MASTER II IFI (Informatique : Fondements et Ingénierie)
Parcours UBINET – Ubiquitous Networking and Computing

Rapport de Stage de Fin d'études 2011

MINIMIZATION OF NETWORK POWER CONSUMPTION WITH WAN OPTIMIZATION

Stagiaire :
PHAN Truong Khoa

Maître de stage :
Prof. Joanna Moulierac

Tuteur Enseignant :
Prof. Karima Boudaoud

Equipe-Projet : Mascotte

Laboratoire de Recherche : INRIA et I3S (CNRS/UNS)

1er Mars 2011- 31 Août 2011

Minimization of network power consumption with WAN Optimization

August 26, 2011

Abstract: Reducing energy consumption has become a key issue for industry, and therefore it is also an important field of networking research. Recently, many studies of energy-aware routing have been conducted. The purpose of these works is to aggregate traffic demands over a subset of the network links, allowing other links to be turned off to save energy. Since energy-aware routing uses fewer number of links at any moment, it is important to make sure that links are not overloaded. As a result, link capacity is the main constraint in energy-aware routing problem. In this work, we propose a new energy-aware routing model with the support of WAN Optimization Controller (WOC) - a device that can compress or eliminate redundant data traffic. Using the WOC, the capacity of network links is virtually increased, providing more space to aggregate traffic flows on the WOC-enabled links. We consider a simplified architecture that the router in the future can integrate the WOC (called WOC-router). Based on this architecture, we propose algorithms for the three following cases: (1) All routers on the network are the WOC-routers, we can enable or disable the WOC in routers on demand; (2) There are some WOC-routers that have already been placed on the network, we try to use both these WOC-routers and the normal routers on the network for traffic aggregation; and (3) There are a limited number of WOC-routers, hence we try to find the best location for them on the network. We first formally define the problem and model it as an Integer Linear Program. Then, we propose greedy heuristic algorithms and present simulation results on basic network (square grid) and on some real network topologies. The results show that our algorithms achieve close to the optimal solution. In addition, our new model can save more energy than the existing energy-aware routing without the WOC-routers.

Key-words: power consumption, energy-aware routing, WAN Optimization, traffic redundancy elimination, integer linear programming, algorithms.

Contents

1 Introduction

According to several studies, the emission of CO_2 produced by the Information and Communication Technology (ICT) has been estimated to an approximation of 2% of the total man-made emissions [10]. In fact, this ratio rises up to 10% for some developed countries [29]. Moreover, the studies also show that switches, hubs and routers account for 6 TWh (it costs an estimated 0.5 - 2.4 billion dollars) per year in United States. Therefore, reduction of energy is becoming an important field of research not only on hardware but also on networking technology and protocol.

Some recent studies [3][28] exhibit that the traffic load of routers only has a small influence on their energy consumption. Instead, the dominating factor is the number of active network elements on the network devices such as interfaces (or ports), line cards, base chassis, etc. Therefore, in order to minimize energy consumption, as few network elements as possible should be used while preserving connectivity and QoS. Intuitively, there are multiple paths between a pair of source-destination in the network. When the traffic volume on each link is low, we can aggregate the traffic into fewer links so that the other links do not need to carry any traffic. Routers then can turn off the idle links (or precisely, the network interfaces of the two routers at the ends of the links) for energy reduction. Although today's routers cannot turn off the line cards and bring them back to active state quickly, we believe that these advances will come in the future, especially if they offer a big energy savings.

In general, the network designers usually build many redundant links and over-provision link bandwidth to accommodate traffic bursts and to allow rerouting when links fail. Therefore, to ensure QoS, the existing works on energy-aware routing always limit the maximum utilization of any link (for instance, maximal link utilization is set to 50% in [17][15][27]). As a result, link capacity is the main constraint when aggregating traffic flows to a subset of links. In this work, we utilize the WAN Optimization Controller (WOC) - a device that can compress or eliminate redundant data traffic, and hence virtually increase capacity of the links. Therefore, with the support of the WOC, more traffic flows can be aggregated and thus, more links can be turned off to save energy. However, based on our real experiments, we notice that when a WOC is enabled, it consumes around more 30 watt second in comparison with no WOC. Therefore, in order to evaluate the global power consumption of the network, we should consider both the number of active links and the WOC-routers.

In summary, the contributions of this work are the following:

- We introduce state-of-the-art WOC and the techniques used to reduce traffic load on the network.
- We formally define the problems of energy-aware routing with WOC and model them as Integer Linear Program (ILP).
- Energy-aware routing problems are known to be NP-hard [19][18], and therefore finding the optimal solution becomes impractical even for small networks. Therefore, we present greedy heuristic algorithms that can be used for large networks. The heuristics are validated by comparing with the results given by the ILP.
- By simulation, we present the energy savings on a set of basic and real network topologies. We exhibit that at least one third of network interfaces can be turned off in general case. In addition, we also discuss the impact of energy-aware routing on route length.

The rest of this report is structured as follows:

- We summarize related work in Section 2.
- In Section 3, we present WOC functionality and the compression techniques that can be used on router to reduce traffic load.
- We model the problems using ILP, and then propose greedy heuristic algorithms and present simulation results in Section 4.
- We have conclusions and discuss the future work in Section 5.

2 Related work

Measurement of energy consumptions. Several empirical measurements showed that energy consumption of network devices is largely independent of their load. In particular, in [28], the authors were interested by routers' energy consumption. They observe that for the popular Cisco 12000 series, the consumption at a load of 75% is only 2% more than at an idle state (770W vs. 755W). In [3], the authors show through experiments that the power consumption depends on the number of active ports. Hence, explicitly disabling unused ports on a line card can reduce the device power consumption [3].

Energy minimization. The work on energy consumption in the Internet has been first evoked as a hypothetical working direction by Gupta et al. in 2003 [13]. The authors proposed putting unused network interfaces into sleep mode to save energy. Besides, the impacts on fault tolerance and the required changes to routing protocol of this approach were also discussed. Recently (2008 - present), researchers started to massively invest their effort in this research area [8][22]. Chabarek et al. in 2008 [28] modeled the problem as an ILP and showed how much energy can be saved on different network topologies. In [17], the problem was formulated using ILP and some heuristics, that progressively switch off routers and links, were examined. In [19], the authors proved that there is no polynomial-time constant factor approximation algorithm for this problem. They gave theoretical bounds for specific network topologies and present heuristic to find solution close to the optimal one. In [15], the authors introduced a power-aware traffic engineering model. The problem was first modeled as ILP, then they presented heuristic by refining the problem formulation and considering practical constraints such as maximal link utilization, maximal path length and load balancing.

In our work, we also work on the direction of energy-aware routing. However, different from existing work, we try to use the link capacity in a more efficient way by compressing traffic redundancy on the network. We introduce in next Section the state-of-the-art the techniques used to reduce traffic load on the network.

3 Reducing traffic load in network

Internet traffic exhibits large amount of redundancy when different users access same or similar contents. Therefore, several works [2][6][12][24][25][26][9] have explored how to eliminate the redundant traffic in the network, ranging from object-level to packet-level approach.

- *Object-level approach:* for example, the classical Web caches work at object level to serve popular HTTP objects locally [9]. In similar spirit, CDN [1] and peer-to-peer caches [20] also perform object-level duplicate removal.

- *Packet-level approach:* Spring et al. [26] developed the first system which can remove redundant bytes from any traffic flow on the network. They call this approach as *protocol independent techniques* because it operates below the application layer and attempts to remove any redundant byte that appears on the network. Following this approach, several commercial vendors have introduced "WAN Optimization Controller (WOC)" devices which can remove duplicate content from network transfer [4][6][21]. WOC devices are installed in individual sites of small ISPs and enterprises to offer end-to-end redundancy elimination between pairs of sites. In recently (2008 - present), the success of WAN deployment has motivated researchers to explore the potential of network wide redundancy elimination (RE). For example, Anand et al. [25][2][24] have considered the benefits of deploying packet-level RE in routers across the entire Internet and they have shown that packet-level RE is more effective than the object-level one. In our work, we consider in more detail the WAN Optimization and the RE integrated in routers.

3.1 WAN Optimization Controller

WAN Optimization Controller (WOC) is a device installed at access links of small ISPs or enterprises to improve network performance for WAN links. From the network perspective, a large enterprise usually has three different kinds of offices inside the network: branch offices, regional offices and data centers. The branch offices usually connect to the regional offices and the data centers by WAN links with low bandwidth, high latency. Therefore branch offices are the ones suffering most from a poor network performance. One solution to improve the performance over WAN connections is to pay more money to buy higher bandwidth for the WAN links. However, WOC is another approach to overcome the transport and link capacity limitations. It includes many techniques working together such as application acceleration, TCP acceleration, data compression, data suppression, etc... [11]. In our work, we focus on the two techniques used in WOC to reduce traffic load: data compression and data suppression.

3.1.1 Data compression

Traditional data compression is a technique to encode data that consists of fewer bits than the original data representation. Each packet can be compressed by a compression algorithm such as Lempel-Ziv or DEFLATE [11]. For instance, the DEFLATE algorithm replaces repeated string with pointers and further uses Huffman coding to efficiently encode symbols that frequently occur. This relies on the assumption that both the senders and the receivers use the same compression algorithm. However, it is well-known that DEFLATE does not compress small packets well [2], therefore it can not significantly reduce traffic load on the network. In this report, we focus on data suppression - a technique used in WOC to eliminate redundant data traffic.

3.1.2 Data suppression

Data suppression is also commonly called packet-level redundancy elimination. The principle of data suppression is to detect patterns of data that have been sent over the network. As shown in Fig. 1, the patterns of previously sent data are stored in the database of the accelerators (or WOC) at both the sending and the receiving side. Whenever the accelerator on the sending side notices the same kind of data pattern coming from the sending host, it sends a small signature instead of the original data. The receiving accelerator then recovers the original data by looking up the signature in its database. Because signatures are only a few bytes in size, thus sending signatures instead of actual data gives significant bandwidth savings.

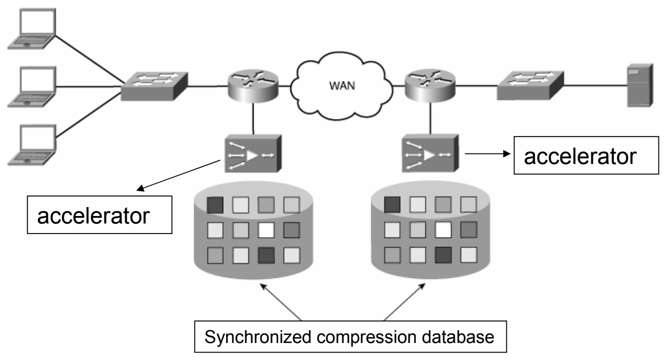


Figure 1: Data suppression and synchronized compression history [11]

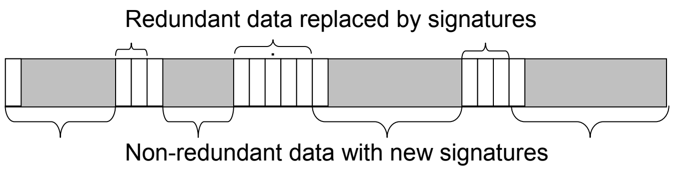

Redundant data replaced by signatures
Non-redundant data with new signatures

Figure 2: Encoded message with suppression technique [11]

When peering accelerators perform data suppression, only the signatures of the data segments which were stored in database are sent. Otherwise, the accelerators generate signatures for new data segments. As shown in Fig. 2, an encoded message consists of many small signatures. Besides, we can see some bigger pieces (with grey color) along with a small signature in the head. They are the new data segments and the new generated signatures which have not been stored in the database.

When an encoded message is received, the receiving accelerator begins to decode this message. This is done by replacing each signature sent without attached data with the corresponding data pattern in the local database. Any data pattern that has an accompanying signature is added to the local compression database and the signature is stripped from the encoded message. It is noteworthy that data suppression is commonly implemented in the network or transport layer, thus it does not differentiate among applications. For example, downloading an object from a website populates the local suppression database. The signatures related to this object can be used later for e-mail application when sending with the same (or modified) object. Furthermore, an object that is sent for the first time may be compressible because of some common data patterns from previous objects already stored in the data suppression database. When deploying the accelerators, a best practice is to ensure that enough storage capacity is allocated to provide at least one week's worth of compression history [11]. For instance, if a branch office is connected to the corporate network via a T1 line (1.544 Mbps), the necessary amount of disk capacity should be 1.5 GB [11]. Many of the data suppression implementations available in accelerators today do not use a simple single-level data pattern identification process (or per-packet suppression). However, most of them use a hierarchical means of identifying data patterns (or session-based suppression).

3.1.2.1 Per-packet suppression: Figure 3 describes the process of per-packet suppression that is divided by 7 steps:

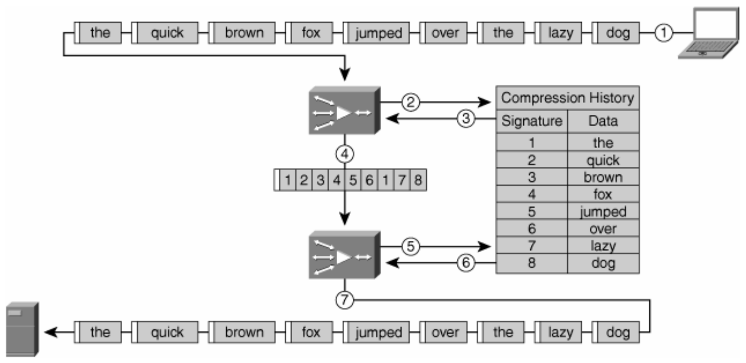


Figure 3: Encoding and decoding in per-packet suppression [11]

1. A stream of data is sent as a series of packets.
2. The accelerator verifies the existence of each packet in its local database.
3. Redundancy is eliminated and redundant segments are replaced by signatures.
4. The encoded packet is sent across the network and intercepted by remote accelerator.
5. The remote accelerator compares the contents of the encoded packet with items in its local database.
6. The remote accelerator replaces signatures with data patterns from its database.
7. The remote accelerator reconstruct the original data and forwards it to the intended destination.

The challenge of per-packet suppression is when the repeated patterns are not located in the same position after some changes have been applied to the original data. This causes a large number of new compression library entries to be created and thereby reducing compression benefits. The example in Fig. 4 shows the disadvantage of per-packet suppression when a small change is applied - a new character "a" is added to the head of the original string "the quick brown fox jumped over the lazy dog". In this case, the accelerator does not recognize the similarity between the original and the modified string, thus there is no compression and new entries would be added to the local database.

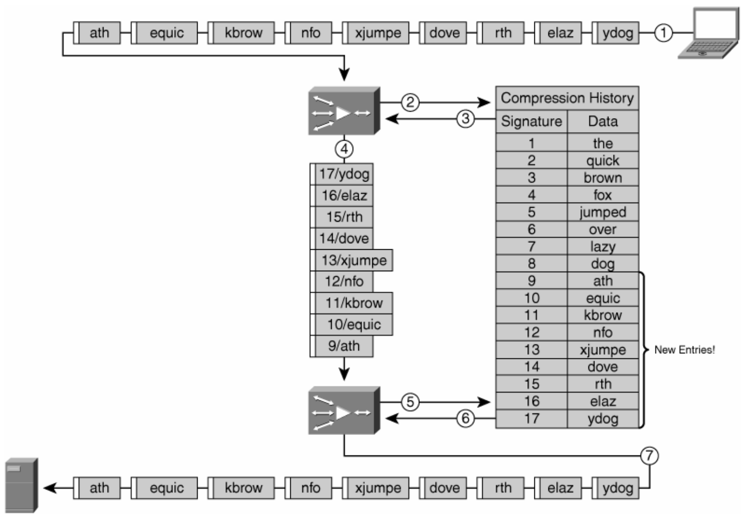


Figure 4: Per-packet suppression challenges with data locality [11]

3.1.2.2 Session-based suppression: Unlike per-packet suppression, the session-based suppression analyzes a block of data at multiple layers which is able to generate additional signatures that refer to a large number of data patterns and signatures (Fig. 5). In this way, if a greater degree of redundancy is identified within the data patterns, session-based suppression allows a smaller number of signatures to be used for the original data being transmitted. Session-based suppression generally provides better performance of compression even under significant amounts of data change. It has the ability to extend the compression domain for data suppression functionality by leveraging an intermediate data buffer.

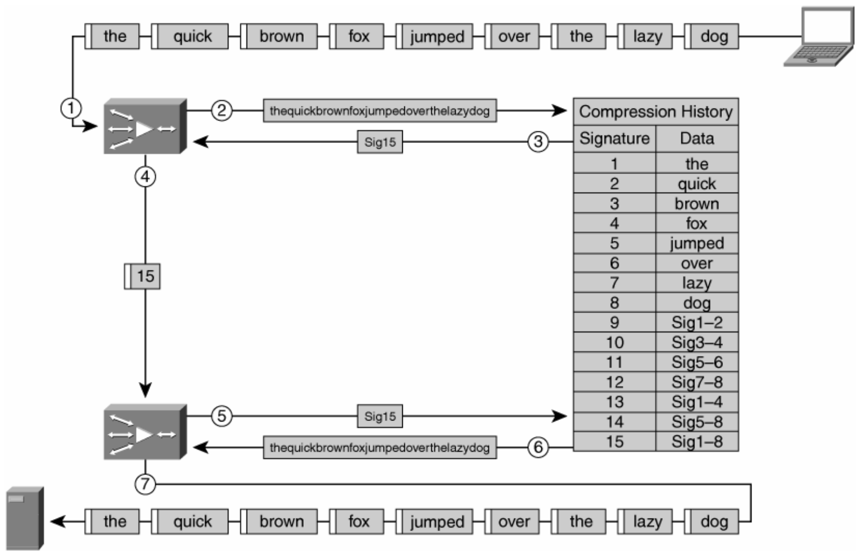


Figure 5: Compression with session-based suppression [11]

The example of Fig. 6 shows that the session-based suppression provides far better identification of the changed string than the per-packet suppression (Fig. 4), resulting in more efficient use of the compression library, bandwidth savings, and overall better performance.

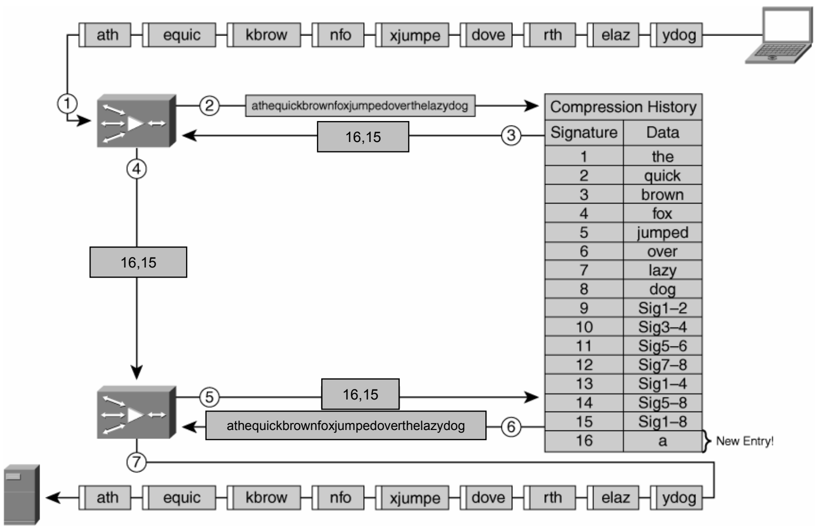


Figure 6: Session-based suppression with changed data [11]

3.1.3 Experiments of WOC in reducing traffic load

We present some results of using WOC from previous studies [11][14]. In addition, we have also set up a small test-bed to show the effectiveness of the WOC (Section 3.1.3.2).

3.1.3.1 Results from previous studies

1. *Results from Acme Data Corporation*
   Acme Data Corporation is a data processing company which has eight remote branch offices [11]. The accelerators are installed at the access links of the primary data center and eight remote offices. As we can see from Fig. 7, before deploying the accelerators, the average link utilization is 4Mbps. However, after using the accelerators, the throughput is significantly reduced to only 100Kbps. It is because the accelerators have eliminated all the redundant data on the network links (from 12:00 to 12:20, it is lunch time, thus there is not too much traffic).

2. *Results with Blue Coat System accelerator devices* [14]
   The experiments are done: (1) without Blue Coat, a WOC device from Blue Coat System [4], (2) a cold test - the file passing through the accelerator for the first time and (3) a warm test - the file is already in the cache. Fig. 8 measures the bandwidth gain that can be reached when using acceleration to open a 2.1 MB file. As we can see, the cold test (2) reduces 30% of traffic load in comparison with no acceleration. And this suppression ratio is over 99% for the warm test (3)

3. *Results for HTTP traffic*
   In this test, HTTP is used to download a high definition picture (6 MB) from a website. As the previous definition, the cold test means that the traffic is going through the accelerator for the first time and the warm test means that the traffic has already been going through the accelerator. As shown in Fig. 9, the warm test significantly reduces over 99% of the data traffic.

4. *Results for Messaging Application Programming Interface using accelerators (MAPI)*
   The test with MAPI was performed by sending an 8.3 MB email containing 3 photos. This email was

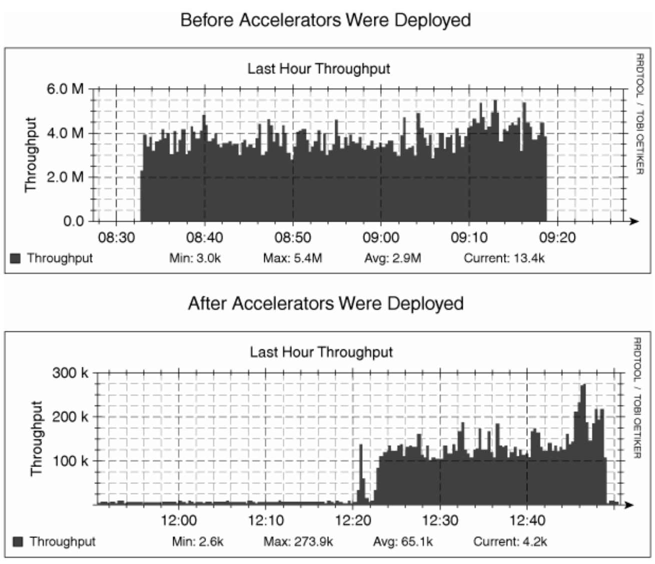


Figure 7: Acme Data WAN utilization before and after [11]

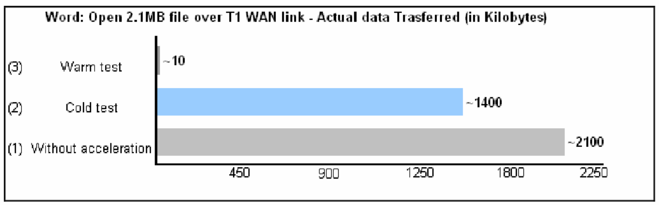


Figure 8: Results with Blue Coat System accelerator devices [14]

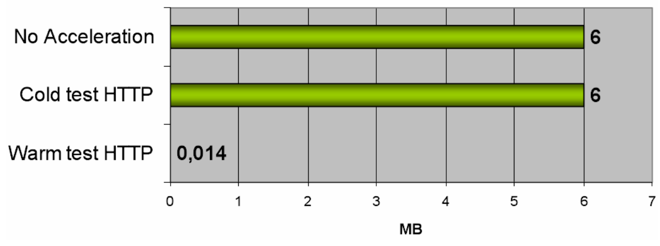


Figure 9: Download a high definition picture (6 MB) [14]

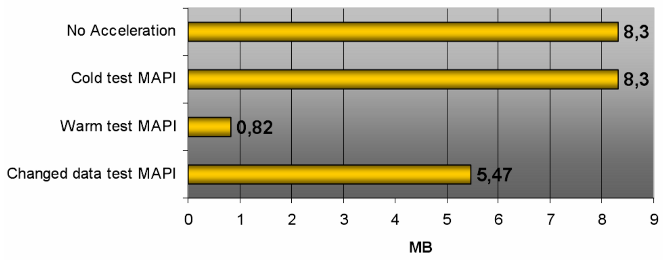


Figure 10: Results with MAPI using accelerators [14]

sent from the global data center to a user in the local data center. To test the performance of MAPI acceleration, a similar email was also sent through the network in which two photos had been changed. The results in Fig. 10 show the efficiency of using the accelerators in reducing traffic load. Over 90% of redundant data is eliminated by the warm test. Moreover, the accelerators still work well, reducing 35% of traffic load when some parts of the previous sent data (two photos in this test) have been changed.

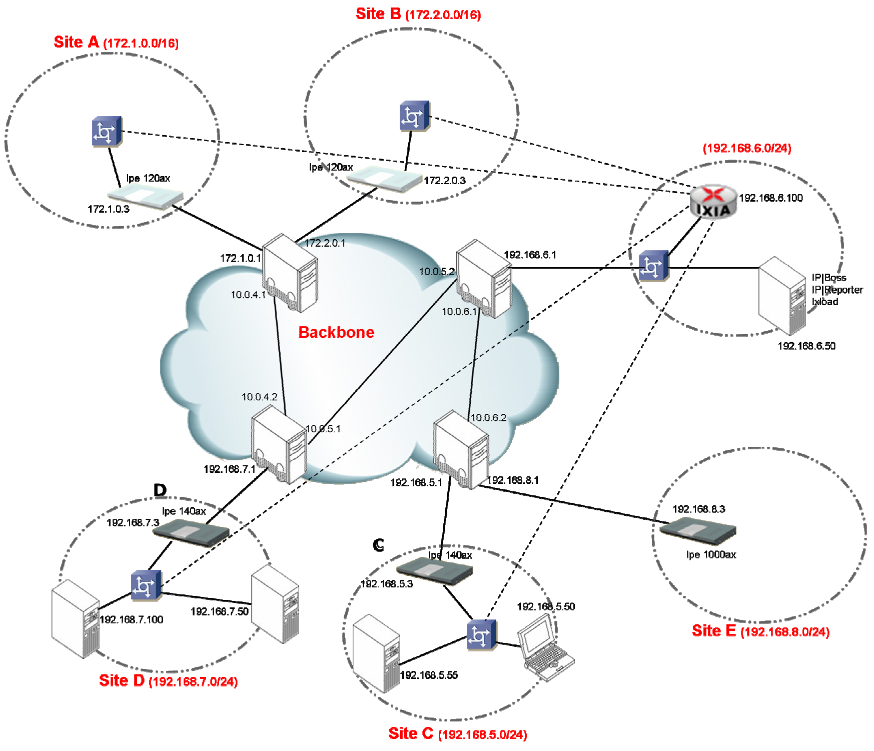


Figure 11: Testbed for the experiments

3.1.3.2 Results from our test-bed We have set up a small test-bed as shown in Fig. 11 where the Ipanema 140AX WAN Optimization Controller devices are used. These WOCs can perform both compression and redundancy elimination. The WOCs are also plugged in the Watt meters in order to perform power consumption measurements. We installed the WOCs at the access links of Site C and Site D. From a computer in Site C, we send files to other computer in Site D. Inside each site, the bandwidth capacity of links is very high, however, the bottleneck is the link from the WOC to outside.

As shown in Fig. 12, the transmission rate is far better when we enable the compression and suppression at WOCs. In this test, we send a file with high potential of compression and high redundancy ($\sim 98\%$). When the WOCs are disabled, the tranission rate is low ($\sim 1Mbps$) due to the limited bandwidth of the bottleneck link. However, when we enable the WOCs, almost traffic passing through the bottleneck link are the small signatures because the sent file consists much of redundancy and high compression ratio. The remote WOC then reconstructs the original data and forward to destination computer. This explains why the transmission rates are faster when we enable WOCs ($\sim 60Mbps$).

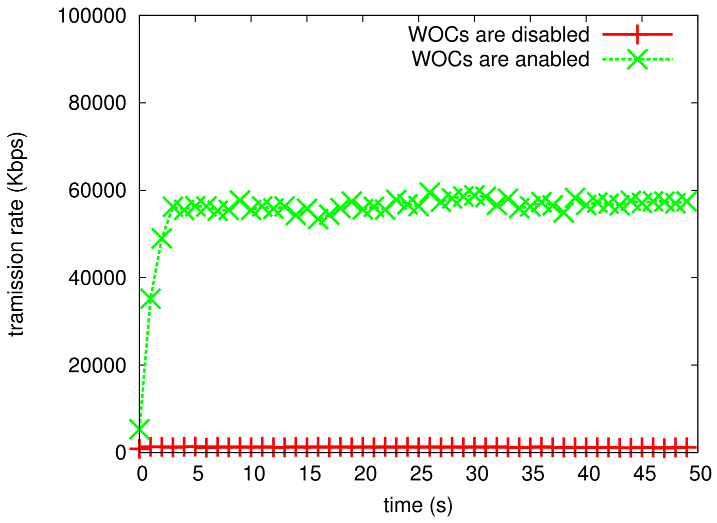


Figure 12: Transmission rate

In the next test, we create and send two files which are varied from the potential of redundancy elimination (RE) and compression (LZ or ZIP). The important thing we notice is that, in general when a WOC is enabled, it consumes more around 30 watt second (Fig. 13). Therefore, from the viewpoint of energy savings, we should consider the WOC in order to evaluate the global power consumption of the network. We will discuss more on this in Section 4.

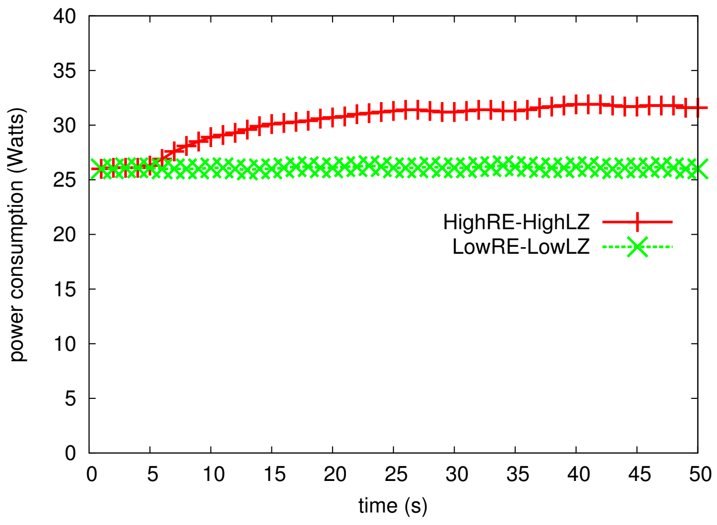


Figure 13: Power consumption of WOC

In summary, the real results (both from our experiments and other resources) show that the WOC is useful in reducing traffic load, and hence significantly improves the network performance. However, we notice that enabling WOC implies extra power consumption. It means that from the view point of energy savings, we should carefully consider when enabling a WOC. It should be also noteworthy that the WOCs are installed at the access links of small ISPs and enterprises. However, the problem of energy-aware routing is to find a routing solution for the core network. This means we need to consider the link capacity in the backbone, not only in the access network. This raises a new issue: whether traffic load in the backbone network is significantly reduced when the WOCs are integrated in routers across the Internet? The next

Section investigates in more details the reduction of traffic load in the core network.

3.2 Packet caches on routers

As shown in Section 3.1, WAN Optimization is a useful technique to improve network performance in small ISPs and enterprises. The success of WAN deployment has motivated researchers to explore the benefits of deploying packet-level RE in routers across the entire Internet [25][2][24][12]. The basic idea of this proposal is that: each router in an ISP network maintains a cache of recently forwarded packets. Upstream routers on a link use this cache to identify common content with new incoming traffic and replace these duplicate bytes by small signatures on the fly. Downstream routers look up the signatures in their local caches and reconstruct the original packets. The core techniques used here are similar to those in the WAN Optimization (Section 3.1). In more detail, for every incoming packet, routers run an algorithm to compute a set of fingerprints. Rather than using MD5 hash, the algorithm uses a sliding hash function which significantly cuts down the hash computation time per packet [26]. After computing fingerprints for an arriving packet, each fingerprint is checked against the fingerprint store. If a match is found, it means that the incoming packet has bytes in common with an in-cache packet. The algorithm will try to expand this packet to find a bigger matching region. Each matching region is then removed from the incoming packet and replaced by a small signature. This signature will be used by downstream routers to reconstruct the original packet from their local cache. It is important to make sure that the cache on downstream router is consistent with the upstream one.

Obviously, there are two key challenges that hinder the deployment of RE on routers. First, a significant number of memory accesses and heavy computation are required during various stages of RE. Second, a large amount of memory is required for fingerprints and packets stored at routers. Anand et al. [24] introduces SmartRE which considers these challenges in the design. The authors show that on the desktop 2.4 GHz CPU with 1 GB RAM used for storing caches, the prototype can work at 2.2 Gbps for encoding packets (finding fingerprints and replacing matching regions by signatures) and at 10 Gbps for decoding or reconstructing the original packets. Moreover, they believe that a higher throughput can be attained if the prototype is implemented in hardware. Therefore, the key challenges of limitation in memory and CPU can be overcome.

Another interesting fact is the benefit of deploying packet-level RE in routers or how much load on network links can be reduced when deploying packet caches on routers. Several real traffic traces have been collected from many networks such as at 11 corporate enterprises in US [2], at a large university in US [25] and at 5 sites of a large corporate network in North America [12]. The authors in [2][25][12] conclude that the bandwidth savings of RE and compression can be up to 50%. In addition, a further 10-25% traffic load can be reduced when considering redundancy-aware routing - traffic flows from the same source are aggregated on same links to achieve inter-flows RE [25].

In summary, based on the results of WAN Optimization and the packet-level RE on routers, we can see that the volume of traffic can be significantly reduced. This means that if the WOCs are implemented in routers, there is a high potential to aggregate traffic flows. This is useful for traffic engineering and energy-aware routing. In the next section, we explore the benefits of deploying WOC in routers from the view point of energy-aware routing.

4 Energy-aware routing with WOCs

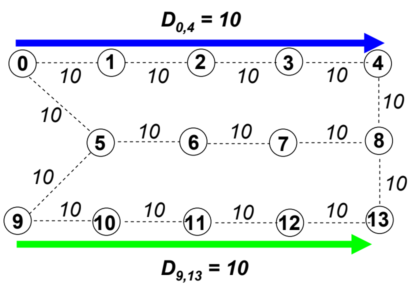
Recently, there are several studies on energy-aware routing that consider link capacity as the main constraint [8][22]. In our work, we use the WOCs to efficiently use the link capacity, and hence provide better solutions to save more energy. We divide the problem into 3 separate cases and propose the corresponding algorithms for finding a routing solution for each case. We introduce in Table 1 a summary of notations used in algorithms.

G = (V,E)	Network topology with V - the set of vertices (modeling routers) and E - the set of edges (modeling links).
D	A set of all demands.
D_{st}	Volume of the traffic demand from a source $s \in V$ to a destination $t \in V$.
C_{uv}	Capacity of an edge $(u, v) \in E$.
λ	Capacity/Demand ratio where $\lambda = C_{st}/D_{st}$.
f_{uv}^{st}	Flow on edge (u, v) corresponding to the demand D_{st}.
fc_{uv}^{st}	Compressed flow on edge (u, v) corresponding to the demand D_{st}.
x_{uv}	Binary variable which says if edge (u, v) is used or not.
γ	compression factor.
$N(u)$	All the neighbors of a vertex u in network topology G

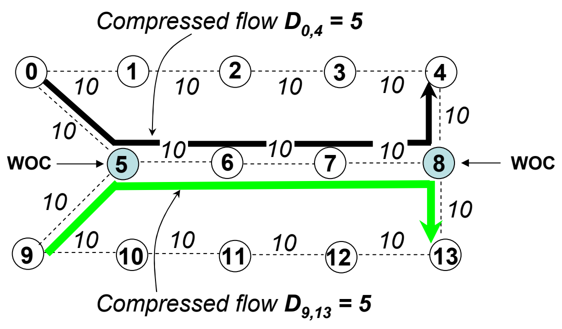
Table 1: Summary of notations

4.1 Case 1 - All routers on network are WOC-routers

The hypothesis is that every router in the network has a built-in WOC which can be enabled or disabled on demand. The energy-aware routing tries to route the traffic demands in a way that minimizes the number of active links.



(a) Without WOC (8 active links)



(b) With WOCs enabled at 5, 8 (7 active links)

Figure 14: Routing strategy with and without WOC

We present in Fig. 14 a simple example to show the efficiency of using WOCs in energy-aware routing. The numbers on the links are the link capacity. There are two demands $D_{0,4}$ and $D_{9,13}$, both with traffic volume of 10. It means that we need to find a routing solution for the two flows (0 to 4) and (9 to 13) which satisfies the link capacity constraint. As shown in Fig. 14a, the unique routing solution represented by the two line arrows passing through 8 links in total. It means that the best solution without WOC requires 8 active links. Meanwhile, we show in Fig. 14b: if the WOC at routers 5 and 8 are enabled, and they can compress 50% of the traffic, then we have a better solution with only 7 active links. In this work, we first model the energy-aware routing with WOC as an integral multicommodity flow problem using Integer Linear Program. Energy-aware routing problems are known to be NP-hard [19][18], and therefore finding the optimal solution becomes impractical even for small networks. Therefore, we present greedy heuristic algorithm that can be used to find solution for large networks.

4.1.1 Optimal solution - Integer Linear Program

As we have seen during the experiment (in Section 3.1.3.1) a WOC that performs compression increases its power consumption (around 30 Watts second). It represents approximately 30% of the consumption of a router interface. Therefore, we must consider the number of enabled-WOCs along with the number of active links in order to evaluate the global power consumption of the network. The objective of the ILP is to find a routing solution that minimizes the total power consumption of the network

We define a compression factor γ to represent the amount of data that can be compressed by WOCs.

For instance: with $\gamma = 2$, a traffic demand of 2 Mbps will be reduced to $2/\gamma = 1$ Mbps or it is equivalent that 50% of the traffic can be compressed. We differentiate the usual flow f_{uv}^{st} from the compressed flow fc_{uv}^{st}, on link (u, v) for a demand (s, t). If WOC is enabled at node u and it performs a whole compression of the demand (s, t) (which has not been compressed yet), then the sum of the usual flows entering u is equal to γ times the sum of the compressed flows leaving u. Hence, we can get the following equation where $N(u)$ is a set of neighbor nodes of u on the network:

$$\sum_{v \in N(u)} f_{vu}^{st} = \gamma \sum_{v \in N(u)} fc_{uv}^{st}$$

For each link $(u, v) \in E$, we introduce a binary variable x_{uv} which determines if the link is used or not: $x_{uv} = 0$ if $\sum_{(s,t)\in D} \left(f_{uv}^{st} + f_{vu}^{st}\right) = 0$ and $x_{uv} = 1$ if $\sum_{(s,t)\in D} \left(f_{uv}^{st} + f_{vu}^{st}\right) > 0$. We consider a simplified architecture where each line card on router has one network interface. Based on ISP estimations, a power consumption of a single line card is equal to 100Ws [18]. Therefore, when a link (u, v) is used ($x_{uv} = 1$), the two network interfaces (or the two line cards) u and v are enabled, and they consume around 200Ws. We also define a binary variable $nodes_u$ which is equal to 1 if the WOC on node u is enabled (it performs compression or decompression), and is equal to 0 otherwise.

The *Objective Function* is then to minimize the total power consumption of the network:

$$\min \left[200 \sum_{(uv)\in E} x_{uv} + 30 \sum_{v \in V} nodes_u \right] \tag{1}$$

subject to:

"*Flow conservation constraint*": $\forall (s, t) \in D, \forall u \in V$,

$$\sum_{v\in N(u)} f_{vu}^{st} - \sum_{v\in N(u)} f_{uv}^{st} + \gamma \Big(\sum_{v\in N(u)} fc_{vu}^{st} - \sum_{v\in N(u)} fc_{uv}^{st} \Big) = \begin{cases} -D_{st} & \text{if } u = s, \\ D_{st} & \text{if } u = t, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

"*Capacity constraint*": $\forall (u, v) \in E$, C_{uv}: capacity of link (u, v)

$$\sum_{(s,t)\in D} \left(f_{uv}^{st} + f_{vu}^{st} + fc_{uv}^{st} + fc_{vu}^{st} \right) \le x_{uv} C_{uv} \tag{3}$$

"*Compressed node*": $\forall u \in V, (s, t) \in D$,

$$M.nodes_u \ge \sum_{v\in N(u)} \left(fc_{uv}^{st} - fc_{vu}^{st} \right) \tag{4a}$$

$$M.nodes_u \ge \sum_{v\in N(u)} \left(fc_{vu}^{st} - fc_{uv}^{st} \right) \tag{4b}$$

Equation (1) minimizes the total power consumption of the network. Equation (2) states the flow conservation constraint where traffic flows are routed using a fluid model and we differentiate between the compressed and the normal flows. Note that, in this model, we cannot compress a flow that has already been compressed. Constraint (3) forces the link load to be smaller than the link capacity. Like some existing works [19][15][18], we consider a simplified model where link load is the total volume of all the flows passing through this link. In addition, to accommodate traffic bursts, we should limit the maximum utilization over any links in the network. For instance, the authors in [15][18] suggest to set the maximum link utilization $\alpha = 50\%$. Therefore, the capacity using in equation (3) would be set to αC_{uv}. The constraints (4a) and (4b) (where M is a big constant number) are to make sure when the WOC in node u is disabled ($nodes_u = 0$), it cannot perform compression or decompression. This means the difference between the sum of the compressed flows that leave and enter node u is equal to zero.

4.1.2 Heuristic solution

In this section, we present our heuristic algorithm to find a routing solution. At the beginning, we assume that all routers enable WOC. Therefore, we can virtually decrease the volume of all the traffic demands to $D^{st}_{comp} = D^{st}_{real}/\gamma$. The heuristic algorithm then tries to find a routing solution for all compressed demands so that it minimizes the number of active links. After that, based on the routing found in the previous step, we develop another algorithm to find unused WOCs and disable them to save energy. For more detail, the algorithm goes through two steps:

- Step 1 - find a routing solution for all the compressed demands in which we try to minimize the number of active links.
- Step 2 - using the routing found on step 1, find nodes to enable the WOC so that the number of enabled-WOCs is minimized and the capacity constraint is satisfied.

Algorithm 1: *Step 1* - removing less loaded links
Input: An undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has a capacity C_e and a residual capacity R_e (depending on the demands passing through the edge e)

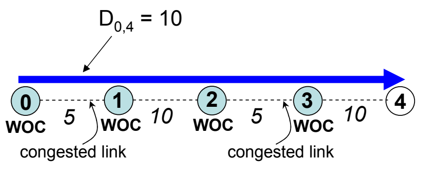
```
$\forall e \in E, R_e = C_e$
Compute a feasible routing for all the compressed demands with the Algorithm 2.
while edges can be removed do
    remove the edge $e$ that has not been chosen and has smallest value $C_e/R_e$
    compute a feasible routing with Algorithm 2
    if no feasible routing exists, put back $e$ in G
end
return the feasible routing if it exists.
```

Algorithm 2: *Step 1* - finding a feasible routing
Input: An undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has a capacity C_e, a residual capacity R_e, an initial weight w and a set of demands D, each demand has a volume of traffic D_{st}.

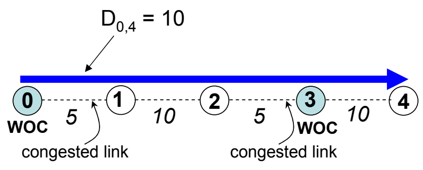
```
Set up weight for all links $w_e$ = the number of demands
Sort the demands in random order
while $D_{st}$ is a demand in $D$ with no routing assigned do
    compute the shortest path $SP_{st}$ with the metric weight ($w_e$)
    assign the routing $SP_{st}$ to the demand $D_{st}$
    $\forall e \in SP_{st}, R_e = C_e - D_{st}$
    $\forall e \in SP_{st}, w_e = w_e - 1$
end
return the routing (if it exists) assigned to the demands in $D$
```

The algorithm in step 1 is similar to the LESS LOADED EDGE HEURISTIC in [19]. We start from the whole network, compute the feasible routing for the compressed demands (Algorithm 2 - Step 1) and try to remove in priority links that are less loaded (Algorithm 1 - Step 1). Initially, all links on the network are set up with the same weight w which is equal to the total number of demands. To find a routing solution, the demands are considered one by one in random order. We compute the shortest path for each demand with the metric w on links. After finding a shortest path for a demand, the weight of links that have carried the shortest path is updated: $w = w - 1$. Using this weight metric in the shortest path, we implicitly set high priority on links that have already been selected to be reused and hence, reduce the number of active links. We also consider the metric C_e/R_e on links where R_e is the residual capacity on link e when the previous demands have been routed. This metric represents the load on link and is used to select link to remove in "Algorithm 1 - Step 1".

In step 2 - finding a solution to enable WOCs, we use the routing found in step 1 as the input of the algorithm. Now, we consider the feasible routing with the real traffic demands, that is to say the traffic demands that are not compressed. Therefore, some links can be congested because the total traffic volume of demands may exceed the link capacity. We then find flows that travel through the congested links and find the starting and ending points of congested links to enable the WOCs. The example of Fig. 15 shows the congested links and the solutions to enable WOCs. The flow of demand (0,4) has traffic volume of 10. Note that, the number on links in the Fig. 15 indicates the value of link capacity. Therefore the two links (0, 1) and (2, 3) are congested. When a flow traveling many congested links, a naive solution is to enable WOCs at the two end-points of each congested link as shown in Fig. 15a. However, a better solution should be to enable WOCs only at the starting and ending points of all the congested links that the flow passing through. In this example, we only need the WOCs at router 0 and router 3 to be enabled (Fig. 15b).



(a) Naive solution of enabling WOCs



(b) Better solution of enabling WOCs

Figure 15: Solution of enabling WOCs

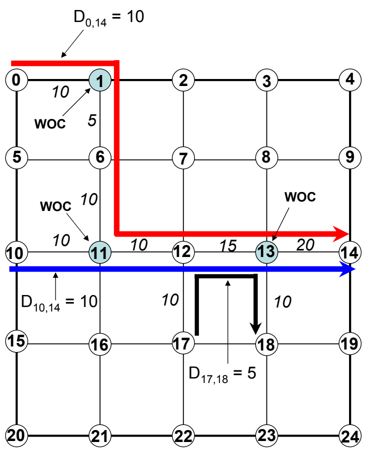


Figure 16: The congested links and the enabled WOCs

In Fig. 16, links (1, 6), (11, 12) and (12, 13) are congested because $D^{0,14}_{1,6} > C_{1,6}$, $D^{0,14}_{11,12} + D^{10,14}_{11,12} > C_{11,12}$ and $D^{0,14}_{12,13} + D^{10,14}_{12,13} + D^{17,18}_{12,13} > C_{12,13}$. For the flow $f_{0,14}$, WOCs at router 1 and router 13 need to be enabled to compress data and reduce the amount of traffic passing through the congested links. Similarly, the WOCs at (11, 13) and at (12, 13) should be enabled for the flows $f_{10,14}$ and $f_{17,18}$, respectively. Note that, for our assumption, a WOC can selectively compress the flows passing it. For example, the WOC at 11 only compresses the flow $f_{10,14}$, it is not necessary to compress the flow $f_{0,14}$ because it has been compressed before. We can see that when compressing the flow $f_{0,14}$, the amount of traffic passing through all the congested links (links (1, 6), (11, 12) and (12, 13)) is also reduced. Therefore, we call $f_{0,14}$ be "must-be-compressed-first" flow because it is possible to release all the congested links when compressing this flow. We decide to compress this flow first and check the network again to find if there are congested links or not. Assume that the compression ratio $\gamma = 2$. Hence, in this example, the links (11, 12) and (12, 13) are still congested. Now, we remove the compressed flow $fc_{0,14}$ and its bandwidth occupation, then the available capacity of links (11, 12) and (12, 13) are 5 and 10, respectively. Now, the flow $f_{10,14}$ becomes must-be-compressed-first flow. We again try to compress the flow $f_{10,14}$, and finally there is no congested link on the network (link (12,13) is also released from congestion). Therefore, in the final solution, the WOCs at router 1, 11 and 13 will be enabled.

In summary, the algorithm tries to find the solution of enabling WOCs for all the must-be-compressed-first flows. If the network is still congested, we enable WOCs for the remaining flows until all congested links are free. We believe the strategy of this algorithm can reduce the number of enabled WOCs on the network.

Algorithm 3: *Step 2* - finding a solution to turn on WOCs so that the number of enabled WOCs is minimized
Input: An undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has capacity C_e. A feasible routing solution in Step 1. A set of demands D, each demand has a volume of traffic D_{st}

```
while network is congested and there are must-be-compressed-first flows do
    Find enabled WOCs for must-be-compressed-first flows.
    Remove the compressed flows.
end
while network is still congested do
    find enabled WOCs for the remaining flows.
end
return all enabled WOCs on the network.
```

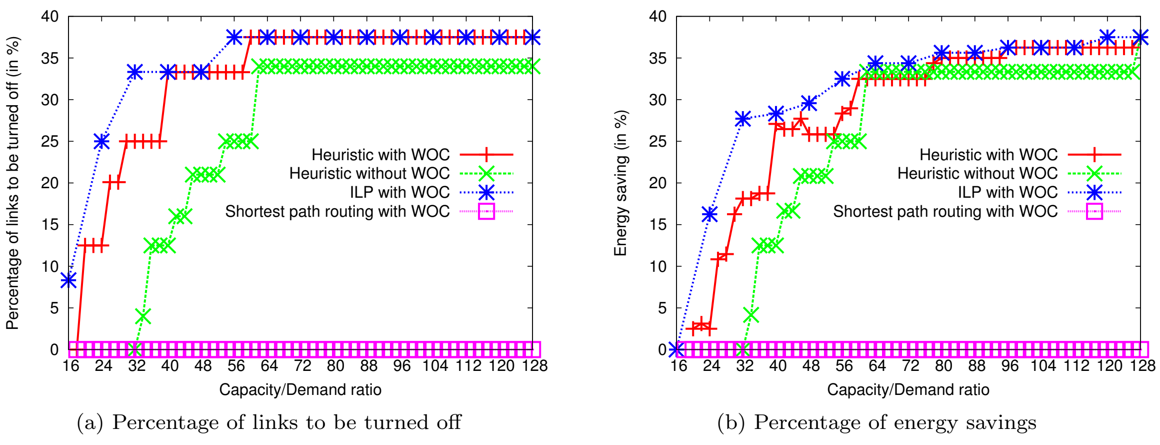
4.1.3 Simulation results

We implemented this model using the Sage, an open source mathematics software [23]. Sage translates the model into a form that can be recognized by CPLEX [7] for solving optimization problems. Because CPLEX takes a long time to find the optimal solution even for a small network, we force the program to stop after a run time of one hour. We collect simulation results for the square grid and some real backbone network topologies. Because of the high potential in compressing traffic shown in section 3, we use the compression factor $\gamma = 2$ for all the simulations.

4.1.3.1 Results on 4x4 and 10x10 grid We present results for the 4 x 4 grid (heuristic without WOC [19], ILP with WOC, heuristic with WOC and shortest path routing with WOC) and for a 10 x 10 grid (heuristic without WOC [19], heuristic with WOC and shortest path routing with WOC). For simplicity, all links are set up with the same capacity C and the demands are all-to-all (one node has a set of demands to send traffic to all remaining nodes in the network) with the same traffic volume D.

In Fig. 17a, we present the percentage of links over the whole network that can be turned off for a grid 4 x 4. The x axis represents the "capacity/demand ratio" (C/D). As we can see, more network interfaces can be turned off with the support of WOC. Another interesting observation is that when the "capacity/demand ratio" of each link is less than 36, there is no routing solution without WOC. A big ratio of capacity/demand is needed for finding the solution because we use all-to-all routing where there are total 240 demands for a grid 4x4. However, with WOC, it starts to turn off unused links to save energy when the "capacity/demand ratio" is equal to 18. The explanation is that when traffic demands are compressed using WOCs, there is high possibility to aggregate these demands into a small set of links, and hence we can find feasible routing and more network interfaces can be turned off.

In Fig. 17b, we explore how much energy can be saved when taking into account the power consumption of both network interfaces and WOCs. The power consumption of a link is equal to 200 watts. We compute the energy consumption of the network when all the links are active: [(number of links on network)*200]. As shown in Section 1.1, the power consumption of network with WOC is calculated as: [(number of active



(a) Percentage of links to be turned off (b) Percentage of energy savings

Figure 17: Simulation results for a 4x4 grid

links)*200+ (number of WOCs)*30]. Then, the difference between these two values represents how much energy can be saved. We can see the gap between the heuristic with WOC and heuristic without WOC is smaller than in Fig. 17a. It is because for the heuristic with WOC, the enabled WOCs also contribute in the total energy consumption of the network. In general, the heuristic with WOC can work well and approximate to the results of ILP.

We present in Fig. 18 the number of WOCs is used by the heuristic and the ILP. The increase of capacity/demand ratio means that the links have more capacity to aggregate traffic. Therefore, it seems that it is not not necessary to enable WOCs when the links have enough capacity for traffic aggregation. As shown in Fig. 18, in general, the number of enabled WOC is reduced when the capacity/demand ratio increases. However, there are some points, for example at the capacity/demand 48 for the heuristic and at 56 for the ILP, the number of enabled WOC increases. It is because at these points, enabling more WOCs allows to turn off more links and hence, reducing the global power consumption of the network.

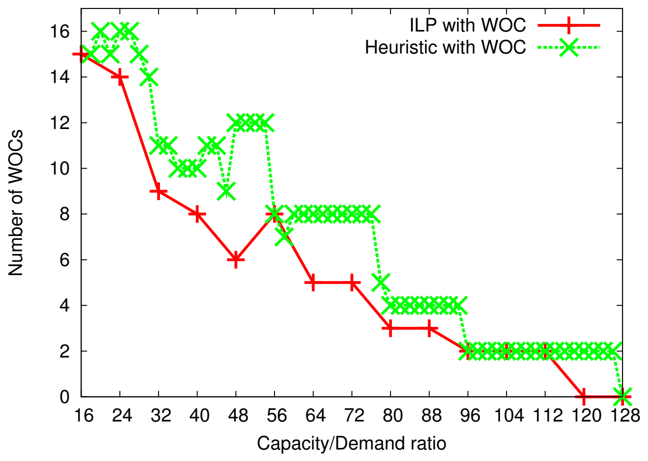
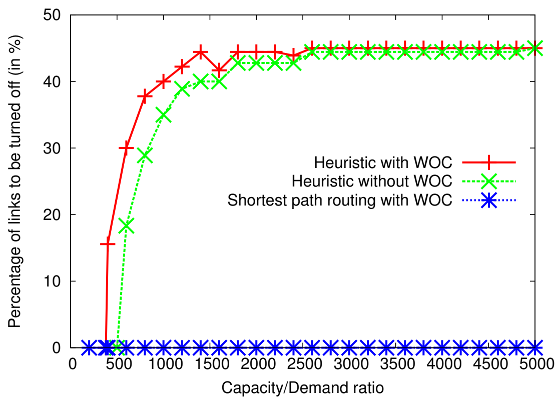


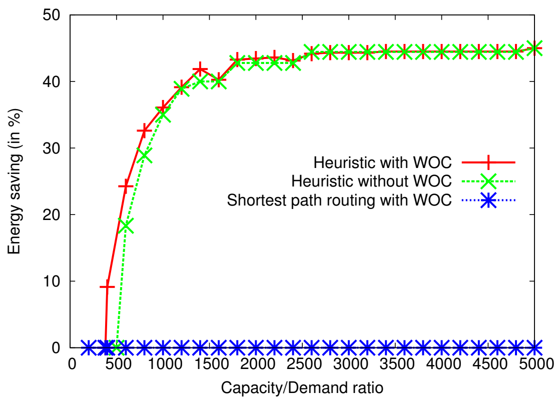
Figure 18: Number of enabled WOCs

For the grid 10 x 10 (Fig. 19), we see that the gap between the heuristic with WOC and heuristic without WOC is small. However, take a closer look, we can see that the heuristic with WOC is more efficient when the capacity/demand ratio is small. For instance, it starts to find solution that can turn off 15% of links when the capacity/demand ratio is 385 while there is no solution for the heuristic without WOC. Moreover, the heuristic without WOC starts to find routing solution (but no links can be turned off) at the capacity/demand ratio of 500, meanwhile the heuristic with WOC can save up to 30% of the links. In general, we have the same conclusion with grid 4 x 4: more network interfaces can be turned off with the support of WOC, and hence more energy can be saved.

In addition, we consider shortest path routing with WOCs where there is no aggregation on flows. This is similar to the case of traditional shortest path routing used in [25]. The results of both grid 4x4 and 10x10 show that: without considering energy-aware routing, there is no energy saving even if we enable all the WOCs in routers.



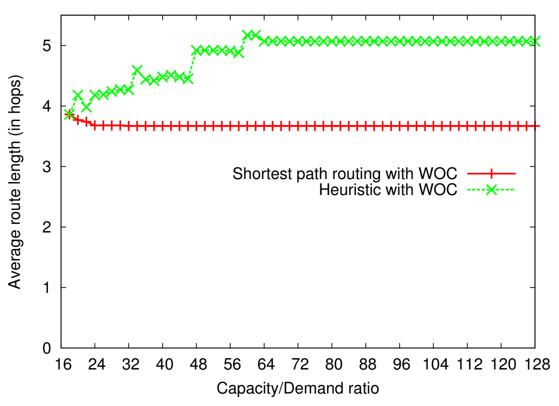
(a) Percentage of links to be turned off



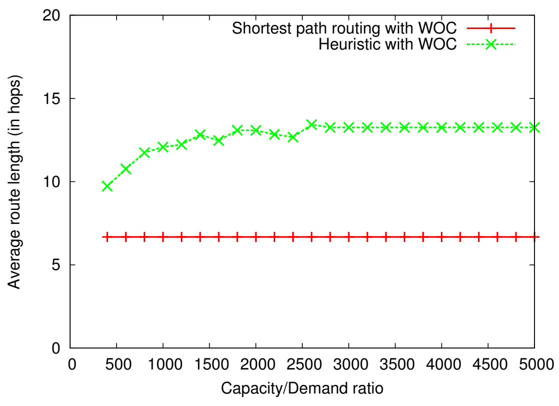
(b) Percentage of energy savings

Figure 19: Simulation results for a 10x10 grid

We also compare the average route length for the heuristic algorithm and the shortest path routing in Fig. 20. As we can see, when turning off links on the network, we save energy but the traffic demands need to route on longer paths. When the capacity/demand increases, more traffic demands can be aggregated to a small number of links, resulting in longer routing paths. However, when this ratio is large (for example 64 for the grid 4x4 and 2600 for the grid 10x10), the routing solutions do not change much, and hence the average route length seems to be unchanged.



(a) Average route length (grid 4x4)



(b) Average route length (grid 10x10)

Figure 20: Average route length on grid topology

4.1.3.2 Results on General Networks We study ten classical network topologies extracted from SNDLib (http://sndlib.zib.de). In our experiments, we explore how many network interfaces can be turned off and how much energy can be saved for different ranges of over-provisioning factor. To compare with the solution without WOC, we consider a range of capacity/demand ratio λ_{noW} starting from the smallest value that allows to route all the demands without compression (over-provisioning factor equals to λ_{noW}) [19].

Network topologies	$\mid V \mid$	$\mid E \mid$	λ_{noW}	λ_W	Over-provisioning factor					
					with WOC			without WOC		
					λ_{noW}	$2\lambda_{noW}$	$3\lambda_{noW}$	λ_{noW}	$2\lambda_{noW}$	$3\lambda_{noW}$
Atlanta	15	22	38	22	31.8%	36.4%	36.4%	0%	32%	36%
New York	16	49	15	8	57.1%	65.3%	67.3%	2%	59%	63%
Nobel Germany	17	26	44	23	30.8%	38.5%	38.5%	0%	35%	39%
France	25	45	67	36	37.8%	46.7%	46.7%	0%	42%	44%
Norway	27	51	75	41	41.2%	47.1%	49%	12%	43%	47%
Nobel EU	28	41	131	66	31.7%	34.2%	34.2%	12%	32%	34%
Cost266	37	57	175	88	29.8%	36.8%	36.8%	3.5%	32%	35%
Giul39	39	86	85	43	40.7%	51.2%	53.5%	0%	45%	50%
Pioro40	40	89	153	77	48.3%	55.1%	56.2%	0%	53%	54%
Zib54	54	80	294	168	27.5%	32.5%	33.8%	0%	30%	33%

Table 2: Gain of network interfaces (in %)

Network topologies	$\mid V \mid$	$\mid E \mid$	λ_{noW}	Over-provisioning factor					
				with WOC			without WOC		
				λ_{noW}	$2\lambda_{noW}$	$3\lambda_{noW}$	λ_{noW}	$2\lambda_{noW}$	$3\lambda_{noW}$
Atlanta	15	22	38	23.6%	30.1%	36.4%	0%	32%	36%
New York	16	49	15	52.2%	61.6%	65.2%	2%	59%	63%
Nobel Germany	17	26	44	23.9%	33.9%	36.8%	0%	35%	39%
France	25	45	67	33.8%	44%	45.4%	0%	42%	44%
Norway	27	51	75	36.2%	45%	47.5%	12%	43%	47%
Nobel EU	28	41	131	27.7%	30.9%	34.2%	12%	32%	34%
Cost266	37	57	175	25.3%	33.6%	35%	3.5%	32%	35%
Giul39	39	86	85	36.5%	48.4%	51.1%	0%	45%	50%
Pioro40	40	89	153	45.3%	53.2%	54.5%	0%	53%	54%
Zib54	54	80	294	23.2%	31.2%	32.7%	0%	30%	33%

Table 3: Gain of power consumption (in %)

From Table 2 and Table 3, we can see that with WOC, we can save energy with smaller capacity/demand ratio ($\lambda_W \simeq \lambda_{noW}/2$ because we choose the compression ratio that is equal to 2). When λ is large, both the solutions (with and without WOC) converge to almost the same value of gaining of network interfaces and power consumption. For example, at $3\lambda_{noW}$, both solutions with and without WOCs achieve the same value in energy savings and spared network interfaces.

4.2 Case 2 - WOCs have already been placed on network

In this scenario, a set of WOC-routers has already been placed on the network. The purpose of our algorithm is to find a routing solution that minimizes the number of active links.

4.2.1 Optimal solution - Integer Linear Program

Similar to the Case 1, the *Objective Function* of Case 2 is also to minimize the total power consumption considering the number of active links and the number of enabled WOC-routers. The Integer Linear Program of the Case 1 can be used where we add a constraint that only the enabled WOC-routers can compress or decompress data.

Objective function:

$$min\Big[\ 200\sum_{(uv)\in E} x_{uv} + 30\sum_{v\in V} nodes_u\Big]$$

subject to:

"*Flow conservation constraint*": $\forall(s,t)\in E, \forall u\in V$,
if (u is a WOC):

$$\sum_{v\in N(u)} f^{st}_{vu} - \sum_{v\in N(u)} f^{st}_{uv} + \gamma\Big(\sum_{v\in N(u)} fc^{st}_{vu} - \sum_{v\in N(u)} fc^{st}_{uv}\Big) = \begin{cases} -D_{st} & \text{if } u = s,\\ D_{st} & \text{if } u = t,\\ 0 & \text{otherwise.}\end{cases}$$

else:

$$\sum_{v\in N(u)} f^{st}_{vu} - \sum_{v\in N(u)} f^{st}_{uv} = \begin{cases} -D_{st} & \text{if } u = s,\\ D_{st} & \text{if } u = t,\\ 0 & \text{otherwise.}\end{cases}$$

AND

$$\sum_{v\in N(u)} fc^{st}_{vu} - \sum_{v\in N(u)} fc^{st}_{uv} = 0 \qquad if\ u \neq (s,t)$$

"*Capacity constraint*": $\forall(u,v)\in E$, C_{uv}: capacity of link (u,v)

$$\sum_{(s,t)\in D}\Big(f^{st}_{uv} + f^{st}_{vu} + fc^{st}_{uv} + fc^{st}_{vu}\Big) \leq x_{uv}C_{uv}$$

"*Compressed node*": $\forall u\in V, (s,t)\in D$,

$$M.nodes_u \geq \sum_{v\in N(u)}\Big(fc^{st}_{uv} - fc^{st}_{vu}\Big)$$

$$M.nodes_u \geq \sum_{v\in N(u)}\Big(fc^{st}_{vu} - fc^{st}_{uv}\Big)$$

4.2.2 Heuristic algorithm

Algorithm 4: removing less loaded links
Input: an undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has a capacity C_e and a residual capacity R_e (depending on the demands supported on e). A set of demands D, each demand has a volume of traffic D_{st}.

```
$\forall e \in E, R_e = C_e$
Compute a feasible routing of the demands with the Algorithm 5.
**while** *edges can be removed* **do**
    remove the edge $e'$ that has not been chosen once, with the smallest value $C_{e'}/R_{e'}$
    compute a feasible routing with Algorithm 5
    if no feasible routing exists then put back $e'$ in G
**end**
return feasible routing if it exists.
```

Similar to Step 1 in the Case 1, we start from the whole network and demands are considered one by one in random order. We then compute the shortest path for each demand with the metric weight w (defined

Algorithm 5: finding the feasible routing
Input: An undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has a capacity C_e and a residual capacity R_e (depending on the demands supported on e). A set of demands D, each demand has a volume of traffic D_{st}.

```
Set up weight for all links $w_e$ = the number of demands
Sort the demands in random order
while $D_{st}$ is a demand in $D$ with no routing assigned do
    compute the shortest path $SP_{st}$ with the metric weight ($w_e$)
    while there is congestion do
        find compressible parts of the flow $SP^c_{st}$
        remove congested links
    end
    assign the routing $SP_{st}$ to the demand $D_{st}$
    $\forall e \in SP_{st}$ & $e \notin SP^c_{st}, R_e = C_e - D_{st}$
    $\forall e \in SP^c_{st}, R_e = C_e - (D_{st}/\gamma)$
    $\forall e \in SP_{st}, W_e = W_e - 1$
end
```

in Case 1) on links. After finding a shortest path for a demand, we find parts of flow that travel through the WOCs (called compressible parts) (Fig. 21). When updating the residual capacity on link, we consider the real capacity and for those compressible parts, we consider the value of traffic demand be D_{st}/γ. If this flow causes congestion on some links, we remove the congested links and try to find shortest path for this demand until there is no congestion. The algorithm is finished when we can route all the traffic demands without congestion otherwise, the algorithm return failure.

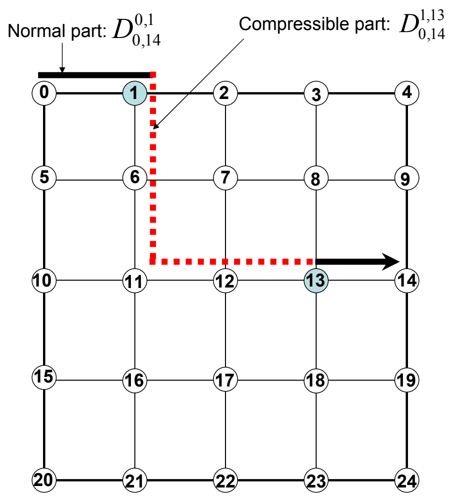
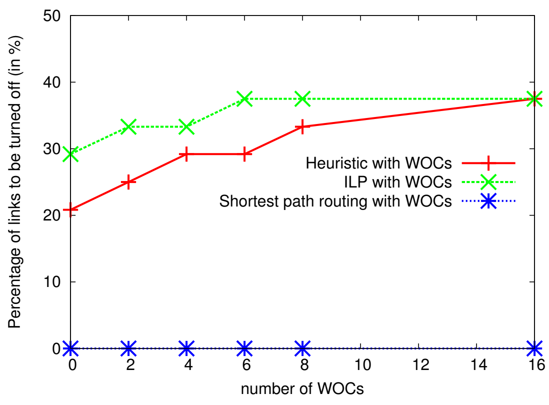

Normal part: $D^{0,1}_{0,14}$
Compressible part: $D^{1,13}_{0,14}$
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

Figure 21: Compressible part of a demand

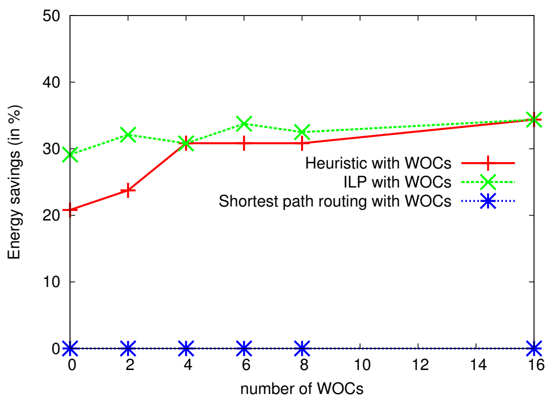
4.2.3 Simulation results

We present the results for grid 4 x 4 with capacity/demand ratio is 60. At first, we run the ILP in Case 3 (Section 4.3) to find the best placement corresponding to number of WOCs. Then, using these placements of WOCs, we compare the heuristic solution and the solution given by the ILP in case 2.

As shown in Fig. 22, our heuristic algorithm can work well and close to the solution given by the ILP. However, when there are 4 WOC-routers on the network, the ILP gives solution which consumes more energy than the case of 2 WOC-routers. The explanation is that because we force the ILP to stop when the run time exceeds one hour, thus the solution given by the ILP is not an optimal solution. And similar to case 1, the energy-aware routing takes longer route (Fig. 23) but there is no energy savings for the shortest path routing (Fig. 22).



(a) Percentage of links to be turned off



(b) Percentage of Energy savings

Figure 22: Simulation results for a 4 x 4 grid

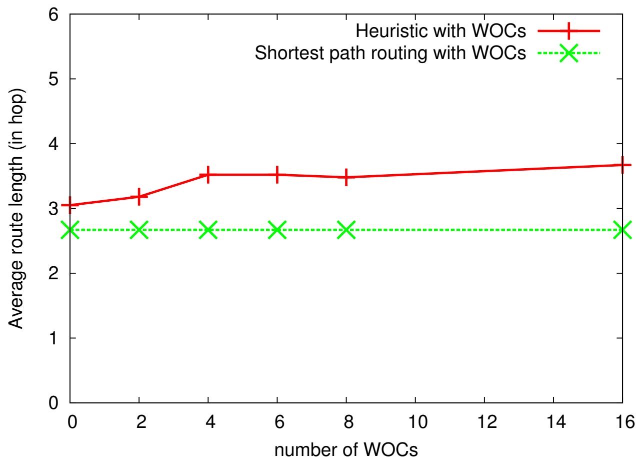


Figure 23: Average route length for a 4 x 4 grid

4.3 Case 3 - Limited number of WOCs

We define MAXWOC as the maximum number of WOC-routers that can be placed on the network. Our objective is to find a routing that satisfies link capacity constraint while respecting the global power consumption of the network.

4.3.1 Optimal solution - Integer Linear Program

Similar to the Case 1, the *Objective Function* of Case 3 is also to minimize the total power consumption of both the network interface and the WOC. The Integer Linear Program of the Case 1 can be used for Case 3 where we add one more constraint of the number of WOCs.

Objective function:

$$min\Big[\ 200 \sum_{(uv)\in E} x_{uv} + 30 \sum_{v\in V} nodes_u \Big]$$

subject to:

“*Flow conservation constraint*”: $\forall (s,t) \in E, \forall u \in V,$

$$\sum_{v\in N(u)} f^{st}_{vu} - \sum_{v\in N(u)} f^{st}_{uv} + \gamma\Big(\sum_{v\in N(u)} fc^{st}_{vu} - \sum_{v\in N(u)} fc^{st}_{uv} \Big) = \begin{cases} -D_{st} & \text{if } u = s, \\ D_{st} & \text{if } u = t, \\ 0 & \text{otherwise.} \end{cases}$$

“*Number of WOCs constraint*”:

$$\sum_{u\in V} node_u \le MAXWOC$$

“*Capacity constraint*”: $\forall (u,v) \in E$, C_{uv}: capacity of link (u,v)

$$\sum_{(s,t)\in D} \Big(f^{st}_{uv} + f^{st}_{vu} + fc^{st}_{uv} + fc^{st}_{vu} \Big) \le x_{uv} C_{uv}$$

“*Compressed node*”: $\forall u \in V, (s,t) \in D,$

$$M.nodes_u \ge \sum_{v\in N(u)} \Big(fc^{st}_{uv} - fc^{st}_{vu} \Big)$$

$$M.nodes_u \ge \sum_{v\in N(u)} \Big(fc^{st}_{vu} - fc^{st}_{uv} \Big)$$

4.3.2 Heuristic algorithm

The algorithm can work in a similar way with the Case 1. However, instead of returning only one feasible routing solution for the demands (which has the minimum number of active links), we keep all possible sets of feasible routing that are found in step 1 in increasing order of the number of active links. We iteratively select one feasible routing solution that has smallest number of active links and apply the placement algorithm (Algorithm 8). The loop is stopped when we find a placement that the number of WOCs is less than MAXWOC.

Algorithm 6: *Step 1* - removing less loaded links
Input: An undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has a capacity C_e and a residual capacity R_e (depending on the demands supported on e)

```
$\forall e \in E, R_e = C_e$
Compute a feasible routing of the demands with the Algorithm 7.
list-of-routes = null
while edges can be removed do
    remove the edge $e$ that has not been chosen once, with the smallest value $C_e/R_e$ compute a
    feasible routing with Algorithm 7
    append the feasible routing to list-of-routes
end
return list-of-routes
```

Suppose that the compression ratio is γ, we first virtually decrease all the traffic demands to $D^{st}_{comp} = D^{st}_{real}/\gamma$. The heuristic algorithm then goes through two steps:

- Step 1 - find a list of feasible routing for all the compressed demands.
- Step 2 - using the routing found on step 1, find nodes to enable the WOC so that the number of enabled-WOCs is minimized and the capacity constraint is satisfied.

Algorithm 7: *Step 1* - finding a feasible routing
Input: An undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has a capacity C_e, a residual capacity R_e (depending on the demands supported on e) and an initial weight w for computing shortest path. A set of demands D, each demand has a volume of traffic D_{st}

```
Set up weight for all links $w_e$ = the number of demands
Sort the demands in random order
while $D_{st}$ is a demand in $D$ with no routing assigned do
    compute the shortest path $SP_{st}$ with the metric weight ($w_e$)
    assign the routing $SP_{st}$ to the demand $D_{st}$
    $\forall e \in SP_{st}, R_e = C_e - D_{st}$
    $\forall e \in SP_{st}, w_e = w_e - 1$
end
return the routing (if it exists) assigned to the demands in $D$.
```

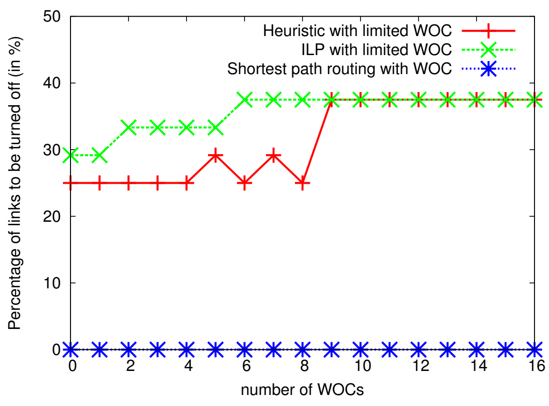
In step 2, we sort the “list-of-routes” found in Step 1 in increasing order of number of active links. We repeatedly select a route with the minimal number of active links in the “list-of-routes” as the input for Step 2 and try to find solution to enable WOCs (same as in the Case 1). The algorithm will stop when the number of WOC is less or equal than MAXWOC and the selected route is returned.

Algorithm 8: *Step 2* - finding a solution to turn on WOCs so that the number of enabled WOCs is minimized
Input: An undirected weighted graph $G = (V, E)$ where each edge $e \in E$ has capacity C_e. A list of feasible shortest routes (“list-of-routes”) found in Step 1. A set of demands D, each demand has a volume of traffic D_{st}

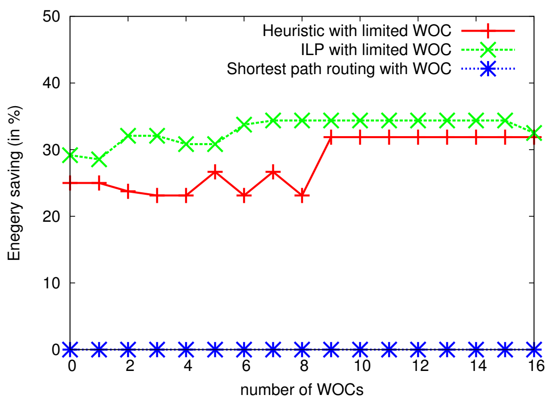
```
Sort the “list-of-routes” in increasing number of active links
while “list-of-routes” is not NULL do
    select and remove “routes” with the smallest number of active links from “list-of-routes”
    while network is congested and there are must-be-compressed-first flows do
        Find placement of WOCs for must-be-compressed-first flows
        Remove the compressed flows
    end
    while network is still congested do
        find placement of WOCs for remaining flows
    end
    if number of WOCs $\leq$ MAXWOC then
        return “routes” and placement of all used WOCs on the network.
    end
end
return “No solution”
```

4.3.3 Simulation results

We present the optimal and the heuristic results for grid 4 x 4 with the capacity/demand ratio is 60 and MAXWOC is increasing from 0 to 16 (the number of vertices in grid 4 x 4).



(a) Percentage of links to be turned off



(b) Percentage of energy savings

Figure 24: Simulation results of the version 2 algorithm for a 4 x 4 grid

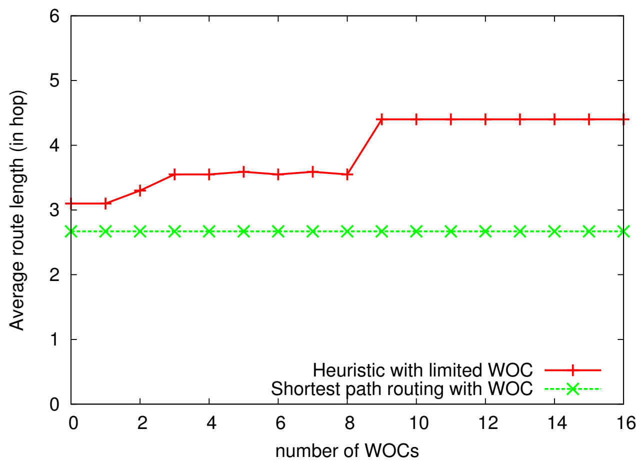


Figure 25: Average route length for a 4 x 4 grid

As shown in Fig. 24, our heuristic algorithm can work well and the result is close to the solution given by the ILP. However, at some points, when there are more WOCs to use, the heuristic and the ILP give solution in which energy consumption is increased. The ILP can not give optimal solution because we force the program to stop after a run time of one hour. For the heuristic, we set the program to stop when the number of enabled WOC is less than or equal to MAXWOC. Thus, the program may stop before finding the best solution. Finally, similar to the previous cases, the shortest path routing does not save energy and the energy-aware routing would take longer route (Fig. 25).

5 Conclusion and Future work

In this work, we present a new energy-aware routing model with WOC support. We formally define the model using Integer Linear Program and propose some greedy heuristic algorithms. The simulation on several network topologies show the significant gain in energy savings when we enable the WOCs. At least one third of the network links can be turned off for a usual range of demands. Moreover, in comparison with the traditional energy-aware routing without WOC, our model works better, especially when the capacity/demand ratio is small. We also prove by simulation that our heuristic algorithms work well and approximate to the near-optimal solutions given by the ILP. As part of future work, we will carry on more simulations where we consider a network topology with real capacity and real redundancy distribution on links. We will also consider about the impacts of energy-aware routing on the network such as the impact on network fault tolerance. Moreover, we plan to develop a distributed algorithm for the energy-aware routing with WOC.

Acknowledgments. We thank David Coudert, Frederic Giroire, Joanna Moulierac and Yaning Liu for their valuable feedback that helped improved our work. This work has been partly funded by ANR DIMA-GREEN (DesIgn and MAnagement of GREEN networks with low power consumption).

References

[1] "Akamai Technologies". http://www.akamai.com.

[2] A. Anand; C. Muthukrishnan; A. Akella and R. Ramjee. "Redundancy in Network Traffic: Findings and Implications". In *Proceedings of SIGMETRIC*, 2009.

[3] P. Mahadevan; P. Sharma; S. Banerjee and P. Ranganathan. "A Power Benchmarking Framework for Network Devices". In *Proceedings of IFIP NETWORKING*, 2009.

[4] "BlueCoat: WAN Optimization". http://www.bluecoat.com/.

[5] "Cisco Content Aware Networks - Some Areas of Interest". http://www.cisco.com/web/about/ac50/ac207/crc_new/ciscoarea/content.html.

[6] "Cisco Wide Area Application Acceleration Services". http://www.cisco.com/en/US/products/ps5680/Products_Sub_Category_Home.html.

[7] "IBM ILOG CPLEX Optimizer ". http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

[8] R. Bolla; R. Bruschi; F. Davoli and F. Cucchietti. "Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-Aware Fixed Network Infrastructures". In *IEEE Communication Surveys and Tutorials*, 2011.

[9] A. Wolman et al. "on the scale and performance of cooperative web proxy caching". In *Proceedings of ACM Symposium on Operating Systems Principles (SOSP)*, 1999.

[10] Global Action Plan. "An Inefficient Truth", 2007. http://globalactionplan.org.uk.

[11] T. Jr. Grevers and J. Christner. "Application Acceleration and WAN Optimization Fundamentals". In *Cisco Press*, 2007.

[12] Y. Song; K. Guo and L. Gao. "Redundancy-Aware Routing with Limited Resources". In *Proceedings of ICCCN*, 2010.

[13] M. Gupta and S. Singh. "Greening of the Internet". In *Proceedings of ACM SIGCOMM*, 2003.

[14] Maiju Kansanen. "Wide Area Network Acceleration in Corporate Networks". In *Master thesis*, 2009.

[15] M. Zhang; C. Yi; B. Liu and B. Zhang. "GreenTE: Power-aware Traffic Engineering". In *Proceedings of IEEE International Conference on Network Protocols (ICNP)*, 2010.

[16] L. Chiaraviglio; M. Mellia and F. Neri. "Energy-aware Backbone Networks: a Case Study". In *Proceedings of GreenComm*, 2009.

[17] L. Chiaraviglio; M. Mellia and F. Neri. "Reducing Power Consumption in Backbone Networks". In *Proceedings of IEEE International Conference on Communications ICC*, 2009.

[18] L. Chiaraviglio; M. Mellia and Fabio Neri. "Minimizing ISP Network Energy Cost: Formulation and Solutions". In *IEEE/ACM Transactions on Networking*, 2011.

[19] F. Giroire; D. Mazauric; J. Moulierac and B. Onfroy. "Minimizing Routing Energy Consumption: from Theoretical to Practical Results". In *Proceedings of GreenCom*, 2010.

[20] "PeerApp: P2P and Media Caching". http://www.peerapp.com.

[21] "Riverbed Networks: WAN Optimization". http://www.riverbed.com/solutions/optimize/.

[22] A. P. Bianzino; C. Chaudet; D. Rossi and J. Rougier. "A Survey of Green Networking Research". www.cl.cam.ac.uk/teaching/1011/R02/papers/green-survey.pdf.

[23] “Sage - an Open Source Mathematics Software”. http://www.sagemath.org/.

[24] A. Anand; V. Sekar and A. Akella. “SmartRE: An Architecture for Coordinated Network-wide Redundancy Elimination”. In *Proceedings of ACM SIGCOMM*, 2009.

[25] A. Anand; A. Gupta; A. Akella; S. Seshan and S. Shenker. “Packet Caches on Routers: The Implications of Universal Redundant Traffic Elimination”. In *Proceedings of ACM SIGCOMM*, 2008.

[26] Neil T. Spring and D. Wetherall. “A Protocol-independent Technique for Eliminating Redundant Network Traffic”. In *Proceedings of ACM SIGCOMM*, 2000.

[27] W. Fisher; M. Suchara and J. Rexford. “Greening Backbone Networks: Reducing Energy Consumption by Shutting Off Cables in Bundled Links”. In *Proceedings of ACM SIGCOMM workshop on green networking*, 2010.

[28] 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*, 2008.

[29] M. Webb. “SMART 2020: Enabling the Low Carbon Economy in the Information Age”. In *The Climate Group, London*, 2008.