Reliable Green Routing Using Two Disjoint Paths

Gongqi Lin, Sieteng Soh, Mihai Lazarescu
Curtin University of Technology
Perth Australia
Gongqi.lin@postgrad.curtin.edu.au

Kwan-Wu Chin
University of Wollongong
Wollongong Australia
kwanwu@uow.edu.au

Abstract—**Network robustness and throughput can be improved by routing each demand *d* via two disjoint paths (2DP). However, 2DP routing increases energy usage while providing lower link utilization and redundancy. In this paper, we address an NP-complete problem, called 2DP-EAR, that aims to switch off redundant nodes and links while guaranteeing two constraints: traffic demands must be afforded 2DP, and maximum link utilization. We design an efficient heuristic, called 2DP by Nodes First (2DP-NF). We have extensively evaluated the performance of 2DP-NF on both real and/or synthetic topologies and traffic demands. As compared to using Shortest Path routing, on the GÉANT network, 2DP-NF can save around 20% energy by switching off links only with negligible effects on path delays and link utilization, even for MLU below 30%. Furthermore, 2DP-NF can obtain 39.7% power savings by switching off both nodes and links on the GÉANT network.**

Keywords – robustness; thoughput; power savings; maximum link utilization; two disjoint paths

I. INTRODUCTION

Recently, disjoint path routing has been proposed to improve network reliability and throughput [1, 2]. Two paths between source node s_d and destination node t_d are node (link)-disjoint, called 2DP-N (2DP-L), if they have no common nodes (links). In [3], the authors combine link/node-disjoint paths with QoS routing to guarantee various performance requirements, *e.g.*, reliability and delay. While over-provisioning resources to provide backup paths using traffic engineering (TE) can help improve reliability, it exhibits poor energy efficiency, especially at low traffic load periods. Recent reports [4, 5] show that power consumption of large wired networks has increased tremendously. In fact, the energy consumption of the Information and Communication Technologies (ICT) sector will grow from 22GW in 2007 to 95GW in 2020, generating 1.4 Gt of CO2 or approximately 2.8% of global warming by 2020 [4].

Recently, the authors of [6] propose a solution that minimizes powered-on links by routing demands with 2DP-L under maximum link utilization constraint. However, their work only considers switching off links and does not address the 2DP-N problem. This problem is significant because routers consume significantly more power as compared to links [7]. Reference [8] proposes a solution to switch-off both links and nodes. However, their solution [8] does not address 2DP routing, which affects fault-tolerance [9]. We note that references [10] and [11] have proposed power-aware routing algorithms, but they do not require generated paths to be link-disjoint or node-disjoint.

In this paper, we extend the work in [6] to reduce power by switching off both nodes and links. Specifically, our contributions are twofold. First, we formulate a problem, called Two Disjoint Paths Energy-Aware Routing (2DP-EAR), to minimize the power consumption of network resources by switching off both unused nodes and links subjected to two constraints: (i) link utilization must be no larger than a given U, and (ii) there are at least T fractions of routes that use 2DP. Our model aims to reduce energy usage of networks that support 2DP routing to improve fault-tolerance and throughput. The formulation extends that of [6], which only considers switching off links. Second, we propose a novel algorithm, called Two Disjoint Paths by Nodes First (2DP-NF), to solve 2DP-EAR. Our 2DP-NF prioritizes switching off nodes to links since nodes consume an order of magnitude more energy [7]. Our extensive simulation results show the advantage of using 2DP-NF. For example, for the GÉANT network, 2DP-NF can obtain 39.7% power savings.

The rest of the paper is organized as follows. Section II gives an overview of the 2DP-EAR problem and summarize of notations. Section III describes our heuristic algorithm, 2DP-NF, to solve the problem. Section IV evaluates the performance of 2DP-NF using both real and/or synthetic topologies and data. Finally, Section V concludes the paper.

II. PROBLEM FORMULATION

In this section, we describe our problem, Two Disjoint Paths Energy-Aware Routing (2DP-EAR). Without loss of generality, we only define the problem for two link-disjoint paths (2DP-L); note that a two node-disjoint paths (2DP-N) is also 2DP-L.

A. Notation

Consider a computer network that is represented by a weighted directed graph $G(V,E)$, where V is the set of n nodes, and E is the set of m links. Each node in V represents a router, and each link $(i, j)\in E$ between nodes v_i and v_j (v_i, $v_j\in V$, $v_i\neq v_j$) represents a communication channel with capacity/bandwidth $c_{ij}>0$. For edge $(i, j)\in E$, let $0\leq f_{ij}\leq c_{ij}$ be the traffic load through edge (i, j), and hence, its utilization is given by $u_{ij}=f_{ij}/c_{ij}$. The utilization u_{ij} is bounded by a threshold $0\leq U\leq 1.0$ and is set by network operators; *i.e.*, $u_{ij}\leq U$. A link's remaining capacity is defined as $r_{ij}=U*c_{ij}-f_{ij}\geq 0$. We assume each link $(i, j)\in E$ can be switched-off independently. Each node v and link (i, j) consumes equal power p_v and p_{ij} respectively. Let E_v be the set of links connected to a node v, and $f_v = \sum_{(v,j)\in E_v} f_{vj}$ and $c_v = \sum_{(v,j)\in E_v} c_{vj}$ represent the total throughput rate and total throughput capacity of node v respectively.

Let D be a set of all demands in $G(V, E)$, and (s_d, t_d, b_d) denote a traffic demand d=1, 2, …, $|D|$ between source node $s_d \in V$ and terminal node $t_d \in V$, where b_d is the amount of traffic exchanged between these nodes. For each demand d, let $SP_d=\{sp_{dq}$ | all paths for demand d indexed by an integer number $q>0\}$. Let $B(sp_{dq})$ be the capacity of any path $sp_{dq}\in SP_d$ calculated by taking the smallest r_{ij} for each link $(i, j)\in sp_{dq}$. Two (s_d, t_d) paths are link-disjoint if they have no common links. Let $DP_d=\{dp_{dl}$ | all two link-disjoint paths for demand d indexed by an integer number $l>0\}$. Note that $dp_{dl}=\{sp_{dx}, sp_{dy}\}$, where $sp_{dx}, sp_{dy}\in SP_d$ have no links in common. We use R_d to denote a route for demand d that contains a single path sp_{dq}, a 2DP-L dp_{dl} or any multiple (s_d, t_d) paths with sufficient capacity to route b_d.

Let R^β be a route set β that contains all R_d for each demand $d\in D$, *i.e.*, $R^\beta=\{R_d^\beta \mid R_d^\beta \subseteq SP_d$ and/or $R_d^\beta \in DP_d\}$. The set of all possible solutions to route demands in D is denoted as $R=\{R^\beta \mid \beta=1, 2, \ldots, |R|\}$. We use $TP^\beta \subseteq R^\beta$ to represent a set of all $R_d^\beta \in R^\beta$ that includes at least one $dp_{dl}\in DP_d$, and $M(R^\beta)$ be the fraction of (s_d, t_d) pairs in R^β that are routed over 2DP-L, *i.e.*, $M(R^\beta)=|TP^\beta|/|R^\beta|$; we set $T_{max}=\max\{M(R^\beta)\}$. Finally, let $S(R^\beta)$ be the total power of active nodes and links used in set R^β and $U(R^\beta)=\max\{f_{ij}/c_{ij} \mid \forall (i, j)\in R^\beta\}$.

B. Problem Statement

Consider a tuple (G, D, T, U), where $G(V, E)$ is a network topology, D is a set of traffic demands, $0\le T\le T_{max}$ is the required fraction of the total number of routes that contains at least one 2DP-L, and U is a configured threshold of maximum link utilization. We define the 2DP-EAR problem as follows.

2DP-EAR: Find a set $R_{\min}\in R$ that can be used to route all demands in D such that

$$S(R_{\min}) = \min\ \{p_v V(R^\beta)+p_{ij}E(R^\beta) \mid R^\beta\in R\} \quad (1)$$

$$M(R_{\min}) \ge T \quad (2)$$

$$U(R_{\min}) \le U \quad (3)$$

Note that $V(R^\beta)$ and $E(R^\beta)$ are the number of power-on nodes and links in each $R^\beta\in R$ respectively. The term $S(R_{\min})$ represents the power consumption of all nodes and links in R_{min} subject to constraints (2) and (3). Constraint (2) requires there to be at least T fraction of routes in R_{min} that include at least one 2DP-L, while constraint (3) ensures the link utilization of each link used in R_{min} be no larger than U. The 2DP-EAR problem is a variant of the multi-constrained path (MCP) problem since it aims to generate an optimal set of feasible routes $R_{\min}$ subject to two constraints, *i.e.*, (2) and (3). Since MCP with more than one constraint is known to be NP-complete [12], we conclude that 2DP-EAR is NP-complete.

III. GREEN ROUTING ALGORITHMS

In this section, we present our approach, Two Disjoint Paths by Node First (2DP-NF), shown in Fig. 1, to heuristically solve the 2DP-EAR problem. Initially, the set of remaining nodes V_r and links E_r are V and E; 2DP-NF produces $V-V_r$ and $E-E_r$ as its outputs. We first describe 2DP-NF for 2DP-L, called 2DP-NF-L. We show how to use 2DP-NF for 2DP-N, called 2DP-NF-N, in Section III.B, and discuss the time complexity of 2DP-NF in Section III.C.

A. 2DP-NF-L Algorithm

As shown in Fig. 1, 2DP-NF-L has six main steps. **Step 1** uses *Yen*'s algorithm [13] to generate $k\ge 1$ shortest paths, $KSP_d=\{sp_{d1}, sp_{d2}, \ldots, sp_{dk}\}$, for each demand d; we assume the delay of each link to be one unit, and thus each path length can measured by its hop count. Note that for each demand d, we have $KSP_d \subseteq SP_d$. Let $KSP=\{KSP_d \mid d=1,\ldots,|D|\}$ be the set of all k-shortest paths for all demands D.

```
2DP-NF-L(G(V,E),D,T,U)
Begin
      // Initially, $E_r = E$, and $V_r = V$
1)    Generate $KSP_d$ in $G(V, E)$ for each demand d;
2)    Call 2DP-Routing(G(V, E), D, T=1.0, KSP), and compute $T_{max}$;
3)    For each $v \in V_r$ do
          If $f_v$ == 0 then
              $V_r = V_r - \{v\}$; // remove v
              $E_r = E_r - E_v$;
       End-For
4)    For each $(i, j) \in E_r$ do
          If $f_{ij}$ == 0 && flag(i, j)==false then
              $E_r = E_r - \{(i, j)\}$; // remove (i, j)
          Else
              $r_{ij} = U*c_{ij} - f_{ij}$; // update the remaining capacity
      End-For
      Set flag(i, j)=false for each in (i, j) in $E_r$;
5)    For each $v \in V_r$ in increasing order of $|E_v|*(f_v/c_v)$ do
          //routing D in G with one less node is feasible
          If (2DP-Routing(G($V_r$–{v}, $E_r$-$E_v$),D,T, KSP) == true) then
              $V_r = V_r - \{v\}$; // remove v
              $E_r = E_r - E_v$; // remove connected links
              Go to 3);
6)    For each $(i, j) \in E_r$ in descending order of its $r_{ij}$ do
          //routing D in G with one less edge is feasible
          If (2DP-Routing(G($V_r$, $E_r$–{(i, j)}),D,T, KSP) == true) then
              $E_r = E_r - \{(i, j)\}$; // remove (i, j)
               Go to 4);
       End-For
   Return $E–E_r$ & $V–V_r$
End
```

Figure 1. 2DP-NF-L Algorithm

Step 2 uses function **2DP-Routing**() to distribute the traffic of each demand $d\in D$ through its candidate paths KSP_d, and computes T_{max}; we will describe **2DP-Routing**() and T_{max}'s calculation later in this section. **Step 3** switches off each unused node v, *i.e.*, each node with f_v=0, and its incident links. **Step 4** turns off each unused link (i, j) with variable *flag*(i, j)=false, *i.e.*, each link with f_{ij}=0, and calculates the spare capacity $r_{ij}=U^*c_{ij}-f_{ij}$ of other links. Specifically, we set *flag*(i, j)=true for each link (i, j) in a 2DP so that the links are not switched off, *i.e.*, the path comprises of the links becomes a backup path. **Step 5** aims to switch-off node v, starting from v with fewest connected links $|E_v|$ and lowest link utilization (f_v/c_v); intuitively, because such node is used by fewer flows, rerouting these flows successfully is more probable. If there exists a feasible R^{β} without using v and its incident links E_v, *i.e.*, **2DP-Routing**() returns true, the step switches off node v and all links in E_v and repeats Step 3 to update affected nodes and links; otherwise, **Step 5** is repeated using the next candidate node. **Step 6** aims to switch off each link with the largest spare capacity; this step uses **2DP-Routing**() to check if all traffic can be routed through the remaining links in E_r, while satisfying the required constraints. The step is repeated for the next candidate link if **2DP-Routing**() fails to generate a feasible R^{β}; otherwise, we repeat Step 4.

```
2DP-Routing(G(V_r, E_r), D,T, KSP)
Begin
  Temp_TP= Φ;
  For each d ∈D do
    /* Part 1 */
      Call Find-2DP(G(V_r, E_r), KSP_d) to generate DP_d;
    /* Part 2 */
      If |Temp_TP|<T*|R^β| then
        If |DP_d|>0 then
          If Distribute-2DP (KSP_d , DP_d, b_d)==true then
            Insert R_d^β into Temp_TP;
          Else If Use-Non-2DP(KSP_d, b_d)==false then
            Return false;
        Else //if |DP_d|==0
          If Use-Non-2DP(KSP_d, b_d)==false then
            Return false;
      Else // Use any route for each remaining demand
        If Use-Non-2DP(KSP_d, b_d)==false then
          Return false;
  End-For
  If |Temp_TP|<T*|R^β| then
      TP^β=Temp_TP;
      Return true;
  Else
      Return false;
End
```

Figure 2. Function 2DP-Routing()

Function **2DP-Routing**(), shown in Fig. 2, contains two parts. **Part 1** uses function **Find-2DP**(), shown in Fig. 3, to generate each 2DP-L, $dp_{dl}=\{(sp_{dx}, sp_{dy}) \mid sp_{dx}\in KSP_d\}$, for each demand d. Specifically, function **Find-2DP**() generates a graph $G_1(V_r, E_1)$ by deleting all links in each $sp_{dx}\in KSP_d$ from G. Then, it uses *Yen*'s algorithm to generate k-shortest paths from $G_1(V_r, E_1)$, and stores the paths in the set KSP'_d. Finally, it generates $dp_{dl}=\{sp_{dx}, sp_{dy}\}$ for each path $sp_{dy}\in KSP'_d$ that has no common links with sp_{dx}, and stores the pair in set DP_d in increasing path length $max\{L(sp_{dx}), L(sp_{dy})\}$ order.

```
Find-2DP(G(V_r, E_r), KSP_d)
Begin
    l=1;
    For each sp_dx∈KSP_d do
        Generate G_1(V_r, E_1) from G(V_r, E_r) by deleting all edges in sp_dx;
        Generate k paths for (s_d, t_d) from G_1, and store in KSP'_d;
        For sp_dy∈KSP'_d do
            dp_dl={sp_dx, sp_dy};
            If dp_dl∉DP_d then
                Store (sp_dx, sp_dy) into DP_d;
                l++;
        End-For //for sp_dy
    End-For //for sp_dx
    Return DP_d.
End
```

Figure 3. Function Find-2DP()

Part 2 routes all traffic demands subject to constraint T. It uses the set *Temp_TP*, initially empty, to store R_d^{β} for each demand d. If $|Temp\_TP|\geq T^*|R^{\beta}|$ is true, the remaining traffic demands can be routed via any routes using function **Use-Non-2DP**(), shown in Fig. 4. The function aims to distribute each demand (s_d, t_d, b_d) via its shortest path. However, if the path does not have sufficient capacity, the function will route remaining flow through the next available shortest path. The step is repeated until b_d is completely routed and the function returns true; otherwise, it returns false and **2DP-Routing**() returns false since it fails to route all demands in D. Note that **Use-Non-2DP**() considers shortest paths and therefore uses fewer links for routing a demand than using 2DP-L.

```
Use-Non-2DP(KSP_d, b)
Begin
    Temp_R=Φ;
    // Distribute traffic with multiple non-disjoint paths routing
    For each sp_dq∈KSP_d in ascending order of its length do
        If b≤B(sp_dq) then
            Insert sp_dq in Temp_R;
            Increase f_ij of each link (i, j)∈sp_dq in Temp_R by b;
            Insert each path sp_dq in Temp_R into R_d^β
            Return true;
        Else
            //Route B(sp_dq) flow of the traffic through sp_dq;
            Insert sp_dq in Temp_R;
            b = b – B(sp_dq);
    End-For
    Return false; // If flow b cannot be routed through all paths in KSP_d
End
```

Figure 4. Function Use-Non-2DP()

```
Distribute-2DP (KSP_d , DP_d, b_d)
Begin
 // Distribute traffic with 2DP-N
   For each dp_dl={sp_dx, sp_dy} ∈DP_d in increasing length order do
     If b_d ≤ B(sp_dx)+ B(sp_dy) then
         If b_d ≤ B(sp_dx) then
             Increase f_ij of each link (i, j) in sp_dx by b_d;
             Set flag(i, j)=true for each link (i, j) in sp_dy;
         Else
             Increase f_ij of each link (i, j) in sp_dx by B(sp_dx);
             Increase f_ij of each link (i, j) in sp_dx by b_d-B(sp_dx);
         Insert sp_dy and sp_dx in R_d^β ;
         Return true;
   End for
   Increase f_ij of each link (i, j) in sp_dx∈dp_d1 by B(sp_dx);
   Increase f_ij of each link (i, j) in sp_dy∈dp_d1 by B(sp_dy);
   Insert sp_dy and sp_dx in R_d^β ;
   Return Use-Non-2DP(KSP_d, b_d – B(sp_dx) – B(sp_dy));
End
```

Figure 5. Function Distribute_2DP()

However, if $|Temp\_TP|<T^*|R^{\beta}|$ and at least one 2DP-L of demand d exists, *i.e.*, $|DP_d|>0$, **Part 2** uses **Distribute-2DP**(), described later, to distribute traffic of demand d via its dp_{dl} and insert R_d^{β} into *Temp_TP*. For each demand d, if $|DP_d|=0$ or **Distribute-2DP()** returns false, **2DP-Routing**() uses **Use-Non-2DP()** to route b_d via one or more paths starting from the shortest path in KSP_d. Finally, if all traffic demands are allocated successfully and the requirement $|Temp\_TP|\geq T^*|R^{\beta}|$ is satisfied, $Temp\_TP=TP^{\beta}$ and **2DP-Routing**() returns true. When function **2DP-Routing**() returns true, *i.e.*, it has successfully routed all demands in D, it updates R_d^{β} for each demand d and the total flows on each link $(i, j)\in E_r$, *i.e.*, f_{ij}. When the function returns false, it will maintain the previous routing $R^{\beta\text{-}1}$.

For each demand with $|DP_d|>0$, **Distribute-2DP**(), shown in Fig. 5, routes traffic demand b_d. The function aims to route the traffic d through its 2DP-L, *i.e.*, $(sp_{dx}, sp_{dy})\in dp_{dl}$, and we assume $\mathrm{B}(sp_{dx})\le \mathrm{B}(sp_{dy})$. If $b_d\le \mathrm{B}(sp_{dx})$ then it routes b_d through only a single path sp_{dx}, and sets $flag(i, j)$=true for each (i, j) backup path sp_{dy} that can enhance routing reliability. This flow distribution is different than in [10] that splits flow equally. If $\mathrm{B}(sp_{dx})\le b_d\le \mathrm{B}(sp_{dx})+\mathrm{B}(sp_{dy})$ then it routes the traffic volume $\mathrm{B}(sp_{dx})$ via sp_{dx}, and routes volume $b_d-\mathrm{B}(sp_{dx})$ via sp_{dy}. Note that, $\mathrm{B}(sp_{dq})=\min\{r_{ij}|(i, j)\in sp_{dq}\}$. However, if $\mathrm{B}(sp_{dx})+\mathrm{B}(sp_{dy})<b_d$, it uses function **Use-Non-2DP()** to distribute the remaining flow, *i.e.*, $b_d-(\mathrm{B}(sp_{dy})+\mathrm{B}(sp_{dx}))$. Note that **Distribute-2DP()** returns false when **Use-Non-2DP()** returns false, *i.e.*, it fails to route the remaining flows through the paths in KSP_d; for this case **2DP-Routing**() will use function **Use-Non-2DP()** to route b_d. Recall that **Step 2** of **2DP-NF-L** in Fig. 1 uses function **2DP-Routing**() to initialize the traffic distribution in the network. In this step, we set T=1.0 so that the function routes each demand through its 2DP-L whenever possible, and thus T_{max} is set $|TP^1|/|D|$, where TP^1 includes all demands in D that are routed via 2DP-L in original network.

B. 2DP-NF-N

In general, a network contains fewer 2DP-Ns than 2DP-Ls since each 2DP-N is also a 2DP-L, but not vice versa; thus using the latter for routing is more popular [14]. Further, using link-disjoint paths is much more energy efficient than node-disjoint paths [15]. However, 2DP-N is more resilient to failures than 2DP-L because they protect against both node and link failures. One can use 2DP-NF for applications that require 2DP-N by considering only each 2DP-L, $dp_{dl}=\{sp_{dx}, sp_{dy}\}$, since each set of 2DP-N for each (s_d, t_d) is a subset of its set of 2DP-L for each (s_d, t_d). However, 2DP-NF-N is expected to switch off less number of links than 2DP-NF-L due to fewer candidate 2DP-N.

C. Time Complexity of 2DP-NF

For ***2DP-NF***, *Yen*'s algorithm, see Step 1, incurs $O(kn(m+nlogn))$ time. Note that n and m are the total number of nodes and edges in G respectively. In Step 2, **2DP-Routing()** takes $O(|D|(k^2+mk^2))=O(k^2mn^2)$ time because $|D|\le n^2$; $|D|$ is the total number of traffic demands. Step 3 and 4 require searching all nodes and links in G and therefore has a time complexity of $O(n+m)$. Step 5 needs up to $O(n)$ times to check whether a candidate node and its incident links can be deleted. Step 6 takes $O(m)$ to select each candidate. Thus, for m links, this step has complexity $O(m^2)$. Since **2DP-Routing()** incurs a bound of $O(k^2mn^2)$ and called $(n+m)$ times in Step 5 and 6, the total complexity of *2DP-NF* is $O(kn(m+nlogn)+k^2mn^2+k^2mn^2(n+m))=\mathrm{O}(k^2m^2n^2)$, since $m\le n^2$.

IV. EVALUATION

In this section, we provide detailed experimental findings and present numerical results on the effectiveness of 2DP-NF in reducing power in real networks and its impact on network delay and link utilization. We set $T=T_{\max}$ so that each demand is routed via its 2DP if possible, and use Shortest Path (SP) routing as a benchmark.

A. Experiment Setup

Table I shows three real topologies, *i.e.*, Abilene [16], GÉANT [17], and Sprint [18], and three synthetic topologies R_Abilene, R_GÉANT and R_Sprint that are used to evaluate the performance of 2DP-NF against SP. Each synthetic topology is generated by randomly selecting some nodes from its respective real topology as transit nodes.

TABLE I. NETWORK TOPOLOGIES

Network	Access Nodes	Transit Nodes	T_{max}(%)
Abilene	12	0	83.3
R_Abilene	10	2	100
GÉANT	23	0	100
R_ GÉANT	10	13	100
Sprint	52	0	37.4
R_Sprint	26	26	48.53

From the authors of [16], we obtained 288 traffic matrices for Abilene topology measured on Sep. 5th, 2004 for every 5 minutes within 24 hours. For GÉANT, the traffic matrices used were collected on May 5th, 2005 at an interval of 15 minutes; we obtained both the topology and 96 traffic matrices from the authors of [17]. For Sprint, we set its link capacity using the method in [19], and randomly generate a traffic matrix using the gravity model as [11], which is then scaled to obtain 10 different traffic loads. We refer each traffic matrix as TM_X, where X is the MLU of the network incurred by the traffic when using SP routing; *e.g.*, TM_40 is traffic matrix that produces MLU=40% in Sprint when using SP to route the traffic. For each network, we used **Step 2** of both 2DP-NF-L and 2DP-NF-N to obtain $T_{\max}$=83.3%, $T_{\max}$=100%, and $T_{\max}$=37.4% for Abilene, GÉANT, and Sprint, respectively.

We consider two types of nodes: access and transit [8]. Access nodes are sources and destinations of information and thus cannot be switched-off. In contrast, transit nodes are neither sources nor destinations of traffic; we aim to switch off as many of these nodes as possible. We assume that OC-192/STM-64 (10Gbps) line card is used for all links and thus the maximum delay of a single hop is around 200 ns [20]. Similar to [8], we assume each cable in link (i, j) has the same power consumption p_{ij}=0.6kw and each node v consumes the same power p_v=3kw. The power saving (PS) of each network is calculated as follows,

$$PS=(1-\frac{(m-|E_r|)p_{ij}+(n-|V_r|)p_v}{mp_{ij}+np_v})\times 100\% \tag{4}$$

Our simulations were performed on a Linux PC with 3.07 GHz CPU and 8 GB RAM.

B. Power off links only

We aim to see the effect on energy usage by only switching-off links on both 2DP-L and 2DP-N. In this simulation, we use Abilene, GEANT, and Sprint and use the traffic demands described in Section IV.A. Note that each traffic matrix considers all possible traffic demands, *i.e.*, each node is the traffic source/sink, and thus no node can be switched off. Hence, we calculate power savings using Equation (4) with p_v=0. Fig. 6 shows the power saving for the Abilene network using 2DP-NF with different maximum link utilization U; we use N and L to represent U for 2DP-NF-N and 2DP-NF-L respectively. Note that 2DP-NF-L and 2DP-NF-N could not find any links to be switched off at $U \leq 0.2$ and $U \leq 0.3$ respectively. For $U \geq 0.5$, 2DP-NF-L distributes traffic flow through the same set of paths while 2DP-NF-N requires $U \geq 0.6$ to obtain the same power savings, switching off 20% links, due to fewer candidate paths for 2DP-NF-N. Fig. 7 presents the power saving of GÉANT using 2DP-NF. Our approach obtains the same power saving for U=0.3 to U=1.0, and thus we only show the results for $U \geq 0.3$. Notice that U=0.1 and U=0.2 produce the same power saving, thus we only show U=0.1 for 2DP-NF-L and 2DP-NF-N. Although the power saving curve of 2DP-NF-L fluctuates during the day due to traffic changes, it always remains around 20.27% ~ 25.97%. However, as shown in Fig. 7, 2DP-NF-N performs worse than 2DP-NF-L due to fewer candidate paths, especially for U=0.1 that produces the worst power saving.

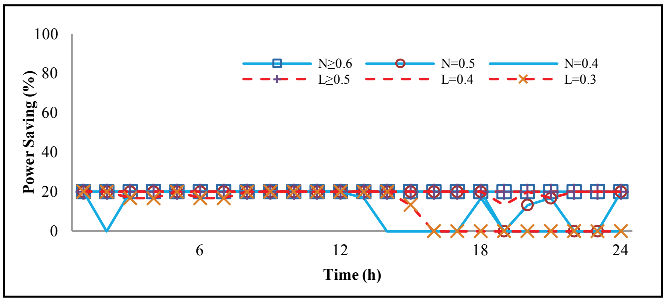

Figure 6. Power Saving of Abilene

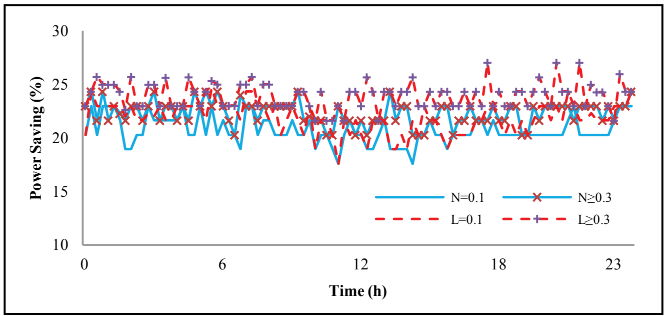

Figure 7. Power Saving of GÉANT

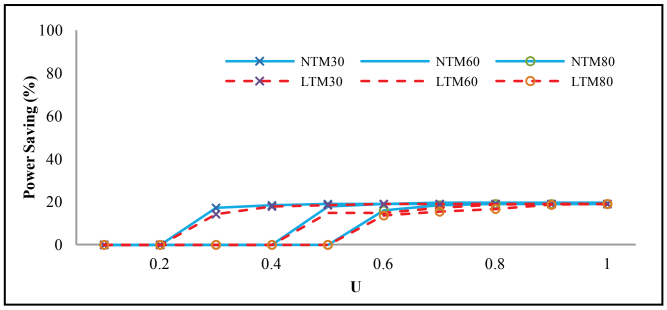

Figure 8. Power Saving of Sprint

Fig. 8 shows the power saving when using 2DP-NF with different TM_X on the Sprint network. We found that 2DP-NF produces the same set of routes with TM_10, TM_20 and TM_30, and thus we report only the result for TM_30. We found a similar situation for TM_40 to TM_60, and for TM_70 to TM_80. Note that 2DP-NF failed to save energy for TM_90 and TM_100. Running 2DP-NF-L with TM_30 (labeled LTM30 in the figure) increases PS from 17.26% to 19.04% when U is set from 0.3 to 0.5, and remains at 19.04% for U>0.5. However, 2DP-NF-N with TM_30 (NTM30) saves less power than 2DP-NF-L when U<0.6, saving only 14.29% at U=0.5. The reason for this is that with 2DP-L, there are more opportunities for the 2DP-NF-L algorithms to exploit candidate paths. The results for 2DP-NF with TM_60 (LTM60 and NTM60) and TM_80 (LTM80 and NTM80) have the same trend as LTM30. However, 2DP-NF fails to save energy when we set U<0.5 for LTM60 and NTM60, and U<0.6 for LTM80 and NTM80.

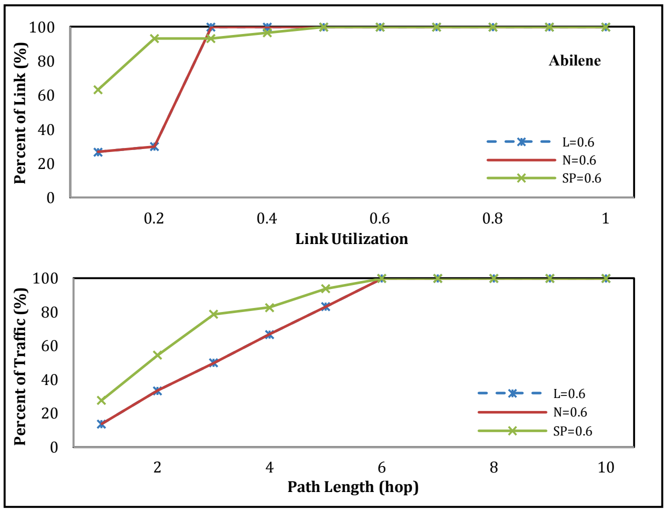

Figure 9. CDF of Link Utilization and Path Length on Abilene

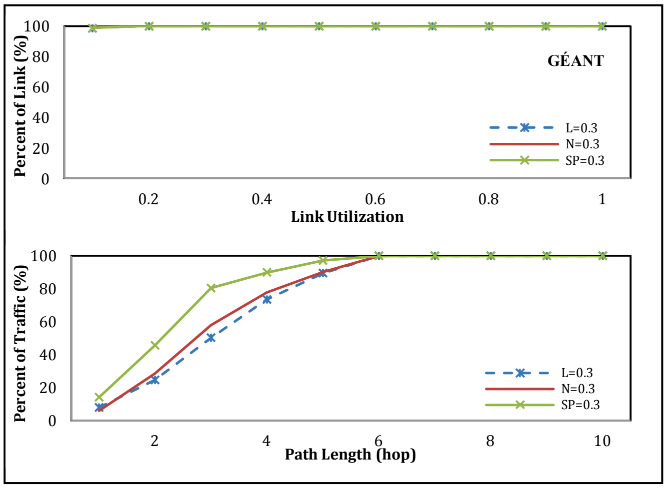

Figure 10. CDF of Link Utilization and Path Length on GÉANT

C. *Effects on Link Utilization and Path Length*

In this subsection, we show the effect of switching off links in Section IV.B on link utilization and path length. For Abilene, shown on the top of Fig. 9, the Cumulative Distribution Function (CDF) of link utilization and path length are the same when we set $U \geq 0.5$. From the figure, we see that the results of 2DP-NF-L are the same as 2DP-NF-N at U=0.6 due to the use of the same routing paths, which is worse than SP. However, as shown at the bottom of Fig. 9, their path length is no longer than the network diameter (6 hops). For GÉANT, shown in Fig. 10, the utilization of all links is less than 0.1 when we set U=0.1 and 0.2, and thus we do not show results here. When we set U=0.3, 2DP-NF-L has 1.35% links with utilization between 0.1 and 0.2, which is larger than 2DP-NF-N that reaches 100% links when the link utilization is less than 0.1. In Fig. 10 (bottom), since there are more candidate 2DP-Ls than 2DP-Ns, 2DP-NF-L can switch off more links, *i.e.*, saving more energy, as compared to 2DP-NF-N, and generates more alternative but longer routes. For Sprint, we obtain the CDF of link utilization and path length for TM_30, TM_60 and TM_80. From Fig. 11 (top), we see that 2DP-NF-L performs slightly worse than 2DP-NF-N, with more links having higher link utilization. Similarly, as shown in Fig. 11 (bottom) in term of path length, 2DP-NF-N performs worse than SP, but it obtains better results than 2DP-NF-L. 2DP-NF-N switches-off less number of links as compared to 2DP-NF-L. Notice that for each switched-off link, 2DP-NF needs to generate one or more alternative longer paths, and therefore more powered-off links correlate to longer paths.

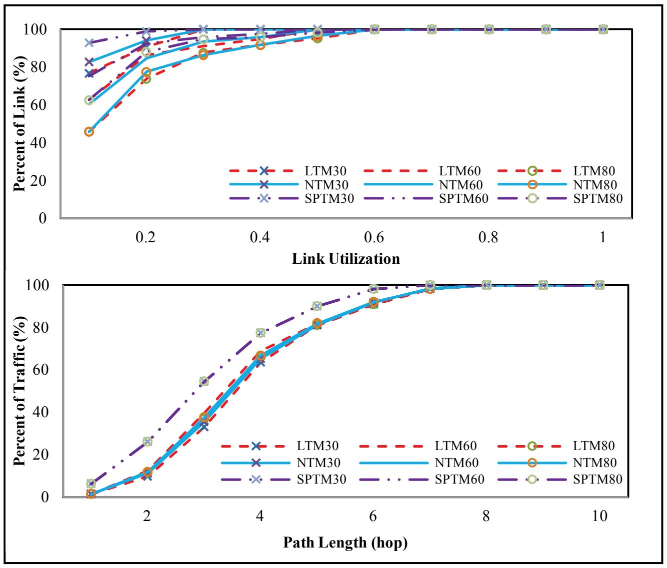

Figure 11. CDF of Link Utilization and Path Length on Sprint

D. Power off nodes and links

In this subsection, we use R_Abilene, R_GÉANT and R_Sprint in Table I each of which contains some transit nodes. For this case, 2DP-NF aims to maximally switch-off all possible links and transit nodes to minimize power usage.

(1) Power Saving

The power savings of these three topologies are shown in Table II. We see that 2DP-NF can switch off one node and seven links on the Abilene network, with power saving of up to 13.3%; for GÉANT, more than half of transit nodes (7/13) can be switched off, and the power saving reaches 39.7%; for Sprint, 7 transit nodes and 87 links can be powered off, achieving 28.5% power savings.

TABLE II. POWER SAVING WHEN NODES AND LINKS CAN BE SWITCHED-OFF

Power Saving	**R_Abilene**	**R_GÉANT**	**R_Sprint**
Off Transit Nodes	1	7	7
Off Links	7	40	87
Power Saving	13.3%	39.7%	28.5%

(2) Effects on Link Utilization and Path Length

Intuitively, switching off nodes and links will affect the link utilization and maximum routing path length (MRPL) since fewer nodes and links are available to carry traffic. Fig. 12, 13 and 14 show the CDF of link utilization and path length for three topologies: R_Abilene, R_GÉANT and R_Sprint with running 2DP-NF and 2DP by shortest path first without powering off nodes and links (2DP-SP).

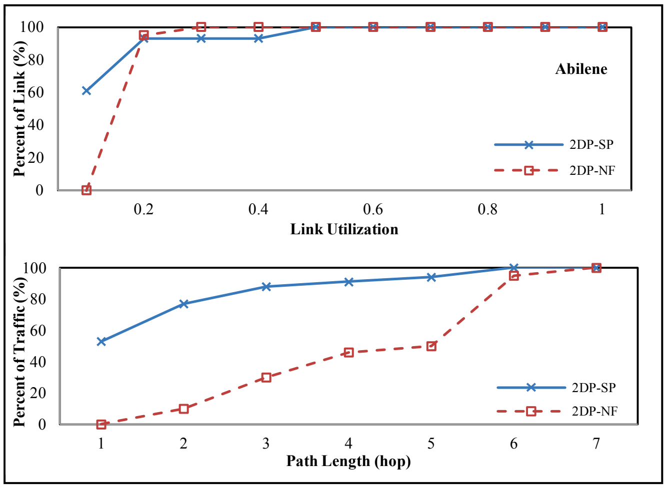

Figure 12. CDF of Link Utilization and Path Length on Abilene

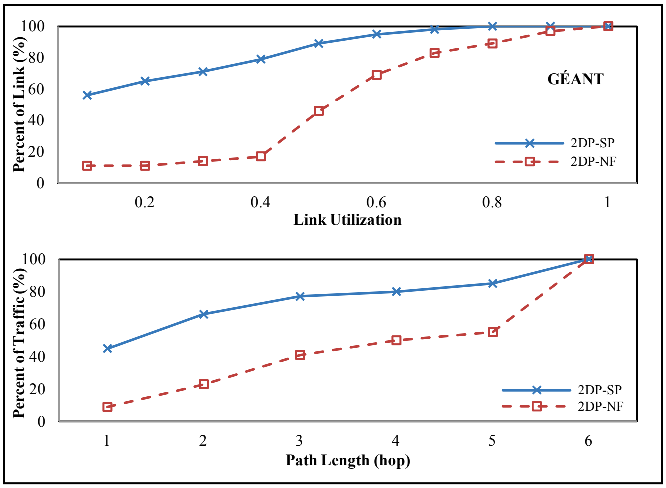

Figure 13. CDF of Link Utilization and Path Length on GÉANT

From Fig. 12 (top), we see that 61% of links using 2DP-SP have utilization no larger than 0.1, which is better as compared to 0% using 2DP-NF. However, while saving 13.3% power usage, 2DP-NF also generates 95% of links with utilization between 0.1 and 0.2, which is better than 2DP-SP that generates only 32% of links with utilization in the range. Further, 2DP-NF generates MLU of 0.3, which is better as compared to 2DP-SP with MLU reaching 0.5. For path length, shown in Fig. 12 (bottom), 5% of traffic demands have MRPL

(7 hops) larger than the network diameter (6 hops) for 2DP-NF while MRPL of all traffics for 2DP-SP is no larger than network diameter.

In Fig. 13 (top), due to off-peak period, 56% of links using 2DP-SP have utilization no larger than 0.1, which is better than the result generated by 2DP-NF, only at 11%. Different from R_Abilene network, 2DP-SP generates MLU value only at 0.8 on R_GÉANT network, which is better than 2DP-NF, up to 1.0. As shown in Fig. 13 (bottom), for 2DP-SP, 45% demands route its traffics through single hop paths, but only 9% demands use single hop routing for 2DP-NF. However, running 2DP-SP and 2DP-NF obtains the same MRPL, which is less than the network diameter, at six hops.

The results for Sprint network are shown in Fig. 14. For link utilization, Fig. 14 (top) shows that 2DP-NF obtains 99% of links with utilization no larger than 0.8, which is only 6% higher than the value generated by 2DP-SP. For routing path length, Fig. 14 (bottom) shows that 2DP-NF increases the length produced by 2DP-SP less as compared to on Abilene and GEANT. In a larger topology, like Sprint, there are more alternative paths for each (s_d, t_d) pair that have routing path length closer to the shortest path. Thus 2DP-NF can produce better results in term of path length on Sprint as compared to on the other two networks. Notice that both 2DP-NF and 2DP-SP obtain the same MRPL, which is equal to the network diameter (8 hops).

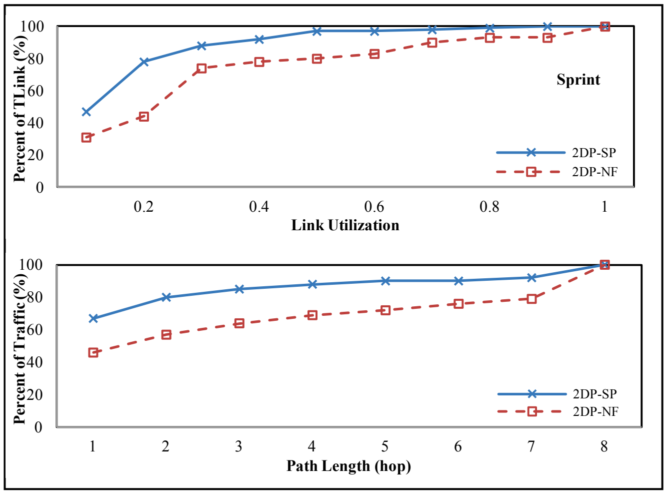

Figure 14. CDF of Link Utilization and Path Length on Sprint

V. CONCLUSION

We have presented a new energy-aware routing problem that aims to maximally switch off unnecessary nodes and links during off-peak periods such that the remaining nodes and links are sufficient to route all traffic demands and that the ratio of using two link-/node-disjoint paths is not less than a given threshold and meets MLU requirement. We have proposed a heuristic technique to solve the problem. Through extensive simulations on both real and synthetic network topologies and traffic demands, we have shown its benefits in reducing the network's energy consumption. In future, we will combine path delay, network reliability and MLU constraints in 2DP routing for energy-aware traffic engineering.

REFERENCES

[1] K. Xiong, Z. D. Qiu, Y. Guo, and H. Zhang. "Multi-constrained shortest disjoint paths for reliable QoS routing". In *ETRI Journal*, vol. 31, no. 5, pp. 534-544, 2009.

[2] Y. Wang, W. Yang, nd J. Wu, "Spare Capacity allocation using partially disjoint paths for dual link failure ". In *DRCN*, Budapest, Hungary, pp. 171-178, 2013.

[3] Y. Guo, F. Kuipers and P. V. Meeghem. "Link-disjoint paths for reliable QoS routing." In *Int. J. Commun. Syst.*, vol. 26, pp. 779-798, 2003.

[4] K. W. Roth, F. Goldstein and J. Kleinman, "Office and telecommunications equipment in commerical buildings-volume I: Energy consumption baseline". In *Technical Report 7289500*, Arthur D. Little, Inc., 2002.

[5] A. Qureshi, "Cutting the electric bill for Internet-scale systems". In *ACM SIGCOMM*, Barcelona, Spain, pp. 123-134, 2009.

[6] G. Lin, S. Soh, M. Lazarescu and K. Chin, "Power-Aware Routing in Network with Delay and Link Utilization Constraints". In HPSR, Taipei, Taiwan, July, 2013.

[7] R. Tucker, J. Baliga, R. Ayre, K. Hinton, and W. Sorin, "Energy Consumption in IP Networks". In *Proc. ECOC Symp. Green ICT*, Brussels, Belgium, pp. 1, 2008.

[8] L. Chiaraviglio, M. Mellia, and F. Neri, "Minimizing ISP Network Energy Cost: Formulation and Solutions". In *IEEE/ACM Transactions on Networking*, vol. 20, pp. 436-476, 2012.

[9] G. Lin, S. Soh, M. Lazarescu and K. Chin, "On the Effects of Energy-Aware Traffic Engineering on Routing Reliability". In *APCC*, Indonesia, August, 2013.

[10] W. Fisher, M. Suchara, and J. Rexford, "Greening backbone networks: reducing energy consumption by shutting off cables in bundled links". In *Green Networking*, New Delhi, India, pp. 29-34, 2010.

[11] M. Zhang, C. Yi, B. Liu, and B. Zhang, "GreenTE: Power-Aware Traffic Engineering". In *ICNP*, Beijing, China, pp. 21-30, 2010.

[12] J. M. Jaffe, "Algorithms for Finding Paths with Multiple Constraints". In *Networks*, vol. 14, pp. 95-116, 1984.

[13] J. Y. Yen, "Finding the K Shortest Loopless paths in a network". In *Management Science*, vol. 17, no. 11, 1971.

[14] J. W. Suurballe and R. E. Tarjan. "A quick method for finding shortest pairs of disjoints paths". In *Networks*, vol. 14, pp. 325-336, 1984.

[15] A. Srinivas and E. Modiano, "Minimum Energy Disjoint Path Routing in Wireless Ad-Hoc Networks". In *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM)*, San Diego, USA, pp. 122-133, 2003.

[16] "Yin Zhang's Abilene TM". In http://www.cs.utexas.edu/~yzhang/research/AbileneTM/.

[17] S. Uhlig, B. Quoitin, J. Lepropre, and S. Balon, "Providing Public Introdomain Traffic Matrices to the Research Community". In *ACM SIGCOMM Computer Communication Review*, Pisa, Italy, vol. 36, no. 1, pp. 83, 2006.

[18] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP Topologies with Rocketful". In *IEEE/ACM Transactions on Networking*, vol. 12, no. 1, pp.2-16, 2004.

[19] S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the Tightrope: Responsive Yet Stable Traffic Engineering". In *ACM SIGCOMM*, PA, USA, pp. 253-264, 2005.

[20] "Spirent Signal Delay Network Emulator – DG/OTU/CPRI". In *Datasheet of Spirent Communications, Inc.*, 2010.