Energy-Aware Routing Algorithms in Software-Defined Networks

Rui Wang, Zhipeng Jiang, Suixiang Gao, Wenguo Yang*
School of Mathematical Sciences,
University of Chinese Academy of Sciences
Beijing, China
*Corresponding author: yangwg@ucas.ac.cn

Yinben Xia, Mingming Zhu

Huawei Technologies Co. LTD.
Beijing, China
{xiayinben, tinny.zhu} @huawei.com

***Abstract*—The feature of centralized network control logic in Software-Defined Networks (SDNs) paves a way for green energy saving. Router power consumption attracts a wide spread attention in terms of energy saving. Most research on it is at component level or link level, i.e. each router is independent of energy saving. In this paper, we study global power management at network level by rerouting traffic through different paths to adjust the workload of links when the network is relatively idle. We construct the expand network topology according to routers' connection. A 0-1 integer linear programming model is formulated to minimize the power of integrated chassis and line-cards that are used while putting idle ones to sleep under constraints of link utilization and packet delay. We proposed two algorithms to solve this problem: alternative greedy algorithm and global greedy algorithm. For comparison, we also use Cplex method just as GreenTe does since GreenTe is state of the art global traffic engineering mechanism. Simulation results on synthetic topologies and real topology of CERNET show the effectiveness of our algorithms.**

Keywords—energy-aware; algorithms; integer linear programming; Software-Defined Networks.

I. INTRODUCTION

The total revenue of global operators is about 2 trillion dollars in 2012, and CAPEX expense 3.2 trillion dollars, OPEX expense 1.36 trillion dollars, while energy expenditure accounts ranging from 15 to 50 percentages in OPEX[1]. Besides huge electricity bills, energy expenditure includes fuel oil, batteries, energy device, and energy maintenance, and so on. Meanwhile, data service of telecommunication network explosively grows, leading to constantly expansion of network scale inevitably. And energy expenditure will rise as network expands. Non-renewable resources such as oil will remain more and more higher price in the future. The black hole of energy effect will become larger, forcing operators to carry out energy conservation to maintain profit.

There are three ways of energy conservation from the perspective of operators:

1. Reduce duplication of unreasonable network through network planning and optimization of resources combination.
2. Purchase energy-efficient equipment.
3. Intelligent sleep or regular optimization to realize dynamic supply of network power according to business.

The first two methods are static. Once implemented, the effect of energy saving is fixed and unable to optimize continually. Most operators and equipment providers take this energy saving manner at present. In this paper, we design energy saving optimization algorithm in Software-Defined Networks (SDNs) from the third way. SDN[2][3] is an innovative network architecture in which the network control plane logic is decoupled from the network forwarding hardware and given to the controller. It allows network control logic to be designed and operated on a global network view. Besides all the benefits above, SDN paves a way for green energy saving for the whole network: it can collect the whole network topology and information of real time traffic of every device; what's more, it can get information of users' requests conveniently by northbound API interface.

In SDN controller, Energy saving mainly involves following components: network topology information module, device statistics module, northbound API (containing users' requests information API) module, energy-aware algorithm module, flow table module, intelligent device management module, Openflow protocol agent module, just as shown in Fig. 1. SDN controller first gets information of topology, traffic statistics and users' requests. When there are multiple requests routing between same origin-destination (OD) pair and the traffic of each request path is low, it move the traffic to a fewer number of paths using energy-aware algorithm. After adjusting the users' traffic to reasonable paths, it put idle links and line-cards to sleep. And this can save considerable energy.

The main contributions of this paper are as follows. We consider how to optimize power management globally at network level, rerouting traffic through different paths to

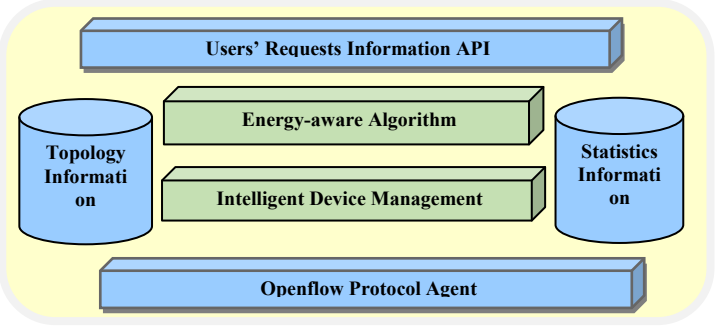


Fig. 1. Energy Saving Components of SDN controller

This work is supported by the National Natural Science Foundation of China (71171189), the key project of the National Natural Science Foundation (11331012), 973 project (2011CB706901) and Huawei Technology Co. Ltd.

adjust the workload of links in an unsplittable way when the network is relatively idle. A 0-1 integer linear programming model is formulated to interpret the goal of minimizing the power of used integrated chassis and line-cards while putting idle ones to sleep under performance constraints such as link utilization and packet delay. We proposed two algorithms to solve this problem: alternative greedy algorithm and global greedy algorithm. Using network topologies and traffic data from synthetic networks and a real network of CERNET, we compare our algorithms with Cplex used in GreenTE since GreenTe is state of the art global traffic engineering mechanism. Simulation results show the effectiveness of our algorithms. And global greedy algorithm can reduce line-cards power consumption by up to 67% of CERNET.

The rest of the paper is organized as follows. Section II shows the related work of energy saving. Section III gives the basic background and its assumptions, and constructs the expand network and formulates the global energy saving problem as a 0-1 integer programming model. Section IV proposes practical solutions with time complexity analysis: alternative greedy algorithm and global greedy algorithm. Time complexity analysis is also presented. Cplex heuristic is built as GreenTE does for a better comparison. Section V shows the experimental results. Section VI gives the conclusion.

II. RELATED WORK

As can be seen from analysis of introduction, the third way of energy conservation is dynamic network energy saving scheme. Since routers are mainly devices running in the existing network, we aim at saving of routers' energy consumption in the whole network. Existing router power optimization technique has focused on two kind of optimization: local optimization and global optimization.

On one hand, most research on router power is locally optimized at component level or link level, treating routers as isolated devices and focusing on reducing power consumption at hardware component level. Gupta et al. [4] considered putting some of the subcomponents such as line cards when they are idle or clock the hardware at lower rates in the Internet. They later explored the idea of uncoordinated sleeping to save energy at link level in Local Area Networks (LANs)[5][6][7] Nedevschi et al.[8] considered creating more opportunities for network components to sleep during idle times by shaping traffic into small bursts and adapting the rate of network operation to the offered workload. However, this kind of local optimization is limited by the inter-arrival time of packets.

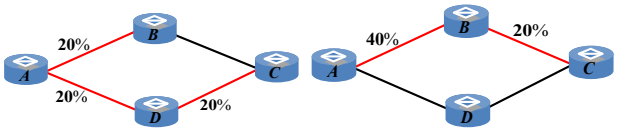
On the other hand, remaining research on router power management is globally optimized at network level. The high path redundancy and low link utilization of today's networks provide opportunities for power-aware traffic engineering. Gupta et al. [4] suggested changing routes during low activity periods so as to aggregate traffic along a few routes while allowing idle devices to sleep. Nonetheless, they did not present actual network level algorithm. Chabarek et al. [9] presented an investigation of potential power saving through network design and routing in wire-line networks without a specific routing design. Heller et al. [10] proposed ElasticTree to manage power by adjusting active links and switches to satisfy changing data center traffic loads using OpenFlow, which was focus on tree-based topologies. It made full use of fat tree topology of data center and simply kept single root tree structure when energy saving. This method was not applicable for general network topology. Vasić et al. [11] proposed EATe to reduce energy consumption of Internet which worked in conjunction with local measures such as sleeping and rate adaptation and required routers coordination. While in SDN, this is not needed because of centralized controller. Zhang et al. [12] proposed a state of the art global traffic engineering mechanism, GreenTE. GreenTE considered how to maximize the total power saving of idle links under maximum link utilization and network delay by rerouting in a splittable way. A Mixed Integer Programming problem was formulated and Cplex, commercial software, was used to solve this problem with a time limit of 300s. This method could not ensure a feasible solution in limited time. They did not propose practical algorithms as well. While our algorithm is more practical and can be scalable to any network topology, especially large complicated network that cannot turn off links manually.

III. NETWORK MODEL AND PROBLEM STATEMENT

A. Basic Background and Assumptions

Global optimization is better than local optimization on energy saving. As shown in Fig. 1, there is only one link can be put to sleep in (a) .While, there are 2 idle links in (b) by rerouting. In this paper, we focus on saving power of router network by global optimization. A router consists of an integrated chassis (include power supply, heat dissipation system, cables etc.), several line-cards and multiple ports. Through power consumption test on one serial of Huawei's equipments, we get power consumption ratio of components of the router as shown in TABLE I.

It is clear that the integrated chassis and line-cards contribute most power consumption of a router. So we concentrate on minimizing the power of used integrated chassis and line-cards during low activity periods while putting idle ones to sleep. Following assumptions are made based on typical router architectures.



(a). local optimization (b).global optimization

Fig. 2. An example of different optimization on energy saving

TABLE I. POWER CONSUMPTION RATIO OF A NETENGINE40E ROUTER

Device	Ratio
100G integrated chassis	56.3%
100G line-card	43.6%
port	0.1%

1. Each port can be connected with another port of any line-card of any router, formulating a physical link. Each port can be only connected to one physical link.
2. A physical link is an actually bidirectional link. Each direction has capacity of C.
3. Any request can get in and out of any port of any line-card when through a router.
4. A port is in use when there is traffic on its connecting link. A line-card cannot put into sleep when at least one port is in use. It can be asleep when there is no traffic through it.
5. Any request must be routed in an unsplittable way. Namely, any request can only be transported via one path.

B. *Problem Statement and Formulation*

Based on assumptions above, we construct a 0-1 integer model to describe this energy saving problem.

1) *Problem Statement*

We model the router network as a directed graph $G(V,A)$, where V is the set of nodes(i.e., routers) and A is the set of links. The capacity, cost, delay of each link $e \in A$ are denoted as c_e, w_e and del_e. There are D requests $LSP = \{(s_i,t_i,d_i,delay_i), i = 1,\cdots,D\}$ routed in the network, where $s_i,t_i,d_i,delay_i$ represent the origin, destination, demand, delay of request i respectively. And request i is transported on path P_i. The power consumption of each integrated chassis and line-cards is given, denoted by PC_v^0, PC_v. The goal is to minimize the power consumption of used integrated chassis and line-cards by rerouting traffic in an unsplittable way so that the maximum link utilization is no more than 50%.

We take a 3-nodes network as simple example to illustrate this problem. Each router has 4 line-cards and connection among line-cards is shown in Fig. 2(b). Assume that the cost and capacity of each link is 1 and 10M respectively. There are 3 requests on the network now. They are routed as follows. The number of used line-cards is 7. Maximum link utilization is 50%.

Lsp1:{A,B,5M}, P_1: link AB， line-cards: A1-B2;

Lsp2:{A,C,5M}, P_2: link AB-BC， line-cards: A2-B4,B3-C1;

Lsp3:{B,C,5M}, P_3: link BC， line-cards: B4 -C4 .

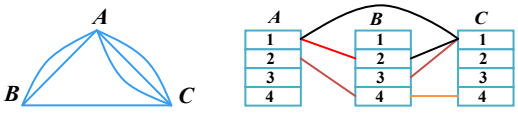
We can reroute the traffic for energy saving as follows.

Lsp1:{A,B,5M}, P_1: link AB， line-cards: A1-B2;

Lsp2:{A,C,5M}, P_2: link AC， line-cards: A1-C1;

Lsp3:{B,C,5M}, P_3: link BC， line-cards: B2-C1.

After rerouting, the number of used line-cards is 3. And maximum link utilization is 50%.



(a).router network (b). inner topology

Fig. 3. A simple example of energy saving problem

2) *Construction of Expand Topology*

In order to formulate the energy saving model, we label each line-card as a node and get the expand network $G' = (V', A')$ according to routers' connection as follows.

For every node $v \in V$, we copy it n_v times, where n_v is the line-cards' number of router v. We connect each copy node to v and get a star graph Star (v), with link cost of 0 and capacity of ∞. The center of the star represents the integrated chassis of the router v. And the leaves denote the line-cards. Connect the stars according to the inner structure of the network, we get the expand network. Fig. 3 shows the expand topology of figure2. We denote red links as $A0$, blue links as $A1$, and copy nodes as $V1$. Denote $|V'|,|A'|$ by n,m.

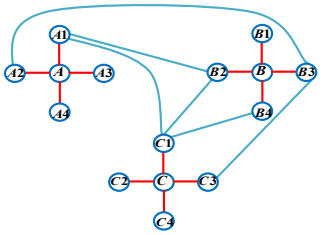


Fig. 4. Expand topology of Fig 2.

3) *The 0-1 Integer Programming Model*

We formulate the model of global energy saving problem as 0-1 integer programming model P1 based on the expand topology. The notation used is shown in TABLE II.

TABLE II. NOTATION USED IN THIS PAPER

Notation	Meaning
y_j	1 if integrated chassis j is in use,0 otherwise
x_v	1 if line-card v is in use,0 otherwise
s_i	The origin of request i
t_i	The destination of request i
d_i	The demand of request i
$delay_i$	The delay threshold of request i
x_v^i	1 if line-card v is used by request i ,0 otherwise
x_{uv}^i	1 if link uv is used by request i ,0 otherwise
c_{uv}	The capacity of link uv
PC_v	The power consumption of line-card v
PC_j^0	The power consumption of integrated chassis j
k	Maximum number of candidate paths for each request
X_{ij}	1 if request i uses the jth candidate path P_j^i, 0 otherwise
δ_{eij}	1 if link e is on the path P_j^i of request i, 0 otherwise

The goal is to minimize the power consumption of used integrated chassis and line-cards. Equation 1 demonstrates that an integrated chassis can be asleep unless all of its line-cards are asleep. That means y_j must be 1 if a

$$\min \quad \sum_{j\in V} PC_j^0 y_j + \sum_{v\in V1} PC_v x_v \qquad P1$$

$$s.t. \quad \frac{1}{|V1|}\sum_{v\in Star(j)} x_v \le y_j, \quad \forall j \in V \qquad (1)$$

$$\frac{1}{D}\sum_{i=1}^{D} x_v^i \le x_v, \quad \forall v \in V1 \qquad (2)$$

$$\sum_{(u,v)\in A} x_{uv}^i - \sum_{(v,u)\in A} x_{vu}^i = \begin{cases} 1, & u = s_i \\ -1, & u = t_i \\ 0, & else \end{cases}, \forall u \in V', i = 1,\cdots,D \qquad (3)$$

$$\sum_{(u,v)\in A} x_{uv}^i \le 1, \qquad \forall u \in N \setminus \{s_i, t_i\}, i = 1,\cdots,D \qquad (4)$$

$$x_{uv}^i \le \frac{1}{2}(x_u^i + x_v^i) \qquad \forall (u,v) \in A1, i = 1,\cdots,D \qquad (5)$$

$$\sum_{i=1}^{D} x_{uv}^i d_i \le 50\% c_{uv} \qquad \forall (u,v) \in A1 \qquad (6)$$

$$\sum_{(u,v)\in A} x_{uv}^i del_{uv} \le delay_i \quad i = 1,\cdots,D \qquad (7)$$

$$y_j, x_v, x_v^i, x_{uv}^i \in \{0,1\}$$

$x_v (v \in Star(j))$ equals 1.Equation 2 ensures that a line-card cannot put to be sleep if at least one request uses it. Equation 3 states the flow conservation constraints. Equation 4 demonstrates that the corresponding path is acyclic. Equation 5 represents that both endpoints are in use if there is traffic on a link. Equation 6 ensures the maximum link utilization constraint. Equation 7 guarantees the delay constraint.

Obviously, this is a 0-1 integer linear programming problem. Generally speaking, ILP problems are NP-hard, thus we cannot solve in polynomial time. Consequently, we propose heuristic algorithms to tackle this problem.

IV. HEURISTIC ALGORITHMS

We propose two heuristic algorithms to solve P1: alternative greedy algorithm and global greedy algorithm. Then we analyze the time complexity of them. And we also formulate a model in order to use Cplex as a comparison.

A. Alternative Greedy Algorithm

The idea of alternative greedy algorithm is based on the original routing of requests, pretend to adjust requests as few as possible. Find the corresponding request that the total consumption is minimum after releasing the traffic of it. Reroute this request in residual network. Renewing paths if the goal value becomes smaller. Repeat the same process until all requests are done. This is one loop. Repeat this until two goal values of adjacent loops are close. Let $f(P)$ denote the total power consumption of the set of paths P .The algorithm is given in Algorithm 1.

Algorithm1. Alternative Greedy Algorithm (AGA)

1. Calculate the initial total power consumption $fbest = f(Paths)$. Let initial set $S = \{1,\cdots,D\}$, $num = 1$;
2. If $|num| \le 10$, go to step 3; else, stop.
3. If $S \ne \phi$, calculate $i^* = \arg\min_{1\le i\le D, i\in S} f(Paths - \{P_i\})$, $S = S \setminus \{i^*\}$,go to step 4; else, $num = num + 1$, go to step 7;
4. Calculate the residual network $G_f^{'}$ for request i^*: release the bandwidth of P_{i^*} , subtract the bandwidth of $Paths - \{P_{i^*}\}$, calculate available bandwidth of each link. Delete corresponding links of if residual bandwidth is lower than d_i . Then let the weight of inbound link of used line-card be 0, other inbound link be PC_v . Process the weight of inbound links of integrated chassis in the same way. Denote the new network as $G_f^{'}$. There are two attributes for each link in $G_f^{'}$: weight and delay.
5. Find the shortest path $P_{i^*}^{'}$ in $G_f^{'}$ using Dijkstra algorithm so that $P_{i^*}^{'} = \arg\min_{P} f\left((Paths - \{P_{i^*}\}) \cup \{P\}\right)$, where P is a path from s_i to t_i under demand and delay constraints. Give priority to weight when using Dijkstra. Select the link with smaller weight if delay constraint is satisfied. Let $fnew = f\left((Paths - \{P_{i^*}\} \cup \{P_{i^*}^{'}\}\right)$;
6. If $gbest > fnew$, $gbest = fnew$, $Paths = (Paths - \{P_{i^*}\}) \cup \{P_{i^*}^{'}\}$, go to step 3;
7. If $fbest \le gbest$, stop; else, $fbest = gbest$, $S = \{1,\cdots,D\}$, go to step 2.

In step 4, the residual available bandwidth of link e in $G_f^{'}$ equals $50\% C_e$ minus the bandwidth of $Paths - \{P_{i^*}\}$ plus bandwidth of P_{i^*} . The shortest path in step 5 means the minimum power consumption path based on the weight of links. Step 2 and 7 are stopping rules. The former represents the loop num limit. The latter represents the process stops if the goal is not improved much.

B. Global greedy Algorithm

The effect of energy saving can be better if there is no limit on adjusting requests as few as possible. So we propose global greedy algorithm. The idea is as follows: sort all requests according to priority order; construct the residual network for each request; reroute request one by one until all requests are arranged. The algorithm is given in Algorithm2.

Algorithm2. Global Greedy Algorithm (GGA)

1. Sort all requests according to priority order. Assume the new order is 1,…, D . Let $fbest = 0$, $Paths = \phi$, $i = 1$;
2. If $|Paths| = D$, stop. $fbest$ is the total power consumption. Else , go to step 3;
3. Construct residual network $G_f^{'}$ of expand network $G^{'}$ for request i : calculate the residual bandwidth for each link, delete corresponding links if residual bandwidth is lower than d_i .Then let the weight of inbound link of used line-card be 0, other inbound link be PC_v . Process the weight of inbound links of integrated chassis in the same way. Denote the new network as $G_f^{'}$. There are two attributes for each link in $G_f^{'}$: weight and delay.

4. Find the shortest path P_i in $G_f^{'}$ using Dijkstra algorithm so that $P_i = \arg\min_P f(P)$, where P is a path from s_i to t_i under demand and delay constraints. Give priority to weight when using Dijkstra. Select the link with smaller weight if delay constraint is satisfied.
5. $fbest = f(Path \cup \{P_i\})$, $Paths = Paths \cup \{P_i\}$, $i = i+1$, go to step 2.

In step 3, the residual available bandwidth of link e in $G_f^{'}$ equals 50% C_e minus the bandwidth used by *Paths*.

C. Time Complexity of Heuristic Algorithms

For Algorithm1, step 1 calculates in $O(D*n)$ in the worst case. In each loop, step 3~6 are repeated at most D times. Step 3 takes $O(D^2*n)$ time. Step 4 requires searching all links in G' and thus has time complexity of $O(Dn+m)$. Step 5 requires running Dijkstra algorithm in $O(n^2)$ time. Thus, Algorithm 1 has worst case time complexity of $O(Dn+10D(D^2n+(Dn+m)+n^2)) = O(D^3n+D(m+n^2))$.

Similarly, Algorithm2 has worst case time complexity of $O(D(Dn+m+n^2)) = O(D^2n+D(m+n^2))$.

D. Using Cplex

For comparison, we also use Cplex to solve this problem just as GreenTe does. This method is based on searching solution in candidate paths instead of all possible paths. First of all, we calculate k-short paths $\{P_1^i, P_2^i, \cdots, P_k^i\}$ as candidate paths for each request i in G' using *k*-shortest paths algorithm[14]. After that, we formulate the model P2 as follows based on candidate paths. Finally, use Cplex to solve P2.

$$\min \quad \sum_{j\in V}^{k} PC_j^0 y_j + \sum_{v\in V1} PC_v x_v \qquad P2$$

$$s.t. \quad \sum_{j=1}^{D} X_{ij} = 1, \ i = 1,\cdots,D \qquad (8)$$

$$\sum_{i=1}^{D}\sum_{j=1}^{k} \delta_{eij} d_i X_{ij} \le 50\% c_e, \ \forall e \in A1 \qquad (9)$$

$$delay(P_j^i) * X_{ij} \le delay_i, i = 1,\cdots,D \qquad (10)$$

$$\eta_e \le \frac{1}{2}(x_v + x_u), \forall e = (v,u) \in A1 \qquad (11)$$

$$\eta_e \ge \frac{1}{D}\sum_{i=1}^{D}\sum_{j=1}^{k} \delta_{eij} X_{ij}, \forall e \in A1 \qquad (12)$$

$$\frac{1}{|V1|}\sum_{v\in Star(j)} x_v \le y_j, \quad \forall j \in V \qquad (13)$$

$$x_v, \eta_e, X_{ij} \in \{0,1\}$$

Each request is required to select only one path from k candidate paths. One change in the model is the addition of variable X_{ij}. If it equals 1, request i uses candidate path P_j^i. Equation 8 ensures the request is unsplittable. Equation 9 and equation 10 are performance constraints. Equation 11 and Equation 12 means a line-card is in use if there is traffic on its corresponding link. Equation 13 is the same as equation 1.

P2 is also a 0-1 integer linear programming model with fewer variables. Using Cplex for solution is time consuming, though the search space is reduced. What's more, Cplex cannot ensure an optimal solution in limited time.

V. Experimental Evaluation

A. Experimental Setup

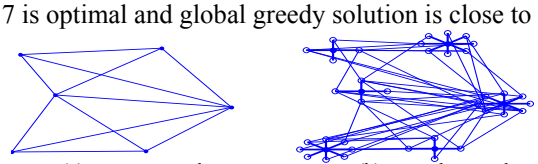
We have run simulations on 3 several synthetic networks with different size and real network of CERNET[14]. Topologies and demand are generated using Matlab, assuming that a router has 4 or 8 line-cards. Greedy algorithms are implemented in VC++. For synthetic networks, the capacity is set to be 5M and demand of requests is uniformly chosen at random from interval [0, 1].

We get a minimum cost routing using greedy algorithm as an initial solution. For sake of testing the effectiveness of our algorithms, we compare alternative greedy algorithm, global greedy algorithm, Cplex method and initial solution. We assume that the power consumptions of different line-cards are the same and target at minimizing the number of used line-cards which is denoted by Lc#.

B. Simulation Results

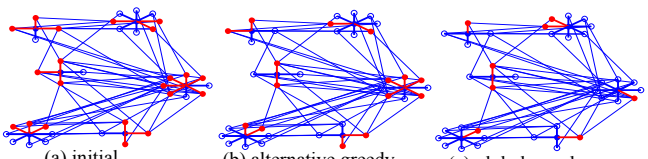
1) Synthetic Networks

The size of synthetic networks is 6, 25 and 400, respectively. Fig. 5 shows a 6 nodes router network with 36 line-cards. The requests' num is 20. Fig.5 and TABLE III. show different results of four solutions. Lc# of initial solution is 21. Global greedy algorithm reduces Lc# by 61% while alternative greedy reduces by 28%. The optimal number of line-cards using Cplex is gradually decreasing when the candidate paths' number k becomes larger. Line-cards' number is still 7 when k is larger than 17. So we can deduce that 7 is optimal and global greedy solution is close to optimal.



(a).router network (b).expand network

Fig. 5. A six nodes network



(a).initial (b).alternative greedy (c).global greedy

Fig. 6. Results for six nodes network

TABLE III. Result for six nodes network using Cplex

k	1	3	4	5	6	7	8	9
Lc#	21	15	14	13	11	11	10	10
k	10	12	14	16	18	20	30	40
Lc#	9	9	8	8	7	7	7	7

We conduct similar experiment on the other two synthetic networks and conclude the results in TABLE IV.Global

greedy solution is always the best compared to other solution. It uses less than half the line-cards' number of initial solution in 25 nodes network. Alternative greedy takes less time than other methods and give a suboptimal choice. As the network size grows, Cplex method becomes time consuming and unfeasible for 400 nodes network. Experiments show that global and alternative algorithms are significantly better than Cplex method and can be used to optimize large networks.

TABLE IV. EXPERIMENTAL RESULTS FOR SYNTHETIC NETWORKS

Networks	**Nodes:6 line-cards:36 requests:20**		**Nodes:25 line-ards:156 requests:131**		**Nodes:400 line-cards:2396 requests:625**	
Solution	**Lc#**	**time**	**Lc#**	**time**	**Lc#**	**time**
Initial	21	<1ms	92	62ms	1636	10.5s
AGA	15	<1ms	80	32ms	1549	4.1s
GGA	8	<1ms	44	63ms	1220	12.3s
Cplex (k=10)	9	600ms	58	12h	--	--

2) CERNET Network

CERNET is the first backbone Internet in China. It has 25 nodes as shown in Fig. 6. We generate 4 line-cards for each common node, 8 for level 1 core nodes and 374 requests.

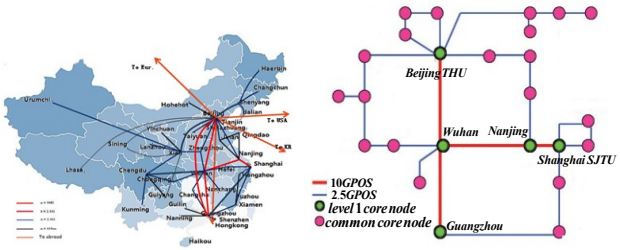


Fig. 7. CERNET Topology

TABLE V. EXPERIMENTAL RESULTS FOR CERNET

Solution	**Lc#**	**time**
Initial solution	71	140ms
AGA	69	47ms
GGA	23	78ms
Cplex (k=10)	58	96s
Cplex (k=20)	56	185s

As shown in TABLE V, global greedy outperforms the other solution. It reduces used Lc# by 67% compared to initial solution. Alternative greedy takes the least time while Cplex takes the most. Alternative greedy solution uses 69 line-cards on account of the special topology. The topology is close to tree topology which makes less choice of different routes for requests. Besides, initial solution of minimum cost routing is a kind of optimized results. This may cause that requests tend to use links with smaller cost. Alternative greedy may perform better when the traffic is relatively decentralized in a strongly connected network. From all these experimental results, we can draw a conclusion that our algorithms can save considerable energy and outperform Cplex method.

VI. CONCLUSION

In this paper, we consider how to optimize router power management globally at network level during low activity periods. One can aggregate traffic by rerouting through different paths to free line-cards as many as possible. We construct the expand network and formulate a 0-1 ILP model to minimize the power of used integrated chassis and line-cards under constraints of link utilization and packet delay and propose alternative greedy algorithm and global greedy algorithm to tackle it. Simulation results based on synthetic and real network show that our algorithms can save considerable energy and outperform Cplex method using in a state of the art global traffic engineering mechanism GreenTE.

Traffic is relatively concentrated after energy saving, which makes the network vulnerable to be effected by link failure and sudden traffic bursts. However, there is no research on energy saving considering the reliability of the network at the same time so far. How to ensure reliability while keeping low energy consuming can be a future task.

REFERENCES

[1] Huang, H.F..New ideas for operators of energy-saving work and comprehensive energy solutions to build green network, Communications World, vol.25, pp. 51, 2013

[2] M. Reitblatt, N. Foster, J. Rexford, and D. Walker. Consistent Updates for Software-Defined Networks: Change You Can Believe In! In Proceeding of HotNets, 2011.

[3] D. Levin, A. Wundsam, B. Heller, Logically Centralized? State Distribution Trade-offs in Software Defined Networks, in Proceedings of ACM SIGCOMM, 2012.

[4] M. Gupta and S. Singh, Greening of the Internet, in Proceedings of ACM SIGCOMM, pp. 19-26, August 2003.

[5] M. Gupta, S. Grover, and S. Singh, A Feasibility Study for Power Management in LAN Switches, in Proceedings of IEEE ICNP, pp. 361-371, October 2004.

[6] M. Gupta and S. Singh, Dynamic Ethernet Link Shutdown for Energy Conservation on Ethernet Links, in Proceedings of IEEE ICC, pp. 6156-6161, June 2007.

[7] M. Gupta and S. Singh, Using Low-Power Modes for Energy Conservation in Ethernet LANs, in Proceedings of IEEE INFOCOM, pp. 2451-2455, May 2007.

[8] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, Reducing Network Energy Consumption via Sleeping and Rate-Adaptation, in Proceedings of USENIX NSDI, pp323-336, 2008,.

[9] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright, Power Awareness in Network Design and Routing, in Proceedings of IEEE INFOCOM, April 2008.

[10] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma,S. Banerjee, and N. McKeown, ElasticTree: Saving Energy in Data Center Networks, in Proceedings of USENIX NSDI, 2010.

[11] N. Vasić and D. Kostić, Energy-Aware Traffic Engineering, in Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking, pp. 169-178, April 2010.

[12] M. Zhang, C Yi, B Liu, B Zhang, GreenTE: Power-Aware Traffic Engineering, In Proceedings of ICNP. Pp. 21-30, October 2010.

[13] CPLEX: High-performance software for mathematical programming and optimization,"[Online]. Available: http://www.ilog.com/products/cplex/

[14] D Eppstein, Finding the k Shortest Paths, SIAM Journal on Computing, Vol.28, pp. 652-673, 1998

[15] CERNET. Available: http://nc.tju.edu.cn/Cernet/Cernet/cernettopo