

# Energy-Aware Networks: Reducing Power Consumption By Switching Off Network Elements

Luca Chiaraviglio \*, Marco Mellia \*, Fabio Neri \*

\* Dip. di Elettronica, Politecnico di Torino, Italy, Email: {last name}@tlc.polito.it

**Abstract**—According to several studies, the power consumption of the Internet accounts for up to 10% of the worldwide energy consumption, and several initiatives are being put into place to reduce the power consumption of the ICT sector in general. To this goal, we propose a novel approach to switch off network nodes and links while still guaranteeing full connectivity and maximum link utilization. After showing that the problem falls in the class of capacitated multi-commodity flow problems, and therefore it is NP-complete, we propose some heuristic algorithms to solve it. Results show that it is possible to reduce the number of links and nodes currently used by up to 25% and 10% respectively while offering the same service quality.

**Keywords:** green networks, power-aware, optimization

## I. INTRODUCTION

Power consumption in general, and of ITC technologies in particular, has become a key issue during the last few years. The ratio of power demand versus power resources is constantly growing, and energy costs are increasing at a constant rate. The electricity jumped up of about 35% in Italy during the period 2004-2007 [1]. Moreover, Green House Gases (GHG) emissions have a negative impact on the world climate change [2], and people are becoming more conscious about the problems that will arise in the near future due to this.

According to a number of studies, ICT alone is responsible for a percentage which varies from 2% to 10% of the world power consumption [3], due to the ever increasing diffusion of electronic devices. In this scenario, the power consumption of telecommunication networks, and of the Internet in particular, is not negligible. For example, considering a data center, the network infrastructure alone is responsible of 23% of the overall power consumption, even without taking into account the energy necessary for equipment cooling [4].

The study of power-saving network devices has been introduced over these years, starting from the pioneering work of [6]. In [7] the ideas of Adaptive Link Rate (ALR) and protocol proxying are proposed. The ALR technique is the capability to dynamically adapt the channel transmission rate according to the actual traffic demand. Proxying instead leverages on the idea of turning off high performance devices during low activity periods while moving offered services to low power (and low performance) components. Both these techniques require the change of protocols, and both consider a single pair of devices, i.e. routers sharing the same link, or a couple of high/low performance servers.

More recently, some effort was devoted to investigate how to reduce the power consumption of the entire network infrastructure, and not of single components only. In [5] some simple measurements about power consumption of networking devices are first presented; then authors consider a network topology and evaluate the total network consumption given the power requirement of each element. They consider two scenarios: in the first one all devices are turned on, while in the second one only the minimum number of elements are actually powered on to guarantee the service. The reduction of the corresponding power consumption is finally evaluated.

In this paper, we consider a wide area network scenario. Given the network topology and a traffic demand, we evaluate the possibility to turning off some elements (nodes and links) under connectivity and Quality of Service (QoS) constraints. The goal is to minimize the total power consumption of a large network, in which usually resource overprovisioning is large. We investigate some simple optimization algorithms. In particular, we selectively power off nodes and links of the topology following different strategies. Results show that it is possible to reduce the percentage of powered nodes and links up to 10% and 25% respectively, while guaranteeing that the resource utilization is still below a given threshold. e.g., 80%.

The paper is organized as follows: Sect. II presents the scenario and the problem formulation; Sect. III describes the implemented heuristics, and results are presented in Sect. IV. Finally, conclusions are drawn in Sect. V.

## II. PROBLEM FORMULATION

An informal description of the design problem studied in this paper is the following:

Given

i) a physical network topology comprising routers and links, in which links have a known capacity, ii) the knowledge of the average amount of traffic exchanged by any source/destination node pair, iii) the maximum link utilization that can be supported, iv) the power consumption of each link and node,

Find

the set of routers and links that must be powered on so that the total power consumption is minimized,

Subject to

flow conservation and maximum link utilization constraints.

We provide an Integer Linear Programming (ILP) formulation of the problem to precisely define it. Let us represent the network infrastructure as a graph  $G = (V, E)$ , where  $V$  is the set of vertices and  $E$  is the set of edges. Vertices represent network nodes, while edges represent network links, being  $N = |V|$  and  $L = |E|$  the total number of nodes and links respectively. Let  $c_{ij}$  be the capacity of link from node  $i$  to node  $j$  and let  $\alpha \in \{0, 1\}$  be the maximum link utilization that can be tolerated. Let  $t^{sd}$  be the average amount of traffic going from node  $s = 1, \dots, N$  to node  $d = 1, \dots, N$ , i.e.,  $\{t^{sd}\}$  represents the traffic demand.

Let  $x_{ij} \in \{0, 1\}$ ,  $i = 1, \dots, N$ ,  $j = 1, \dots, N$  be a binary variable that takes the values of 1 if link from node  $i$  to node  $j$ , i.e.,  $(i, j)$ , is present and powered on. Similarly, let  $y_i \in \{0, 1\}$ ,  $i = 1, \dots, N$  be a binary variable that takes the value of 1 if node  $i$  is powered on. Let  $f_{ij}^{sd}$  denote the amount of flow from  $s$  to  $d$  that is routed through the arc from  $i$  to  $j$ . Similarly, let  $f_{ij}$  be the total amount of traffic flowing on the link from  $i$  to  $j$ .

Finally, let  $\mathcal{P}\mathcal{L}_{ij}$  and  $\mathcal{P}\mathcal{N}_i$  be the power consumptions of the link from  $i$  to  $j$ , and of node  $i$ , respectively.

Given the previous definitions, it is possible to formalize the problem as follow:

Minimize

$$\mathcal{P}_{tot} = \sum_{i=1}^N \sum_{j=1}^N x_{ij} \mathcal{P}\mathcal{L}_{ij} + \sum_{i=1}^N y_i \mathcal{P}\mathcal{N}_i \quad (1)$$

Subject to:

$$\sum_{j=1}^N f_{ij}^{sd} - \sum_{j=1}^N f_{ji}^{sd} = \begin{cases} t^{sd}, & \forall s, d, i = s \\ -t^{sd}, & \forall s, d, i = d \\ 0, & \forall s, d, i \neq s, d \end{cases} \quad (2)$$

$$f_{ij} = \sum_{s=1}^N \sum_{d=1}^N f_{ij}^{sd} \quad \forall i, j \quad (3)$$

$$f_{ij} \leq \alpha c_{ij} x_{ij} \quad \forall i, j \quad (4)$$

$$\sum_{j=1}^N x_{ij} + \sum_{j=1}^N x_{ji} \leq M y_i \quad \forall i \quad (5)$$

Eq. (2) states the classical flow conservation constraints, while Eq. (3) evaluates the total flow routed on each link. Constraint (4) forces the total link offered load to be smaller than  $\alpha$ , while constraint (5) states that a node can be turned off only if all incoming and outgoing links are actually turned off. The big-M method is used to force this constraint,  $M \geq 2N$ .

The presented formulation falls in the class of capacitated multi-commodity minimum cost flow problems (CMCF) [8], i.e., the problem in which multiple commodities have to be routed over a graph with capacity constraints. CMCF problems are known to be NP-hard, so exact methods can only be used to solve trivial cases. In this paper, we therefore propose some simple heuristics in order to solve the design problem also for large networks. Moreover, since  $\mathcal{P}\mathcal{L}$  and  $\mathcal{P}\mathcal{N}$  are

```
//node optimization
sort_nodes(vect_nodes);
for (i=0; i<N; i++) {
    disable_node(vect_nodes[i]);
    compute_all_shortest_path();
    compute_all_link_flow();
    if (check_paths() == false) {
        enable_node(vect_nodes[i]);
        continue;
    }
    if (check_flows() == false) {
        enable_node(vect_nodes[i]);
        continue;
    }
}
//link optimization
sort_links(vect_links);
for (j=0; j<L; j++) {
    disable_link(vect_links[j]);
    compute_all_shortest_path();
    compute_all_link_flow();
    if (check_paths() == false) {
        enable_link(vect_links[j]);
        continue;
    }
    if (check_flows() == false) {
        enable_link(vect_links[j]);
        continue;
    }
}
```

Fig. 1. The pseudo-code description of the proposed algorithms.

difficult to know and vary widely depending on the considered technology, in the following we consider  $\mathcal{P}\mathcal{N} = 1$  and  $\mathcal{P}\mathcal{L} \ll \mathcal{P}\mathcal{N}$  [7], so that the objective function can be pursued by trying to switch off the largest possible number of network elements.

### III. ALGORITHMS

The algorithms we propose consider a network in which all elements are powered on, so that  $x_{ij} = 1 \forall i, j$  and  $y_i = 1 \forall i$ . Each algorithm then iteratively tries to switch off each element (either a node or a link). At each step, traffic is then rerouted on the shortest path for each  $(s, d)$  pair to verify Eq. (2), and the utilization constraint (4) is checked for all links. If no violation is present, then the selected element is powered off. Fig. 1 reports a schematic description of the algorithms.

We implemented two different kinds of algorithms: node-oriented and link-oriented heuristics. We expect that it is more difficult to turn off a node than a single link, but the energy saving introduced in the former case is much larger, as reported in [7]. The two heuristic approaches are therefore combined

so that the nodes are checked first, and then links are possibly powered off at a second stage.

Several policies can be adopted to iterate through the node set. We implemented the following ones:

- random (R)
- least-link (LL)
- least-flow (LF)

According to each heuristic, the node set is first sorted considering a given rule before iterating through all the nodes. In particular, the least-link heuristic sorts the nodes according to the number of links that are sourced and sinked at each node, so that nodes with a smaller number of links are checked first, i.e.,  $V$  is sorted in increasing value of

$$X_i = \sum_{j=1}^N x_{ij} + \sum_{j=1}^N x_{ji}.$$

The least-flow heuristic takes instead into account first the nodes with the smallest amount of information flowing through them, i.e.,  $V$  is sorted in increasing value of

$$F_i = \sum_{j=1}^N f_{ij} + \sum_{j=1}^N f_{ji}.$$

Finally, the random heuristic sorts nodes in random order.

Similarly, considering link heuristics, we implemented two algorithms:

- least-flow (LF)
- random (R)

which leverage on the same intuition as the corresponding node sorting heuristics: the least-flow policy sorts links in increasing order of carried flow, i.e.,  $E$  is sorted in increasing value of  $f_{ij}$ , while the random policy sorts links in random order.

All possible node-link sorting heuristics have been studied. Besides these heuristics, we also tested the corresponding ones in which a decreasing order is adopted. Since they all perform consistently worse, we decide not to consider them in this paper.

#### IV. PERFORMANCE COMPARISON

In order to assess the performance of the proposed heuristics, we consider a simple Wide Area Network scenario. The goal is to show that, for a given (static) traffic demand, it is possible to power off a number of network elements, and to still guarantee full connectivity between sources and destination, while enforcing that the link utilization remains smaller than a QoS threshold.

We suppose the network follows a hierarchical topology, which is typical of WANs. All links are supposed to be bidirectional links, so that if link  $(i, j)$  exists, then link  $(j, i)$  exists as well. Three levels of nodes are considered: core, edge and aggregation nodes.

The network core is composed by few nodes that are highly interconnected by means of high-capacity links. Each link

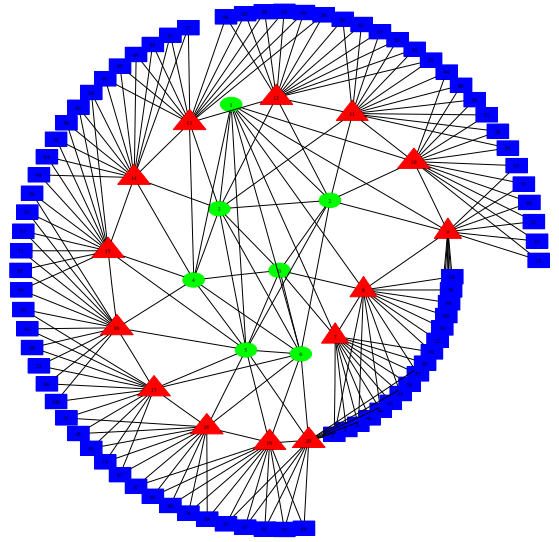


Fig. 2. An example of random topology.

connects nodes which may be also geographically far away, e.g., optical links connecting different cities.

The edge nodes are instead used to interconnect aggregation nodes to the core nodes. Links have middle-range capacity, i.e., smaller capacity than the one of links interconnecting core nodes. Each edge node is connected to some of the closest core nodes, and to other edge nodes. One or more edge nodes can be present in cities, and they collect traffic from aggregation nodes spread within the city boundaries.

The last level of nodes is composed by the aggregation nodes, to which users are directly connected. A Digital Subscriber Line Access Multiplexer (DSLAM) is a typical example of aggregation node. Each node is dual-homed, i.e., it is connected to the closest pair of edge nodes (to guarantee alternate paths in case of failure). The links that connect aggregation nodes to edge nodes have low capacity, i.e., smaller capacity than the one of links interconnecting edge nodes.

Considering the link capacity assignment policy, three classes of links are defined: high, middle-range and low capacity links. Each class has a minimum capacity  $c_{ij}^{min}$  constraint, that was selected to be 15, 5, and 1 units of traffic respectively. Minimum link capacities are also used as link routing weights, so that the routing cost is inversely proportional to the link capacity. This is commonly adopted to force the traffic to be routed through the edge and the core nodes, rather than through aggregation nodes (which are connected by means of low capacity links). A simple minimum cost path is considered as routing algorithm, similarly to what is commonly adopted in the Internet. Furthermore, a QoS constraint is considered, that forces the total traffic flowing through a link to be smaller than a over provisioning factor  $\beta = 0.5$ . Therefore, after routing all the traffic, link capacities are finally assigned so that:

$$c_{ij} = \min(\lceil f_{ij} / \beta \rceil, c_{ij}^{min})$$

Results presented in this paper have been obtained consider-

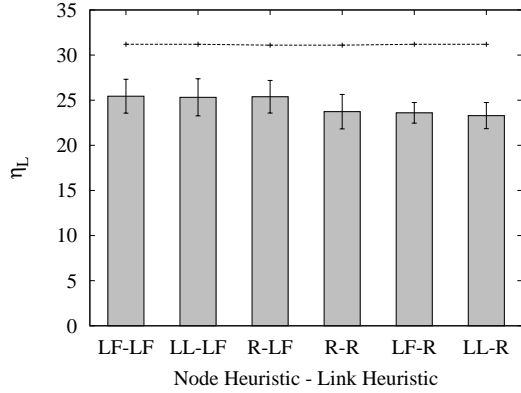


Fig. 3. Comparison of the percentage of links switched off considering different algorithms.

ing randomly generated hierarchical topologies in which 160 nodes are considered. In particular, 10 core nodes, 30 edge nodes, and 120 aggregation nodes are considered. Nodes are assumed to be placed on a plane. Core nodes are randomly connected to other core nodes with probability  $p = 0.5$ . Each edge node is then connected to the two closest core nodes and to another randomly selected close edge node. Finally, aggregation nodes are connected to the two closest edge nodes. An example of the topology obtained is presented in Figure 2. Aggregation, edge and core nodes are represented by squares, triangles and circles respectively.

Only aggregation nodes are possible traffic sources and sinks. For the sake of simplicity, we consider a uniform traffic pattern, so that  $t^{sd} = U[0.5, 1.5]$  units of traffic if  $s, d$  are aggregation nodes;  $t^{sd} = 0$  otherwise.

#### A. Simulation Results

For each considered heuristics, we collected the percentage of links and nodes that are turned off,  $\eta_L$  and  $\eta_N$  respectively. This test was repeated on 20 randomly generated topologies and traffic patterns. Fig. 3 and Fig. 4 show the comparison of the different heuristics by reporting  $\eta_L$  and  $\eta_N$  respectively. Bars report mean values, while the error bars show the standard deviation. Labels on the x-axis report the node-link heuristic combination. A maximum link load factor  $\alpha = 0.8$  was considered. We consider the same traffic demand, so that the network must guarantee to transport the same amount of traffic.

We report also an upper-bound obtained by relaxing constraint (4), so that only the flow conservation constraint are imposed. This is equivalent to find the minimum set of nodes and links that permit to route all the offered flows. This allows to better assess the impact of the QoS constraint, and the quality of the solutions generated by the proposed heuristics.

Considering the links off (Fig. 3), we can see that it is possible to actually turn off about 25% of links in the considered network and traffic scenario. The node selection heuristics show very similar results, while a larger impact of the link heuristics is shown. Indeed, random link selection

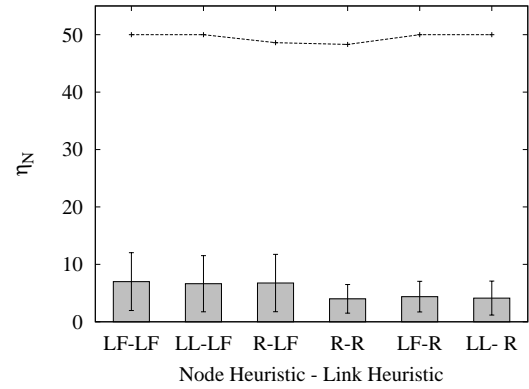


Fig. 4. Comparison of the percentage of nodes switched off considering different algorithms.

heuristics (R-R, LF-R, LL-R) show consistently worse results compared to the least flow selection policy (LF-LF, LL-LF, R-LF). Notice also that the best performing algorithm is only 7 percentage points below the upper bound, which shows that little improvement is possible.

Fig. 4 reports instead the average number of nodes that different heuristics are able to switch off: in this case, only 5-10% of nodes can actually be turned off, since Eq. (5) must be verified. Also in this case there is little impact on the node selection policy, while the LF link selection policy generally performs better. Notice that the upper bound is much higher than any admissible solution, suggesting that the QoS constraint (4) cannot be relaxed.

#### B. Parameter Impact

We performed a study on the impact of the  $\alpha$  parameter, in order to observe the possible range of network elements that can be successfully switched off while guaranteeing a maximum offered load on links. For sake of simplicity, only mean values are reported for each of heuristic combination.

Fig. 5 reports the number of links switched off for  $\alpha \in [0.5, 1]$  in the considered scenario. All algorithms show large improvements for  $\alpha$  up to 0.8; after that, little improvement is noticeable, and a final minor decrease in the average percentage of links that can be turned off is observed for values of  $\alpha > 0.8$ . This is due to the fact that when  $\alpha$  is higher, a larger number of nodes can actually be switched off (see Fig. 6). This reduces the freedom of turning off other links, since not many alternate paths remain available. Fig. 6 shows the average percentage of nodes that are switched off for different algorithms. Similar considerations hold also in this case.

The last set of experiments investigate the impact of the network size on the capability of switching off some elements. Different network topologies were generated, with increasing number of nodes  $N$ . In particular, the relative ratio between core, edge and aggregation node is constant, so that  $x$  node are core nodes,  $3x$  are edge nodes, and  $12x$  are aggregation nodes respectively ( $N = x + 3x + 12x$ ). The aim is indeed

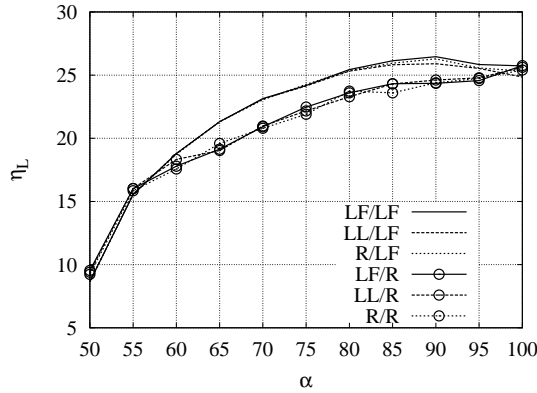


Fig. 5. Percentage of links switched off versus  $\alpha$ .

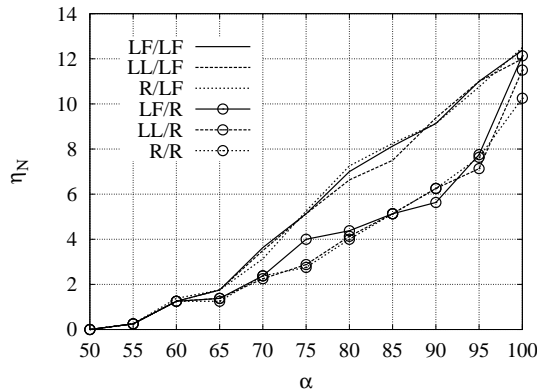


Fig. 6. Percentage of nodes switched off versus  $\alpha$

to generate topologies that have similar characteristics, but different number of nodes.

Traffic is supposed to be uniformly distributed among aggregation nodes only, and routing weights and capacity assignment are performed as previously described. The Least Flow - Least Flow algorithm is adopted to observe the percentage of nodes and links that can be switched off, considering  $\alpha = 0.8$ . Results are averaged over 5 different runs.

Fig. 7 shows the variation of the percentage of links switched off for increasing values of  $x$ . Interestingly,  $\eta_L$  is independent from  $x$ ; indeed, about 25% of links are switched off for all the considered networks. On the contrary, Fig. 8 shows the results considering  $\eta_N$ . In this case, larger values of  $x$  lead to smaller values of  $\eta_N$ ; indeed the percentage of nodes that are switched off decrease from 11% for  $x = 6$  to 3% for  $x = 30$ . The intuition suggests that the distance between nodes increases as  $O(\log N)$ , so that the amount of traffic that is rerouted on alternate paths when a node is switched off increases the offered load on several links and nodes, making it difficult to completely switch off nodes.

## V. CONCLUSIONS AND FUTURE WORK

In this paper we faced a network design problem. We deviated from the traditional formulations of the problem,

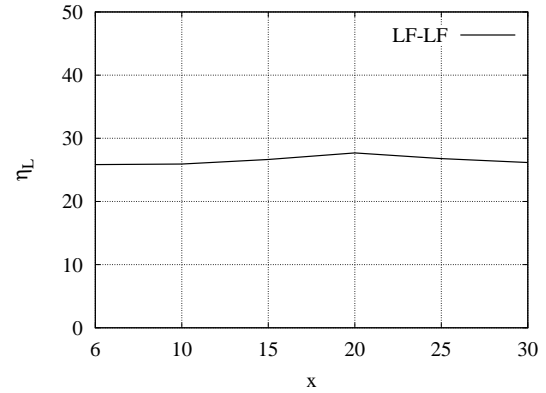


Fig. 7. Percentage variation of links switched off versus different values of  $x$ . The  $LF - LF$  algorithm is considered.

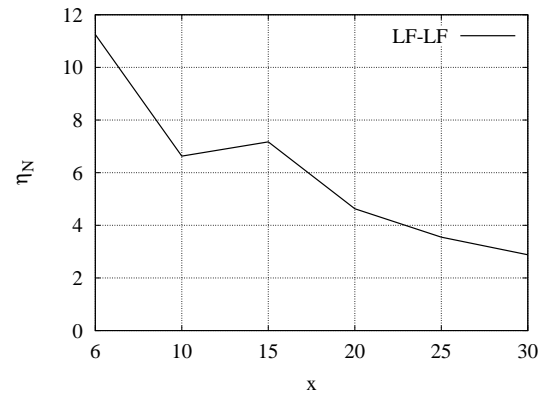


Fig. 8. Percentage variation of nodes switched off versus different values of  $x$ . The  $LF - LF$  algorithm is considered.

in which the objective function is to minimize cost or maximize performance, by considering the minimization of the total power consumed by the network as objective function, while connectivity and maximum link utilization are taken as constraints.

We provided an integer linear programming formulation of the problem, which shows that it is a NP-complete problem. Simple heuristics have been proposed, and their performance assessed considering some simple yet realistic traffic and network scenarios. Results (although dependent in absolute values from the chosen scenario) show that it is possible to switch off both full nodes and links, so that a the total network power consumption can be reduced.

As future work, we plan to evaluate the power saving that can be achieved during off-peak hour, in which the traffic demand is much smaller, so that is possible to reroute traffic on the spare capacity and switch off a large number of nodes and links.

## REFERENCES

- [1] Eurostat Web Page, <http://epp.eurostat.ec.europa.eu/>.

- [2] Katalin Szomolnyi, "GreenHouse Gas Effect of Information and Communication Technologies", *ETNO Project Study*, <http://www.etno.be/>, 2005.
- [3] "An inefficient truth", *Global Action Plan Report*, <http://www.globalactionplan.org.uk/>, December 2007.
- [4] "Where Does Power Go?", <http://www.greendataproject.org/>, January 2008.
- [5] P. Barford, J. Chabarek, C. Estan, J. Sommers, D. Tsiang, S. Wright, "Power Awareness in Network Design and Routing", *IEEE INFOCOM 2008*, Phoenix, USA, April 2008.
- [6] M. Gupta, S. Singh, "Greening of the Internet", *Proceedings of ACM SIGCOMM*, Karlsruhe, Germany, August 2003.
- [7] K. Christensen, C. Gunaratne, B. Nordman, "Managing Energy Consumption Costs in Desktop PCs and LAN Switches with Proxying, Split TCP Connections, and Scaling of Link Speed", *International Journal of Network Management*, Vol. 15, No. 5, pp. 297-310, September/October 2005
- [8] T. G. Crainic, M. Gendreau, I. Ghamlouche, "Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design", *Technical report CRT-2001-01*, Centre de recherche sur les transports, Universite de Montreal, 855.