Robust Energy-Aware Routing with Uncertain Traffic Demands

Heng Lin
Tsinghua University
henglin1991@gmail.com

Mingwei Xu
Tsinghua University
xmw@cernet.edu.cn

Yuan Yang
Tsinghua University
yyang@csnet1.cs.tsinghua.edu.cn

***Abstract*—Energy conservation has become a major challenge to the Internet. In existing approaches, a part of line cards are switched into sleep mode for energy conservation, and the routing is configured carefully to balance energy saving and traffic engineering goals, such as the maximum link utilization ratio (MLUR). Typically, traffic demands are used as inputs, and routing is computed accordingly. However, accurate traffic matrices are difficult to obtain and are changing frequently. This makes the approaches difficult to implement. Further, the routing may shift frequently, and is not robust to sudden traffic changes.**

In this paper, we propose a different approach that finds one energy-aware routing robust to a set of traffic matrices, particularly to arbitrary traffic demands. Such a routing without energy consideration is known as the demand-oblivious routing, and is well studied. However, the problem becomes much more challenging when energy conservation is involved. To overcome the challenges, we first define a new metric, namely oblivious performance ratio (OPR) with energy constraint, which reflects the MLUR distance from a routing to the optimal routing when certain energy conservation requirement is satisfied. We model the problem of minimizing the performance ratio, and analyze the lower and the upper bounds. Then, we propose Robust Energy-Aware Routing (REAR) to solve the problem in two phases. REAR select sleeping links based on extended robust link utilization (ERLU) or algebraic connectivity, and compute the routing based on a classical demand-oblivious routing algorithm. We evaluate our algorithms on real and synthetic topologies. The simulation results show that REAR can save 19% of line card power while the performance ratio is less than 34%.

I. Introduction

Energy conservation has become a global concern nowadays. The Internet is one of the major energy consumers, and its rapid growth makes the green Internet a hot research topic. In the Internet backbone, energy is mainly drawn by routers and switches. Such devices consume almost full power even if the traffic load is small. Thus, an effective method to save energy in the Internet is to aggregate traffic into part of the routers when the traffic load is small, and switch the under-utilized components (routers or line cards) into off/sleep mode. Such a method is known as the energy efficient routing.

An important issue for energy efficient routing is to avoid network congestion after the traffic is aggregated. Many approaches have been proposed in existing works. Most approaches compute the routing based on real-time traffic matrices or link loads, to achieve a good load balancing and avoid congestion. However, such approaches come at a cost of obtaining real-time traffic data. Furthermore, the routing may shift frequently with the traffic changes, and a sudden traffic change may still induce congestions. To this end, we need to study the robustness of energy efficient routing.

Specifically, we study the energy efficient routing when traffic matrices cannot be obtained or predicted precisely, i.e., with uncertain traffic demands. To achieve robustness, we need to find a routing, which can perform near optimally under a range of traffic matrices. The key technique that makes this possible is the advanced *demand-oblivious routing*. A seminal work [2] find that, the distance between the maximum link utilization ratio (MLUR) of a demand-oblivious routing and the MLUR of the optimal routing is bounded. Thus, no matter how the traffic demand changes, the demand-oblivious routing can guarantee certain performance. We note that this conclusion is in a network without off/sleep components, and we need to consider the situation when energy conservation is required.

However, existing demand-oblivious routing algorithms cannot be directly applied to energy efficient routing. There are several challenges. First, we need to define a metric that can effectively measure the distance between a robust energy efficient routing and the optimal routing, because the existing metric for demand-oblivious routing fails in the situation when some components are switched into off/sleep mode. Second, we need to analyze whether the metric can be bounded, just like for demand-oblivious routing. If there exists no bound, then it is not feasible to find a robust energy efficient routing. Third, we need practical algorithms to compute the robust energy efficient routing. Specifically, we need to determine: 1) which routers or line cards should be switched into off/sleep mode, to achieve energy efficiency; and 2) in which path to forward the traffic for robustness, while the path does not traverse the off/sleep components.

In this paper, we overcome the aforementioned challenges. First, we define a new metric, namely the oblivious performance ratio with energy constraint (OPRE). The OPRE reflects the MLUR distance from a routing to the optimal one when certain energy conservation requirement is satisfied. We model the problem of minimizing the OPRE. Second, we prove that there exists a robust energy efficient routing with the minimum OPRE, which has an upper bound given a network. Then, we propose Robust Energy-Aware Routing (REAR) scheme, which uses heuristic algorithms to solve the problem. We develop algorithm XXX, which chooses off/sleep line cards in a way that the OPRE can be minimized potentially. We

then develop algorithm XXX to compute the routing in the remaining topology, by extending existing optimal demand-oblivious routing algorithm. We evaluate our algorithms by simulations on real topologies and synthetic traffic demands with random fluctuations. The results show that REAR can achieve an OPRE of 1.34 while 19% of line card power is saved.

The rest of the paper is organized as follows. Section II shows the related work. Section III presents metric OPRE and formally models the problem. We presents the bounds on OPRE in Section IV, and propose our algorithms in Section V. Section VI shows our simulation setup and results, and Section VII concludes our work.

II. Related Work

related work

III. Problem Statement

A. Background of Demand-Oblivious Routing

As mentioned above, demand-oblivious routing aims at finding one routing that performs near optimally under a range of traffic demands. In traffic engineering, a typical metric to evaluate routing performance is the maximum link utilization ratio (MLUR). Clearly, the MLUR is corresponded with a specified traffic matrix (TM). Demand-oblivious routing defines oblivious performance ratio (OPR) to evaluate the routing performance without the knowledge of TM. We briefly present the background.

Given a TM, the distance between a routing to the optimal one is defined as the ratio between their MLURs. Formally, a network is modeled as undirected graph $G(V, E)$, where V is the set of vertices (nodes), and E is the set of edges (links). Let cap_{ij} denote the capacity of the link $(i, j) \in E$. Let d_{ab} denote the traffic demand from origin node a and destination node b, and m denote the TM that contains d_{ab} for all $a, b \in V$. Let $f_{ab}(i, j)$ be the fraction of d_{ab} that is routed on link (i, j) $(0 \leq f_{ab}(i, j) \leq 1)$. Routing r is specified by $f_{ab}(i, j)$ for all $a, b \in V$ and $(i, j) \in E$, and we will formally define routing consistency later in Section III.C. Let $U_{r,m,G}$ be the MLUR of routing r under TM m. We have

$$U_{r,m,G} = \max_{(i,j)\in E} \frac{\sum_{a,b} d_{ab} f_{ab}(i, j)}{cap_{ij}}. \tag{1}$$

Let $P(\{r\}, \{m\}, G)$ be the *performance ratio* of routing r under TM m, which reflects how far from the routing to the optimal one, and is defined as

$$P(\{r\}, \{m\}, G) = \frac{U_{r,m,G}}{\min_{r'} U_{r',m,G}}. \tag{2}$$

The *oblivious performance ratio* (OPR) for routing r is defined by extending TM m to a set of TMs M, where M can be the set of any TMs. We have

$$P(\{r\}, M, G) = \max_{m\in M} P(\{r\}, \{m\}, G). \tag{3}$$

The target of demand-oblivious routing is to find routing r that minimizes OPR $P(\{r\}, M, G)$. Such a "robust" routing is independent of a specific TM, but can perform near optimally. A seminal work [2] uses linear programming to find the routing that minimizes the OPR, and finds that the optimal solution exists, which means that the OPR is bounded.

B. Oblivious Performance Ratio with Energy Constraint

With energy constraint, a network may have to switch part of line cards into off/sleep mode to save the total energy consumption. This changes the network topology and makes metric OPR fail to evaluate the robustness of a routing. We use an example to show this. Assume that G is a cycle with n unit capacity links. Then, the minimal OPR of G is $2 - 2/n$ [2]. Now we pruning one link from G to save energy, and the topology changes to G^*. Because there is only one routing feasible in G^*, OPR $P(\{r\}, M, G^*)$ equals 1. Since 1 is less than $2 - 2/n$ for $n > 2$, it means that the routing after pruning one link is more robust than before. However, it is false because there are less links and the network is more likely to be congested.

The intrinsic reason for such a "fake robust" is that the topology is not changing when performance ratio is computed in Eq. (2). To address this issue, we extend the definition of OPR. Formally, let G^* be a sub-graph of G that satisfies the energy constraint (We will formally give the model in Section III.C). The extended performance ratio is defined as

$$P(\{r\}, \{m\}, G, G^*) = \frac{U_{r,m,G^*}}{\min_{r'} U_{r',m,G}}. \tag{4}$$

Let $P^*(\{r\}, M, G)$ be the *oblivious performance ratio with energy constraint* (OPRE) for routing r. We define $P^*(\{r\}, M, G)$ by extending m to M, and finding G^* that satisfies the energy constraint and has the minimum performance ratio. We have

$$P^*(\{r\}, M, G) = \min_{G^*} \max_{m\in M} P(\{r\}, \{m\}, G, G^*) \tag{5}$$

Note that when there is no energy constraint, G is the sub-graph that has the minimum performance ratio, and Eq. (5) naturally reduces to Eq. (3).

C. Problem Formulation

We formulate our problem as following:

$$\text{Minimize} \quad P^*(\{r\}, M, G)$$

$$\text{s.t. (1), (4)}$$

$\forall a, b, i \in V$:

$$\sum_{j,s.t.(i,j)\in E} f'_{ab}(i, j) - \sum_{j,s.t.(j,i)\in E} f'_{ab}(j, i) = \begin{cases} 1, & i = b \\ -1, & i = a \\ 0, & i \neq a, b \end{cases} \tag{6}$$

$\forall a, b, i \in V$:

$$\sum_{j,s.t.(i,j)\in E^*} f_{ab}(i, j) - \sum_{j,s.t.(j,i)\in E^*} f_{ab}(j, i) = \begin{cases} 1, & i = b \\ -1, & i = a \\ 0, & i \neq a, b \end{cases} \tag{7}$$

$$\sum_{l\in E^*} p(l) / \sum_{l\in E} p(l) < 1 - \theta \tag{8}$$

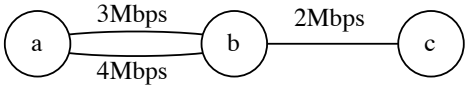


Fig. 1. Model Example, TM I : (a, b, 2M), (a, c, 1M); TM II : (a, b, 1M), (a, c, 1M).

$$0 \leq f_{ab}(i,j), f'_{ab}(i,j) \leq 1 \tag{9}$$

Eq. (6) is our target, which minimize the value of OPRE defined in III.B. Eq. (7) say G^* be a sub-graph of G, which means remove/sleep some links from G, and keep vertices the same. Eq. (8) ensures the graph is connected and define the routing. In details, $O^i_{(i,j)}$ equal to 1 if i is the tail node of link (i,j), and $I_(i,j)^i$ equal to 1 if i is the head node of link (i,j), this equation states the flow conservation constraints. In the other hand, $f_{ab}(i,j)$ is specified by routing γ, if the graph is not connected it will be unavailable for some origin and destination pair. And from Eq. (8) we know that routing r is corresponded with G^*, and r' with G. We calculate the power constraints in Eq. (9), it ensure that we can save more than θ energy consumption. The function p which maps link to power consumption is proposed in [1].

D. Model Example

We will take an example to explain how to choose the link to close in our algorithm. Three hostes include : a, b, c, three links are with repectively capacity of 3M, 4M and 2M. For simpleness, we suppose there are two TM : $(a,b,2M),(a,c,1M)$ and $(a,b,1M),(a,c,1M)$. For each traffic matrix, the optimal route is obvious, we will trace 2M from a to b across the lower link and trace the 1M from a to c across the upper one for the first traffic matrix, whose maximum link utilization is 0.5. we will trace all the traffic acorss the lower link for the second traffic matrix, whose maximum link utilization is 0.5 as well.

Now for some reason, we will choose one link to shut down for power saving without lose connection of the network. There are two choice, remove either the upper link or the lower link. Let us take a little calculation: when remove the upper one, we should change all the traffic acorss the lower link, as a result, in the first TM the maximum link utilization is 0.75 and the second is 0.5; when close the lower link, we should trace all the traffic acorss the upper link, in the first TM the link utilization is 1 and the other is 0.667. So according to our theory, the $P^*(M,G)$ should be 1.5 and the optimal successor network topology will be the one which 3M link is closed.

IV. Algorithm

In our paper, Robust Energy-Aware Routing (REAR) algorithm works in two phases. Firstly, REAR select links should be put to sleep from the origin topology based on extended robust link utilization (ERLU), then compute the robust routing based on demand-oblivious routing algorithm.

A. Extended Robust Link Utilization

The demand-oblivious routing is not always the best one for specific TM, but good enough for a range of TM. we mentioned the definition of link utilization above when demand and routing are determined, as demand become oblivious, the definition is also not suitable. So we define a notation named extended robust link utilization (ERLU) replace the normal link utilization like :

$$u^e_{ij} = \frac{\sum_{a,b} f_{ab}(i,j)}{cap_{ij}} \tag{10}$$

Proposing this definition for two consideration: firstly, the more flows across the link, the greater the u^e_{ij} is; secondly, the more fraction of one flow across the link, the greater the u^e_{ij} is. The link with greater ERLU is also more important than others in a sense. Although there is no demand here, we mean there is more probability that this link have greater link utilization when specific demand come.

B. Algorithm Phase One

REAR sleep as many links as possible without losing connectivity of graph. Originly, we should calculate ERLU of all the links and sleep the lowest one from the graph, then repeat calculate and remove process until arrive some specific threhold. Obviously, it is NP-Hard, following is a heuristic algorithm.

For the origin graph, we calculate the ERLU of every link, and then sort these values from small to big, output ordered list denoted as Γ:

$$\Gamma = \{..., l_i, ..., l_j, ...\} \tag{11}$$

where $u^e_{l_i} < u^e_{l_j}$.

Pay attention we only compute the ERLU once at the begining of algorihtm, and the ordered list Γ show the order of 'importance' among links in the graph.

Now we begin selecting which link should be sleep. We denote the set of the sleeping links as S, and the output of this phase is final graph $G^* = (V, E - S)$. We set $S = \emptyset$, and repeat our selecting process, each iteration we select one link, remove it from Γ and put it into S. In iteration i, algorithm scan links as the order in Γ, we try to remove this link from the graph to check if the graph is still connected and the energy conservation have not arrive the threshold. If so, we select it then go next iteration. Otherwise choose the next link from Γ for trying to remove. Algorithm stop until all the links in Γ is tried but no one is satisfied with both connectivity and power threhold.

But before going ahead our algorithm, there is another thing we should care, how to measure the power of graph. We simple

take an power model from [1] showed in Table II, and defined the difference of power consumption between two graphs as:

$$diff_p = 1 - \theta = \frac{\rho(G^o_{S,l})}{\rho(G^o)} * 100 \quad (12)$$

where $\rho(G^o_{S,l})$ is the power consumption of the final graph, when the links set S and link l are both removed from the origin graph G^o, and the $\rho(G^o)$ is the power consumption of the origin graph.

If we set $diff_p$ valued 90%, it means that whenever we try to remove the link l from the origin graph in iteration, the power consumption should never lower than the 90% of origin one. Following is our implementation:

Algorithm REAR : Phase One

Input: $G(V, E)$, $threhold$;
Output: S in which links should be switched off;
$G(V, E - S)$ which is the final network topology;

1: **for** each link l in E
2: $\Gamma[l] \leftarrow u^e_l$;
3: Resort Γ in increasing order based on u^e_l;
4: $S \leftarrow \emptyset$, $goon \leftarrow true$;
5: **while** $goon$
6: $goon \leftarrow false$;
7: **for** each link l in $\Gamma - S$
8: **if** $G_{S,l}$ is connected and $\rho(G_{S,l})/\rho(G) > threhold$
9: $S \leftarrow S \cup \{l\}$;
10: $goon \leftarrow true$;
11: **break**;
12: **return** $S, G(V, E - S)$;

C. Algorihtm Phase Two

Once we get the output network topology from the first phase, it is time to compute the robust routing. On one hand, computation process may cost too much time if we directly calculate the robust routing in the final graph; on the other hand, the robust routing based on the final graph may not be the best one. There is a heuristic algorithm based on the demand-oblivious routing on the origin graph, which should be obtained at first. then adjust routing in details according to the links we switched off.

The demand-oblivious routing can be computed by a single LP with $O(mn^2)$ variables and $O(nm^2)$ constraints[1] :

min r
$f_{ij}(e)$ is a routing
$\forall$ links l: $\sum_m cap(m)t(l, m) \leq r$
$\forall$ links l, $\forall$ pairs $i \rightarrow j$:
$f_{ij}(l)/cap(l) \leq p_l(i, j)$
$\forall$ links l, $\forall$ nodes i, $\forall$ edges $e = j \rightarrow k$:
$\pi(l, link - of(e)) + p_l(i, j) - p_l(i, k) \geq 0$
$\forall$ links l, m: $\pi(l, m) \geq 0$
$\forall$ links l, $\forall$ nodes i: $p_l(i, i) = 0$
$\forall$ links l, $\forall$ nodes i, j: $p_l(i, j) \geq 0$

where the $cap(l)$ is the capacity of link l; and $\pi(l, m)$ is the weights for every pair of links l, m; and the variables $p_l(i, j)$ for each link l and OD pair i, j is the length of the shortest path from i to j according to the link weights $\pi(l, m)$.

The routing we get indicate how to arrive at destination node from source node for every OD pair in the origin topology. What is different is that, the flow can be splited in the routing, i.e. there may be two paths ($path_1$, $path_2$) both from source node s to destination node d, and the optimal obilious routing trace 70% traffic on $path_1$ and left on $path_2$. Although splitting flow is hard handled, we take a transformation for the case like that: when an flow is coming, there is 70% probability we trace it on $path_1$, otherwise $path_2$. This is easily implementated in real world.

Because all the routing is based on the origin topology, when some links are switched off, we must adjust the routing as well. Supposed there are some paths between two vertices s and d, maybe one link in some paths be removed, and these paths become not reachable any more, we should adjust the traffic in these paths to other paths. Of course, we can not put the whole traffic on another path, this will make some links of the path congested, so we should split the traffic to some paths 'averagely', in the sense of extended robust link utilization. Similarly to link utilization, we define the extended robust link utilization as the maximum extended robust link utilization of links.

Take pair (s, d) for example, routing includes paths from s to d, such as $p_1, p_2, ..., p_n$. While only p_1 trace the removed link, and of course it will be unreachable. We split the traffic of p_1, and put them on other paths to make the extend robust link utilization of paths almost closely.

And this is our implementation:

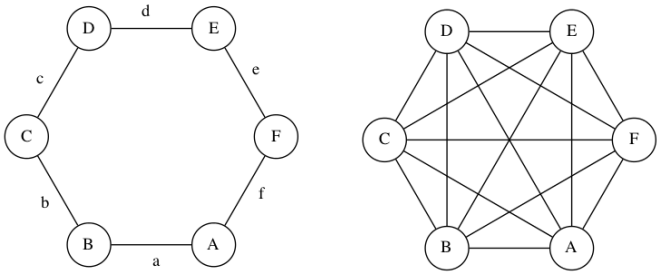
Algorithm REAR : Phase Two

Input: $G(V, E)$ which is the origin topology;
S which is the switch-off links set generated by Phase One;
R which is the Robust Routing on origin topology;
Output: $Routing$ Robust Energy-Aware Routing on new topology

1: $G^* \leftarrow G(V, E - S)$;
2: **for** each link l in S
3: **for** each $s, d, paths$ in R
4: $traffic \leftarrow 0$;
5: **for** each $path$ in $paths$
6: **if** $link$ in $path$
7: $traffic \leftarrow traffic$ + $path.traffic$;
8: $paths$.remove($path$);
9: $yen\_paths \leftarrow$ yens_algorithm(G^*, s,d);
10: **while** $traffic$!= 0
11: sort_paths_by_extend_robust_link_utilization(yen_paths)
12: yen_paths[0].$traffic$ = $traffic$ / N
13: $traffic$ = $traffic$ - $traffic$ / N
14: $paths$.add(yen_paths[0])
15: $paths$.merge();
16: **return** R;

V. Samples and Proof

For explaining OPRE clearly, we will show it in two simple topologies, Cliques and Cycles. With a specific power saving threshold, some links will be removed from the topology then robust routing will be computed. Taking two extreme examples from origin topology, one with no links removed, i.e the topology itself, and the other with no more link can be removed, i.e. spanning tree of the topology. And the corresponding thresholds are 0 and maximum value can be achieved. We prove the upper bound of OPRE for cycles and cliques in these two situations, and we say when taking a threshold between 0 and maximum value, the corresponding OPRE is also bounded.



(a) Circle (b) Clique

Fig. 2. Circle and Clique topology of 6 nodes

Lemma 1. *The upper bound of OPR for C_n (the cycle on n vertices) and K_n (the complete graph on n vertices) is 2-2/n.*

Lemma 1 is proved in [2] and it assumes all the links the same. We extend this lemma on C_n and K_n with different capacity of links. In the latter proof, we use the extended C_n and K_n definition without specified.

Theorem 2. *The upper bound of OPRE for spanning tree of C_n is 1+Max_{cap}/Min_{cap}.*

Proof. Cycles topology connect all the nodes by a cycle, any node in which has two links joined. Figure 2.(a) show the six nodes cycles topology, let nodes named from A to F and links named from a to f. Now suppose a traffic in the topology then calculate the optimal routing for it, let D_a and C_a be the demand and capacity of the link a, without loss of generality, we say the link d is the bottleneck, namely link with MLUR.

The spanning tree of C_n is generated by removing one link from origin topology. There are two ways to generate the spanning tree, remove the link d or remove the other link such as f.

For case I, when we switch off the link d, the origin traffic pass the link will be redirected. The worst situation is that all the traffic on the link d will be put to the other path : $D \leftarrow C \leftarrow B \leftarrow A \leftarrow F \leftarrow E$. After traffic redirection, the bottleneck link may become the link e, we have OPRE in this routing situation:

$$\frac{\frac{D_e+D_d}{C_e}}{\frac{D_d}{C_d}} = (1+\frac{D_e}{D_d})\frac{C_d}{C_e} \tag{13}$$

Because the link utilization of link d is the maximum one, we have :

$$\frac{D_d}{C_d} > \frac{D_e}{C_e} \tag{14}$$

$$\frac{D_e * C_d}{C_e * D_d} < 1 \tag{15}$$

so we have OPRE in case I :

$$(1+\frac{D_e}{D_d})\frac{C_d}{C_e} < \frac{C_d}{C_e} + 1 \tag{16}$$

For case II, similar to case I, all the traffic will be redirected on the other path between F and A. There are two sub cases after traffic redirection, the link D is still the bottleneck or the link E become the new bottleneck. In case II.I, we have OPRE:

$$\frac{\frac{D_d+D_f}{C_d}}{\frac{D_d}{C_d}} = 1+\frac{D_f}{D_d} \tag{17}$$

Because the link utilization of link d is the maximum one in origin topology, we have:

$$\frac{D_d}{C_d} > \frac{D_f}{C_f} \tag{18}$$

$$\frac{D_f}{D_d} < \frac{C_d}{C_f} \tag{19}$$

So we get OPRE in case II.I:

$$(1+\frac{D_f}{D_d}) < \frac{C_d}{C_f} + 1 \tag{20}$$

In case II.II, we get OPRE:

$$\frac{\frac{D_e+D_d}{C_e}}{\frac{D_d}{C_d}} = \frac{D_e * C_d}{D_d * C_e} * \frac{C_d * D_f}{C_e * D_e} \tag{21}$$

Similarly as mentioned above, we get equations :

$$\frac{D_d}{C_d} > \frac{D_e}{C_e} => \frac{D_e * C_d}{C_e * D_d} < 1 \tag{22}$$

$$\frac{D_d}{C_d} > \frac{D_f}{C_f} => \frac{D_f * C_d}{D_d} < C_f \tag{23}$$

So we obtain OPRE:

$$\frac{D_e * C_d}{D_d * C_e} + \frac{C_d * D_f}{C_e * D_d} < 1 + \frac{C_f}{C_e} \tag{24}$$

As a result, the upper bound of extended robust link utilization must be :

$$Max\{1+\frac{C_d}{C_e}, 1+\frac{C_d}{C_f}, 1+\frac{C_f}{C_e}\} < 1 + \frac{Max_{cap}}{Min_{cap}} \tag{25}$$

Particularly, if all the links have the same capacity, the upper bound equal to 2 for the spanning tree. Otherwise, the upper bound only dependent to the greatest ratio of link capacity. □

Theorem 3. *The upper bound of OPRE for C_n is 1+Max_{cap}/Min_{cap}.*

Proof. The robust routing which will not trace demand on one link must be less optimal than the routing can trace demand on any link, and obviously the former routing is one of the routing on the spanning tree of C_n. So the bound of Theorem 2 is also the bound of Theorem 3.

Particularly, If the links of C_n have the same capacity, OPRE will reduce to OPR. According to Lemma 1, we have the inequality:

$$OPRE = OPR < 2 - \frac{2}{n} < 1 + \frac{Max_{cap}}{Min_{cap}} = 2 \tag{26}$$

□

Theorem 4. *The upper bound of OPRE for K_n is 1+Max_{cap}/Min_{cap}.*

Proof. Similar to the proof of Theorem 3, we say the robust routing which will not trace demand on one link must be less optimal than the real optimal robust routing. And it is one routing of topology which remove one link from origin topology K_n. Without loss of generality, we say the link a is the bottleneck, and we will switch off it or other link like b, then consider the OPRE when adjust the demand to other link.

In case I, after we switch off link a, the new bottleneck link must be the affected link, the link contains the origin traffic on link a. Otherwise if the link b is the bottleneck, but we adjust the traffic on other links, it means that we can do the same thing in the origin topology, and the bottleneck become link b rather than a. It is conflict with our assumption. So we have the OPRE as :

$$\frac{\frac{D_b+D_a}{C_b}}{\frac{D_a}{C_a}} = (1 + \frac{D_b}{D_a})\frac{C_a}{C_b} \tag{27}$$

Because the link a is the bottleneck link with the maximum link utilization, like what we do above, we can get

$$(1 + \frac{D_b}{D_a})\frac{C_a}{C_b} < 1 + \frac{C_a}{C_b} \tag{28}$$

In case II, the bottleneck may still be link a or the new link c. If it is link a, and no new traffic across it, the OPRE equal to 1 obiliously. Or we adjust the new traffic D_b on it, we can get the OPRE:

$$\frac{\frac{D_a+D_b}{C_a}}{\frac{D_a}{C_a}} = 1 + \frac{D_b}{D_a} < 1 + \frac{C_b}{C_a} \tag{29}$$

If the bottleneck become new link c, similar to our proof of circle topology, we can get OPRE:

$$\frac{\frac{D_c+D_b}{C_c}}{\frac{D_a}{C_a}} = \frac{D_c * C_a}{D_a * C_c} + \frac{D_b * C_a}{D_a * C_c} < 1 + \frac{C_b}{C_c} \tag{30}$$

So we obtain the same conclusion as circle topology, the OPRE must be :

$$Max\{1 + \frac{C_a}{C_b}, 1 + \frac{C_b}{C_a}, 1 + \frac{C_b}{C_c}\} = 1 + \frac{Max_{cap}}{Min_{cap}} \tag{31}$$

When the links of K_n is the same, Theorem 4 will reduce to Lemma 1. The upper bound is bounded. □

Theorem 5. *The upper bound of OPRE for spanning tree of K_n is $\frac{n^2}{2}Max_{cap}/Min_{cap}$.*

TABLE I
TOPOLOGIES

Topology	Nodes	Links	Links can be Removed
Abilene	12	15	4
Geant	23	37	15
Cernet2	20	22	3

Proof. Every link in the spanning tree will split the graph to two parts, the flow between two parts must need across this link, Without loss of generality, let l_{da} be the link with maximum link utilization. And let l_{be} be the bottleneck link in the optimal routing for origin topology. So we have OPRE:

$$\frac{\frac{\sum_{i,j\in V} D_{ij}}{C_{da}}}{\frac{D_{be}}{C_{be}}} = \sum_{i,j\in V} \frac{D_{ij}}{D_{be}}\frac{C_{be}}{C_{da}} \tag{32}$$

Because the l_{be} is the bottleneck link, we have:

$$\frac{D_{ij}}{C_{ij}} < \frac{D_{be}}{C_{be}} \tag{33}$$

$$\frac{D_{ij}}{D_{be}} < \frac{C_{ij}}{C_{be}} \tag{34}$$

So we have OPRE:

$$\sum_{i,j\in V} \frac{D_{ij}}{D_{be}}\frac{C_{be}}{C_{da}} < \sum_{i,j\in V} \frac{C_{ij}}{C_{be}} \tag{35}$$

And we know that, the flow between two parts of topology must less than $\frac{n^2}{2}$, so OPRE must lower than:

$$\sum_{i,j\in V} \frac{C_{ij}}{C_{be}} < \frac{n^2}{2}\frac{Max_{cap}}{Min_{cap}} \tag{36}$$

□

VI. Experiments and Results

We simulated our algorithm on real world topology, including Abilene, Geant and Cernet2, whose number of nodes and links are listed in Table I. Then we generate random traffic matrix with Gravity Model [3], which assume that the traffic demand between nodes is proportional to their combined capacity of connecting links. To extrapolate a complete TM, we take an attribute margin w to scale the traffic range from $1/w$ to w base on the basic traffic demand. Particularly, when the w limit extremity, we say the traffic matrix is really arbitrary, and our algorithm is irrelevant with traffic.

A. OPRE versus Power Saving

In Figure 3, we see that the value of OPRE is at least 1 for all topologies even no links removed, particularly in Geant the value is 1.24, it means the robust routing always not be the optimal for all TMs. It is reasonable because our robust routing is used for a range of traffic rather than a specific TM.

When we switch off links, the topology lose connectivity and some routing paths will be failed, traffic on those paths will be adjusted to others, which result in the value of OPRE

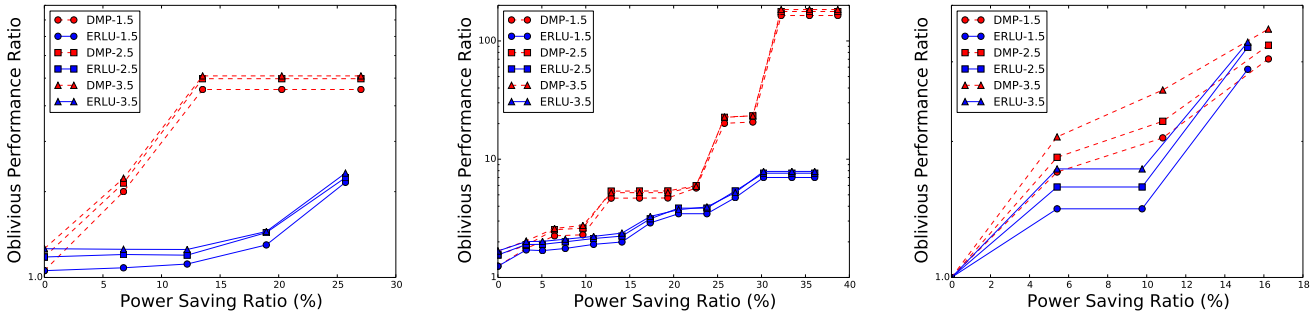


Fig. 3. OPRE versus Power Saving: (1). Abilene, (b). Geant, (c). Cernet2

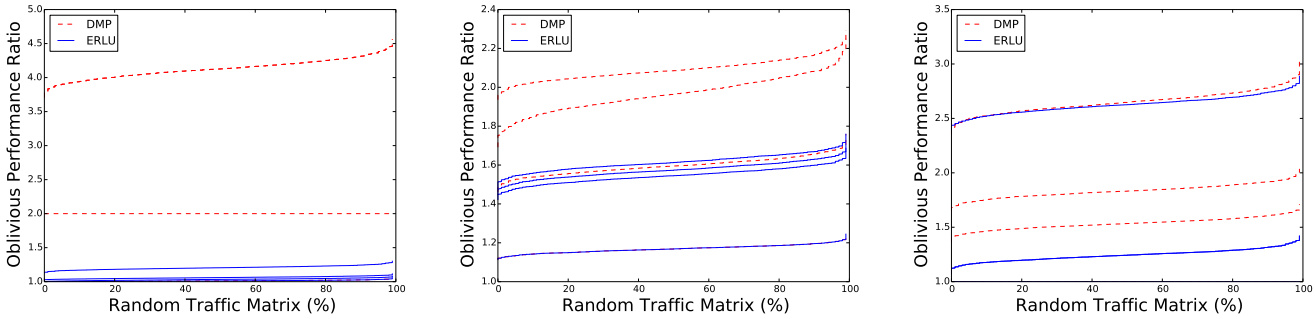


Fig. 4. OPRE versus TMs: (a). Abilene, (b). Geant, (c). Cernet2

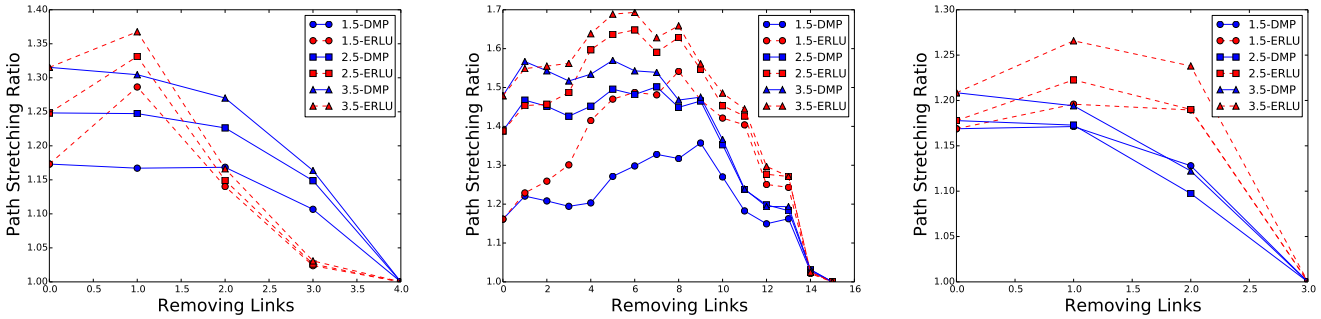


Fig. 5. Path Stretching: (a). Abilene, (b). Geant, (c). Cernet2

increased. We take margin from 1.5 to 3.5 and observe this phenomena from Figure. Pay attention that we scale the y-axis as logarithm for comparing DMP and ERLU in one figure. Intuitively, we can see ERLU is always better than DMP whatever margin or power saving target is, because ERLU consider much more than DMP, such as capacity and robust traffic flows. Another information from Figure 4 is that, we should carefully set the value of power saving for avoiding removing too much links, which result in the OPRE increases rapidly, induce network congestions easily and make the robust routing be less 'robust'.

we concern more about how is the OPRE varifying when achived specific power saving target. Obviously, the more links we removed, the more power we saved, but how to quantify the power of one link is difficult. Green TE proposed a simple power model [1], which can be represented in Table II. In

TABLE II
GREEN TE POWER MODEL

Line-Card	Speed(Mbps)	Power(Watts)
1-Port OC3	155.52	60
8-Port OC3	1244.16	100
1-Port OC48	2488.32	140
1-Port OC192	9953.28	174

which, the total energy of topology is dominant by line cards of routers or switches, when saying switch off the links we mean put down the according line cards.

We compute the power saving ratio as the total power of the removed links over the total power of all the links. In Figure 3 (a), it shows we can save 19% energy only with an OPRE of 1.30 in Abilene topology, and in Geant and Cernet2, the OPRE is little higher. Curves present some scalariform, it means that in some range of power saving, the OPRE rise slowly, and in the end of range, the energy conservation is efficient.

B. OPRE versus Margin

We take the margin w as one of the input for computing robust routing, and margin identify a range of TMs the routing is robust for. Once the w limit extremely, our robust routing is said without knowledge of traffic matrix. Figure 3 shows the OPRE increases little as w increases, particularly in Geant, the difference almost can not be observed. It is obvious because when the w is greater, the traffic matrix is random in more wider range, and our OPRE may achieve worse case with more probability.

C. OPRE versus TMs

To simulate the worst case, we generate 1000 traffic matrices for every topoloy with margin attribute w. Figure 4 shows the OPRE distribution in the process of experiment. For avoiding mess result from too many lines, we just show the first three lines and the base line, which shows the distribution when no links is removed, i.e. seven lines in each figure.

In Figure 4 (a), we can only observe five lines because there are two overlapping lines, which means that when removing links the worst case for the robust routing does not change. Maybe the random traffic matrices is not bad enough, or the removed link really do not affect the MLUR. And from Figure 4, we see that the OPRE is not always be achived for most traffic matrices, the common value is much less than OPRE.

Comparing DMP and ERLU, the conclusion is similar to Figure 3, the latter is always better than the former. Even in some time, ERLU obtains more power saving but with a lower OPRE.

D. Path Stretching

Path stretching must be careful concerned in routing problem. Suppose two vertices connect directly each other in the origin topology, if we remove the connected link then they can connect by another path which is composed by multi links. However we argue path stretching between these situation may be unjust, because path stretching result from the difference of topologies rather than routing selection. So we obtain the shortest path by Dijkstra Algorithm in the final topology, and compare it with robust routings generated by our REAR algorithm. Futher, we show the difference between two ways of removing link, and see the tendency when margin increased.

In Figure 5, we see every line ends with path stretching ratio of 1, because topology become a tree structure when lose too much links, every path between vertices come to be unique, namely each path is the shortest path. Another conclusion is that, even in the origin topology, our robust routing is 17% longer than the shortest one, it is beacuse our algorithm not only consider the MLUR but also the robust performance. However the Dijkstra just take the path length as metric.

We can observe that, ERLU is always shorter than DMP, althoungh in Geant, the worst case is 36% than the shortest path when margin equal to 1.5. And in the first removed links, the path stretching ratio is near 20%.

VII. CONCLUSION

The conclusion goes here.

REFERENCES

[1] M.Zhang, C.Yi, B.Liu and B.Zhang, "GreenTE: Power-Aware Traffic Engineering".

[2] D.Applegate and E.Cohen, "Making Intra-Domain Routing Robust to Changing and Uncertain Traffic Demands: Understanding Fundamental Tradeoffs".

[3] M.Roughan, A.Greenberg, C.Kalmanek, M.Rumsewicz, J.Yates, and Y.Zhang. Experience in measuring backbone traffic variability: models, metrics, measurements, and meaning. InProceedings of the 2nd Internet Measurement Workshop. ACM, 2002

[4] A.P. Bianzino, L. Chiaraviglio, M. Mellia, Distributed algorithms for green IP networks, in: IEEE INFOCOM Workshop on Green Networking and Smart Grid, 2012