greening backbone networks: reducing energy consumption by shutting off cables in bundled links. In *Green Networking*, pages 29–34, 2010.

[8] R. G. Gallager. A minimum delay routing algorithm using distributed computation. *IEEE Transactions on Communications*, 25(1):73–85, 1977.

[9] M. Gupta and S. Singh. Greening of the internet. In *In ACM SIGCOMM*, pages 19–26, 2003.

[10] http://www.lindo.com/. 2012.

[11] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan. A power benchmarking framework for network devices. In *Proceedings of the 8th International IFIP-TC 6 Networking Conference*, NETWORKING '09, pages 795–808, 2009.

[12] R. T. Marler and J. S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, 2004.

[13] P. Patel-Predd. Energy-efficient ethernet. In *IEEE Spectrum*, page 13, 2008.

[14] Y. Shang, D. Li, and M. Xu. Energy-aware routing in data center network. In *Green Networking*, pages 1–8, 2010.

[15] N. Vasić, P. Bhurat, D. Novaković, M. Canini, S. Shekhar, and D. Kostić. Identifying and using energy-critical paths. In *Proceedings of the Seventh Conference on emerging Networking EXperiments and Technologies*, CoNEXT '11, pages 18:1–18:12, 2011.

[16] N. Vasić and D. Kostić. Energy-aware traffic engineering. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, e-Energy '10, pages 169–178, 2010.

[17] Y. Wang, G. Lu, and X. Li. A study of internet packet reordering. In *ICOIN*, pages 350–359, 2004.

[18] Z. Wang and J. Crowcroft. Quality of service routing for supporting multimedia applications. *IEEE Journal on Selected Areas in Communications*, 14:1228–1234, 1996.

[19] M. Zhang, C. Yi, B. Liu, and B. Zhang. Greente: Power-aware traffic engineering. In *Proceedings of the The 18th IEEE International Conference on Network Protocols*, ICNP '10, pages 21–30, Washington, DC, USA, 2010. IEEE Computer Society.