Technische Universität Berlin
Telecommunication Networks Group

EWA – an adaptive algorithm using watermarks for energy saving in IP-over-WDM networks

Filip Idzikowski, Edoardo Bonetto, Luca Chiaraviglio
filip.idzikowski@tu-berlin.de
{edoardo.bonetto|luca.chiaraviglio}@polito.it

Berlin, May 2012

TKN Technical Report TKN-12-002

TKN Technical Reports Series

Editor: Prof. Dr.-Ing. Adam Wolisz

Abstract

We propose an adaptive algorithm, the Energy Watermark Algorithm (EWA), to reduce the power consumption of Internet Protocol (IP)-over-Wavelength Division Multiplexing (WDM) networks.

After presenting a network model and a detailed description of the EWA, we perform a sensitivity analysis and evaluate the performance of the algorithm considering real network scenarios. We show that the EWA can effectively switch off network devices, while limiting the costs of reconfiguration in the network.

Contents

List of Acronyms **2**

1 Introduction **3**

2 Network model **4**

3 Energy Watermark Algorithm **6**

3.1 General idea 6

3.2 Details of EWA 6

3.2.1 Ensuring Routability of Demands 6

3.2.2 Establishing Necessary Lightpaths 7

3.2.3 Releasing Unnecessary Lightpaths 10

3.3 Differences to related work 10

4 Scenarios **11**

4.1 Networks and traffic 11

4.2 Power and CapEx 13

5 Results **14**

5.1 Power consumption of Line Cards (LCs) 14

5.2 Break down of power over IP components 15

5.3 Costs of energy savings 16

6 Conclusion **19**

References **20**

List of Acronyms

CapEx Capital Expenditures

EWA Energy Watermark Algorithm

FCS Fabric Card Shelf

GA Genetic Algorithm

GMT Greenwich Mean Time

IP Internet Protocol

LC Line Card

LCS Line Card Shelf

LT Logical Topology

MDT Mountain Daylight Time

OXC Optical Cross-Connect

QoS Quality of Service

SBN Static Base Network

TM Traffic Matrix

TREND Towards Real Energy-efficient Network Design

WDM Wavelength Division Multiplexing

1 Introduction

According to different studies, the power consumption of the Internet ranges between 2% and 10% of the worldwide power consumption [1], and several initiatives are being put into place to reduce the power consumption of telecommunications networks. Of particular note is TREND [2], the Network of Excellence on energy-efficient networking which supported this work. Towards Real Energy-efficient Network Design (TREND) brings together manufacturers, telecom operators and universities with the aim of providing new solutions to efficiently manage energy consumption in networking devices. TREND targets energy efficiency for all the current network segments, ranging from access to core networks.

Core networks consume a non-negligible amount of power [3]. This is due to the fact that core devices exchange huge amount of data, hence consuming a large amount of power. Thus, a natural question is how to optimize a core network in terms of power consumption. Rather than working on single devices, in this work we consider an approach to *globally* reduce the power consumption of the network, by proposing an algorithm to selectively switch off network components. The switch-off is a promising approach to reduce power, since nowadays power consumption of current devices hardly scales with current load. Moreover, the load varies over time, with a peak during the day and an off peak during the night. This suggests that many devices could be potentially turned off during off-peak hours [4].

It is necessary, however, when some devices are switched off, to guarantee an adequate Quality of Service (QoS) using only the devices that are still powered on. In the literature, constraint on the maximum link utilization has been a popular way to guarantee the QoS, i.e., when devices are switched off the maximum link load is below a given threshold. However, this is not sufficient to guarantee a realistic QoS, because traffic is difficult to predict, and switching off a network device is a complex task. This is due to the fact that the switch-off process has to be coordinated with the other devices in order to move traffic from the current device to the other ones that remain powered on. However, the traffic shifting is not instantaneous, leading to potential losses of data and consequently QoS degradation. Additionally, even the switch-on process is not instantaneous, since it may take some time to power on the circuits and recover from a suspended state. For all these reasons, the energy-aware process has to be carefully planned in order to limit the QoS deterioration.

In this work, we propose EWA, an algorithm to reduce the power consumption of IP-over-WDM networks, in which LCs can be switched off and switched on based on the traffic variation over time. We search for appropriate settings of the EWA parameters in order to reduce power consumption of the network, and limit the frequency of changing of power state of each device leading to rerouting of traffic. Specifically, we consider the amount of traffic which has to be rerouted when some devices in the network change their power state.

EWA takes an adaptive approach and utilizes the network configuration from the previous time period. The algorithm is based on [5], but is more aggressive in switching off devices in order to save energy, and is able to adapt to higher changes of traffic between two consecutive time periods. We evaluate EWA over different scenarios, assuming as realistic assumptions as possible. Our results show that EWA is able to wisely adapt power consumption while reducing the amount of reconfigured traffic.

The rest of this technical report is organized as follows. The network model is presented in Section 2. EWA and its details are presented in Section 3. Results of an energy evaluation study based on realistic input data (Section 4) are presented in Section 5. Finally, conclusions are drawn in Section 6.

2 Network model

We consider an IP-over-WDM network, where Optical Cross-Connects (OXCs) are interconnected by fibers in the WDM layer, and IP routers are interconnected by lightpaths in the IP layer. IP routers have a modular structure and are composed of one Line Card Shelf (LCS) or of several LCSs interconnected by one or more Fabric Card Shelves (FCSs) [6]. LCs are located in LCSs and they are the end-points of lightpaths. A lightpath is a concatenation of WDM channels and is terminated by a transmitter and a receiver, both located in a LC of a router. Trunks of parallel lightpaths form a logical link and all the logical links together with IP routers constitute the Logical Topology (LT) of the network.

Notation More formally, the LT is modeled as a directed graph $H = (V, L)$ where V is the set of all nodes in the network and L is the set of supplied logical links on which lightpaths can be established. Each lightpath has bitrate C. Power consumed by a LC, LCS and FCS is denoted by $\mathcal{P}^{LC}$, $\mathcal{P}^{LCS}$ and $\mathcal{P}^{FCS}$, respectively.

We consider a set of time periods T consisting of past and future time periods ($T = T_{past} \cup T_{fut}$, $T_{past} \cap T_{fut} = \varnothing$). A Traffic Matrix (TM) $D(t)$ for each time period $t \in T$ contains traffic demands between the nodes $(a, b) \in V \times V$ with values $d^{ab}(t)$. Traffic exchanged during $T_{past} \subset T$ is used to determine the set of installed devices. We call this procedure Static Base Network (SBN) design, and use the Genetic Algorithm (GA) [7] with the objective of Capital Expenditures (CapEx) minimization for this purpose. The SBN is dimensioned to satisfy the maximum TM D_{SBN}, based on the set of past time periods T_{past}:

$$d^{ab}_{SBN} = \max_{t \in T_{past}} d^{ab}(t), \quad \forall a, b \in V \tag{1}$$

The EWA is executed at each t out of the set of future time periods $T_{fut} \subset T$. The duration of each time period $t \in T_{fut}$ is denoted as Δt. Since EWA is executed at each $t \in T_{fut}$, we introduce variables that are updated every Δt. The flow variables $f^{ab}_{ij}(t) \in \{0, 1\}$ denote whether the traffic demand originated at node a and targeted to node b traverses the logical link from i to j at time t. Single-, shortest-path routing of traffic demands over the LT is assumed. Moreover, the variables $y_l(t)$ determine the number of lightpaths established on the logical link $l \in L$ at time t, and determine the powered on LCs. Finally, $x^{LC}_i(t) \in \mathbb{Z}_+$is the number of LCs powered on at each node i at time t, which is bounded by the number of installed line cards X^{LC}_i in each node of the SBN.

Using the terms explained above, we define the network configuration as the set of network nodes V with installed LCs X^{LC}_i powered on or off, established lightpaths forming logical links $y_l(t)$ and IP routing of traffic demands $f^{ab}_{ij}(t)$.

Metrics A trade-off between reconfigured traffic and the power consumption is investigated in this work. Therefore we look at the following metrics.

Power consumption of all LCs active in the network as a function of time is defined as:

$$P^{LC}(t) = \mathcal{P}^{LC} \sum_{i \in V} x^{LC}_i(t) \tag{2}$$

Power consumption of active LCSs and FCSs is determined by the number of active LCs. The number of LCSs used at each node is expressed as:

$$x^{LCS}_i(t) = \lceil x^{LC}_i(t) / W_{LCS} \rceil \tag{3}$$

where W_{LCS} is the capacity (in terms of LCs) of a LCS. The number of FCS is determined by the number of LCSs:

$$x^{FCS}_i(t) = \begin{cases} 0 & \text{if } x^{LCS}_i(t) \leq 1 \\ \lceil x^{LCS}_i(t) / W_{FCS} \rceil & \text{otherwise} \end{cases} \tag{4}$$

where W_{FCS} denotes the maximum number of LCSs that a FCS can interconnect. Power consumption of active LCSs in the network is given by:

$$P^{LCS}(t) = \mathcal{P}^{LCS} \sum_{i \in V} x_i^{LCS}(t) \tag{5}$$

Similarly, the power consumption of active FCSs is defined as:

$$P^{FCS}(t) = \mathcal{P}^{FCS} \sum_{i \in V} x_i^{FCS}(t) \tag{6}$$

The total power consumption of the whole network is hence defined as:

$$P^{TOT}(t) = P^{LC}(t) + P^{LCS}(t) + P^{FCS}(t) \tag{7}$$

We consider the traffic that needs to be rerouted in order to reduce power consumption of the network. Therefore, let us define as $r_{ij}^{ab}(t) \in \mathbb{R}_+$ the amount of reconfigured traffic between a and b on the logical link from i to j at time t with respect to time $t-1$, with t belonging to the set of future time periods T_{fut} without the first time period (denoted as $t \in T_{futt}$):

$$r_{ij}^{ab}(t) = \begin{cases} d^{ab}(t) \cdot f_{ij}^{ab}(t) - d^{ab}(t-1) \cdot f_{ij}^{ab}(t-1) & f_{ij}^{ab}(t) > f_{ij}^{ab}(t-1) \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

We introduce the reconfiguration ratio over all subsequent pairs of time periods in T_{fut} as:

$$\xi = \frac{\sum_{t \in T_{futt}} \sum_{i \in V} \sum_{j \in V} \sum_{a \in V} \sum_{b \in V} r_{ij}^{ab}(t)}{\sum_{t \in T_{fut}} \sum_{a \in V} \sum_{b \in V} d^{ab}(t)} \tag{9}$$

This metric captures the amount of traffic which is reconfigured over all time periods in T_{fut}, normalized by the total amount of traffic which is exchanged in the network. The ξ may be greater than 1 since the reconfigured traffic is counted multiple times if it passes through multiple logical links form the source to the target.

Finally, we define the overload ratio metrics to capture overload traffic in all periods $t \in T_{fut}$. We look at the overload ratio at each $t \in T_{fut}$ in the network before reconfiguration determined by EWA, and in the network after the reconfiguration:

$$\phi^{PRE} = \frac{\sum_{t \in T_{fut}} \sum_{i \in V} \sum_{j \in V} \max\left(\sum_{a \in V, b \in V} d^{ab}(t) \cdot f_{ij}^{ab}(t-1) - \sum_{l \in L_{(i,j)}} C y_l(t-1), 0\right)}{\sum_{t \in T_{fut}} \sum_{a \in V} \sum_{b \in V} d^{ab}(t)} \tag{10}$$

$$\phi^{POST} = \frac{\sum_{t \in T_{fut}} \sum_{i \in V} \sum_{j \in V} \max\left(\sum_{a \in V, b \in V} d^{ab}(t) \cdot f_{ij}^{ab}(t) - \sum_{l \in L_{(i,j)}} C y_l(t), 0\right)}{\sum_{t \in T_{fut}} \sum_{a \in V} \sum_{b \in V} d^{ab}(t)} \tag{11}$$

Flow variables $f_{ij}^{ab}(t-1)$ at previous time period $t-1$ (before reconfiguration) are used in Eq. (10), and flow variables $f_{ij}^{ab}(t)$ at current time period t (after reconfiguration) are used in Eq. (11). Both ϕ^{PRE} and ϕ^{POST} should be as low as possible (ideally 0) to prevent service disruptions and loss of QoS. Note that we take into account each hop that a demand passes through when calculating the overload, even though traffic can be dropped only once in the network. Therefore, ϕ^{PRE} and ϕ^{POST} may also be greater than 1. Since EWA adapts the network configuration to the current traffic demands $\phi^{PRE} \geq \phi^{POST}$.

3 Energy Watermark Algorithm

We first present the general idea of the EWA, then we detail the algorithm and finally we show the differences with respect to related work.

3.1 General idea

The EWA adapts the network to current traffic situation in order to save energy on one hand, and limit the load on logical links in order to ensure certain QoS on the other hand. EWA uses a low and a high watermark (W_L and W_H) defined as thresholds on the utilization of the last lightpath on a logical link. Exceeding the W_H triggers attempts to establish additional lightpath(s) in order to avoid overload of the network. Exceeding the W_L triggers attempts to release lightpath(s) in order to switch off idle LCs and save energy. EWA makes sure that the maximum utilization of last lightpath on a logical link ψ is not exceeded when trying to release lightpaths.

Alg. 1 shows the main pseudocode of EWA. Details of its subroutines with corresponding pseudocodes are presented in the next section. The algorithm takes as input the network configuration in previous time period $t-1$ (network nodes V with installed LCs X_i^{LC}, established lightpaths forming logical links $y_l(t-1)$ and IP routing of traffic demands $f_{ij}^{ab}(t-1)$), TM $D(t)$ for the current period t, capacity of a lightpath (WDM channel) C, W_L, W_H and ψ. Updated network configuration is returned as output of the algorithm.

EWA first checks whether all the demands in the current network configuration are routable, and iteratively tries to establish additional lightpath(s) for the unroutable demands (if any), starting from the largest ones (line 1 and Alg. 2). The logical links on which watermarks are exceeded are identified next (line 2), and violation of the W_H is checked, starting from the logical links with the highest utilization of the last lightpath (line 3 and Alg. 3). For each overloaded logical link (from the most overloaded to the least overloaded), the algorithm first tries to increase the capacity of the logical link if a demand with the same source and target flows through it. If this is not the case, attempts are made to establish lightpath(s) for the possibly biggest demand flowing through the overloaded logical link.

Once load is lower than W_H for all logical links, or it is impossible to reduce overload anymore, violation of the W_L is checked starting from the least loaded logical links (line 4 and Alg. 4). One lightpath per iteration is tried to be released making sure that ψ is not exceeded.

Algorithm 1 Pseudo-code of EWA.

Input: netConfig from period $t-1$, current TM $D(t)$, C, W_L, W_H, ψ
Output: Updated netConfig
1: ensureDemandsRoutability(netConfig, $D(t)$, C);
2: sortedLLsExceedingWMs = getSortedLLsExceedingWMs(netConfig, $D(t)$, W_L, W_H);
3: establishNecessaryLpaths(netConfig, $D(t)$, C, sortedLLsExceedingWMs, W_L, W_H);
4: releaseUnnecessaryLpaths(netConfig, $D(t)$, sortedLLsExceedingWMs, W_L, W_H, ψ);

3.2 Details of EWA

Three subroutines of Alg. 1 are presented in this section.

3.2.1 Ensuring Routability of Demands

The main requirement on an energy-saving algorithm is not to influence the connectivity of the network. This means that given a set of traffic demands, there must be at least one path in the LT to route each traffic demand. Therefore the first step of EWA is to ensure that all the demands are routable. The corresponding subroutine is shown in Alg. 2. The current network configuration, current TM $D(t)$, and capacity C of a lightpath are provided as input. Updated network configuratoin is returned by the subroutine.

In the first step, the unroutable demands are identified (line 1). Then, logical link(s) are iteratively tried to be established between the source and target nodes of the unroutable demands as long as there are unroutable demands, or it is impossible to establish corresponding logical links any more (line 3). For this reason the unroutable demands need to be sorted in a descending order according to their values (lines 4 – 5) if this has not been done so far (indicated by the demandIndex equal to 1, see lines 2 and 9). If a logical link is successfully established (line 7), then the list of unroutable demands is updated (lines 8 – 9). Note that a new logical link may change routing of several traffic demands, not only the addressed unroutable ones. If establishing of a logical link was unsuccessful, the next biggest demand is chosen (line 11) in order to establish a logical link between its source and target nodes. The algorithm terminates, when all demands are routable or it is impossible to create a logical link between source and target nodes of any unroutable demand (line 13).

Algorithm 2 Pseudo-code of ensureDemandsRoutability.

Input: netConfig, TM $D(t)$, C
Output: Updated netConfig

1: unrtableDemands=getUnrtableDemands(netConfig, TM);
2: demandIndex=1;
3: **while** (unrtableDemands.size()!=0) && (demandIndex $\leq$ unrtableDemands.size()) **do**
4: **if** demandIndex==1 **then**
5: unrtableDemands=sortDemands(unrtableDemands);
6: **end if**
7: **if** establishLL(unrtableDemands(demandIndex), C, netConfig) **then**
8: unrtableDemands=getUnrtableDemands(netConfig, TM);
9: demandIndex=1;
10: **else**
11: demandIndex++;
12: **end if**
13: **end while**

3.2.2 Establishing Necessary Lightpaths

Once the routability of demands is ensured, the appropriate capacity of the network needs to be provided in order to meet the current traffic demands. The subroutine shown in Alg. 3 takes as input (additionally to the TM $D(t)$) current network configuration, capacity C of a lightpath, sorted list of logical links where the watermarks are exceeded, and the values of the low and high watermarks W_L and W_H. Updated network configuration and sorted list of logical links exceeding watermarks are returned as output.

The subroutine starts with the logical link with the highest utilization of the last lightpath (line 1), and tries to off-load the logical links in the decreasing order of their utilization, as long as the high watermark W_H is exceeded at the currently visited logical link LL (lines 2 – 3). All demands on the current logical link LL are identified (line 4), and a flag signaling the failure of lightpath establishment is set to true (line 5). This flag is needed to stop the attempts to establish a lightpath in the case a lightpath has already been established (see line 14) and to iterate through logical links (see line 41). Indeed the choice of the source and target of the lightpath to be established depends on the overloaded logical link LL, the TM (or precisely the value of the demand with the same source and target nodes as the LL), and the current capacity of the logical link LL (lines 6 and 12):

1. If the demand value exceeds the capacity of the logical link LL (line 6), the capacity of the LL is tried to be exceeded by establishing additional lightpath(s) in parallel to the existing ones (lines 7 – 8). In the case of a success the flag is set to false (line 9), and the sorted list of logical links exceeding watermarks is recalculated (line 10).

2. If either the demand with the same source and target as the logical link LL does not flow through LL or this demand does not exist in the TM or its value does not exceed the capacity of the logical link LL (line 12), another strategy is taken to choose lightpaths to be established to resolve violation of W_H. The demands flowing through the logical link LL are sorted from the biggest one to the smallest one (line 13). While the set of demands is not empty, and no additional lightpath has been established (line 14), the biggest (non-zero) demand flowing through the logical link LL is searched for (lines 16 – 30) under the constraint that the source and end node have sufficient number of available transmitters and receivers to establish a logical link of needed capacity (lines 19 – 20). If this requirement cannot be fulfilled, the corresponding demand is removed from the list of the demands that can potentially be offered a new (direct) ligthpath (line 25). The same happens if the corresponding traffic demand does not exist in the current TM (lines 27 – 28).

   Once the demand corresponding to the maximum load has been found and has a positive value (line 31), capacity of a new logical link in terms of lightpaths is calculated (line 32), and the new logical link is tried to be established (line 33). If this attempt is successful, the flag is set to false (line 34), and the sorted list of logical links exceeding watermarks is recalculated (line 35). In any case, the corresponding demand with the same source and target as the LL attempted to be established is removed from the list of demands that can potentially be offered a new (direct) logical link (line 37).

Eventually, if no new lightpath was established in the current iteration (line 41 out of the loop in lines 2 – 46), the logical link index is decremented (line 42), that is the logical link with the next biggest utilization of the last lightpath is considered. In the case of success of the establishment of a lightpath, the logical link index is reset (line 44) to indicate again the logical link where utilization of the last lightpath is the highest.

Although the subroutine seems to be quite complex, the loops usually need to be visited only few times due to the adaptive character of the EWA. This is however highly dependent on the characteristics of the traffic, which is expected to smoothly follow the day-night pattern in the core network due to the aggregation.

Algorithm 3 Pseudo-code of establishNecessaryLpaths.

Input: netConfig, TM $D(t)$, C, sortedLLsExceedingWMs, W_L, W_H
Output: Updated netConfig and sortedLLsExceedingWMs
1: LLindex=sortedLLsExceedingWMs.size();
2: **while** (LLindex > 0) && (sortedLLsExceedingWMs(LLindex).utilLastLpath() $> W_H$) **do**
3: LL=sortedLLsExceedingWMs(LLindex);
4: demands=LL.getDemandsOnLL(netConfig);
5: failedToEstablishLpath=true;

```
    if (demands.contains(LL.id())) && (TM.contains(LL.id())) && (TM.value(LL.id()) > LL.capacity()) then
      neededCap=ceil(LL.getTotalLoad(netConfig, TM)/C) - LL.capacity();
      if establishLL(LL.src(), LL.tgt(), neededCap, netConfig, TM) then
        failedToEstablishLpath=false;
        sortedLLsExceedingWMs = getSortedLLsExceedingWMs(netConfig, TM, W_L, W_H);
      end if
    else
      demands=sortDemands(demands, TM);
      while (demands.size() > 0) && failedToEstablishLpath do
        maxLoad=0.0; newLLsrc=null; newLLtgt=null;
        for i=1; i≤demands.size(); i++ do
          demand=demands.get(i);
          if TM.contains(demand.id()) then
            neededCap=ceil(TM.getValue(demand.id())/C);
            if (TM.value(demand.id()) > maxLoad) && (demand.src().availTXs() ≥ neededCap) && (demand.tgt().availRXs() ≥ neededCap) then
              maxLoad=TM.value(demand.id());
              newLLsrc=demand.src();
              newLLtgt=demand.tgt();
            else if (demand.src().availTXs() < neededCap) || (demand.tgt().availRXs() < neededCap) then
              demands.remove(demand);
            end if
          else
            demands.remove(demand);
          end if
        end for
        if maxLoad > 0 then
          neededCap=ceil(maxLoad/C);
          if establishLL(newLLsrc, newLLtgt, neededCap, netConfig, TM) then
            failedToEstablishLpath=false;
            sortedLLsExceedingWMs = getSortedLLsExceedingWMs(netConfig, TM, W_L, W_H);
          end if
          demands.remove(demand(newLLsrc, newLLtgt));
        end if
      end while
    end if
    if failedToEstablishLpath then
      LLindex--;
    else
      LLindex=sortedLLsExceedingWMs.size();
    end if
  end while
```

3.2.3 Releasing Unnecessary Lightpaths

The last step of the EWA outlined in Alg. 1 is to attempt to release unnecessary lightpaths in order to save energy. Alg. 4 shows the procedure of releasing unnecessary lightpaths.

The following data is provided as input to the algoirhtm: current network configuration, TM $D(t)$, sorted list of logical links exceeding watermarks, the low and high watermarks (W_L and W_H) the watermarks themselves as well as the maximum utilization of the last lightpath on a logical link ψ. Updated network configuration and sorted list of logical links exceeding watermarks are returned as output.

The logical links are visited starting from the one having the lowest utilization of the last lightpath as long as there are unvisited logical links exceeding the low watermark W_L (lines 1 – 2). A flag is used in a similar way as in Alg. 3. The flag in Alg. 4 indicates the failure of the lightpath release, and not establishment. It is also initialized as true (line 3). Logical links are taken from the sorted list (line 4), and a single lightpath out of all lightpaths constituting this logical link is attempted to be released (line 5) keeping the constraint of routability of all demands in the current TM, and the constraint on the maximum utilization of the last lightpath on all the logical links ψ. If the attempt is successful, the flag is changed to true (line 6), and the sorted list of logical links exceeding watermarks is recalculated (line 7). Eventually, depending on the state of the flag (line 9), the index of the logical link is either incremented (line 10) to visit the next logical link if no lightpath has been released, or reset to 1 (line 12) to start again with the logical link with the lowest utilization of the last lightpath if a lightpath has been released.

Algorithm 4 Pseudo-code of releaseUnnecessaryLpaths.

Input: netConfig, TM $D(t)$, sortedLLsExceedingWMs, W_L, W_H, ψ
Output: Updated netConfig and sortedLLsExceedingWMs

```
1:  LLindex=1;
2:  while (LLindex ≤ sortedLLsExceedingWMs.size()) &&
    (sortedLLsExceedingWMs(LLindex).utilLastLpath() < W_L) &&
    (sortedLLsExceedingWMs.size() > 0) do
3:    failedToReleaseLpath=true;
4:    LL=sortedLLsExceedingWMs(LLindex);
5:    if releaseLastLpath(LL, netConfig, TM, ψ) then
6:      failedToReleaseLpath=false;
7:      sortedLLsExceedingWMs = getSortedLLsExceedingWMs(netConfig, TM, W_L, W_H);
8:    end if
9:    if failedToReleaseLpath then
10:     LLindex++;
11:   else
12:     LLindex=1;
13:   end if
14: end while
```

3.3 Differences to related work

To the best of authors' knowledge, no heuristic based on watermarks triggering establishing and releasing of lightpaths has been investigated so far in the literature focusing on energy consumption. Our work was inspired by [5]. The main difference to [5] (apart from the focus) is that EWA does not forbid deleting lightpath(s) in the case when some lightpath(s) have already been added in the same run of the algorithm. This allows more aggressive attempts to release lightpaths leading to power saving and is not critical for the operation of the network, since the releasing of lightpaths is performed in the last step of the algorithm (line 4 of Alg. 1). The lightpath(s) are deleted only after establishing the new ones, and the necessary rerouting can be performed in a controlled way. Moreover, differently from [5], EWA is able to add or delete more than one lightpath during the execution of the algorithm. Preliminary investigations have shown that adding or deleting of a single lightpath is insufficient to adapt the network to the traffic changes based on input data originating from measurements. Finally, the parameter ψ is added in order to trade between QoS and energy saving.

Another work which uses the concept of watermarks is [8]. The authors of [8] introduce the load balance indicator (bound on maximum lightpath load of the new logical topology at reconfiguration point), which is similar to ψ, however defined on a lightpath, and not on (the last lightpath of) a logical link. Moreover, the watermarks are used only to trigger solving the optimization problem, which does not use the watermarks themselves, but only the load balance indicator. The optimization problem gives the possibility to re-distribute the load in the network by rerouting only, without adding or deleting lightpaths, however solving an optimization problem is time consuming and therefore impractical in real operation. The authors of [8] propose also a so-called softbound approach, where reconfiguration is done only after three observation periods since the previous reconfiguration if in between there is no violation of the high watermark. Eventually an approximate mathematical model is proposed in [8], where the added and deleted lightpath(s) are chosen among some candidate lightpaths. Algorithms for the selection of the candidate lightpaths are also described in [8]. They perform the routing and wavelength assignment in order to reduce the complexity of the mathematical formulation.

4 Scenarios

In the following, we introduce the networks and the traffic scenarios that we are considering. Then, we detail the power and the CapEx values that we are using.

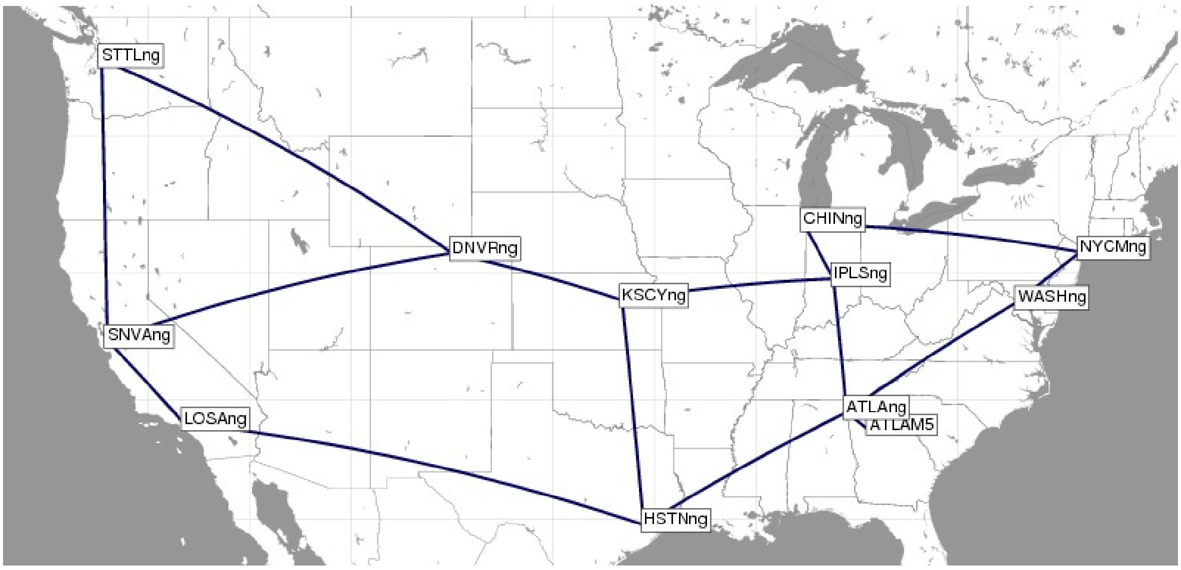
4.1 Networks and traffic

We select two networks, Abilene and Géant, which are represented respectively in Fig. 1(a) and Fig. 1(b) [9]. During the evaluation of the results, we use traffic demands measured on these networks, and available at [9].

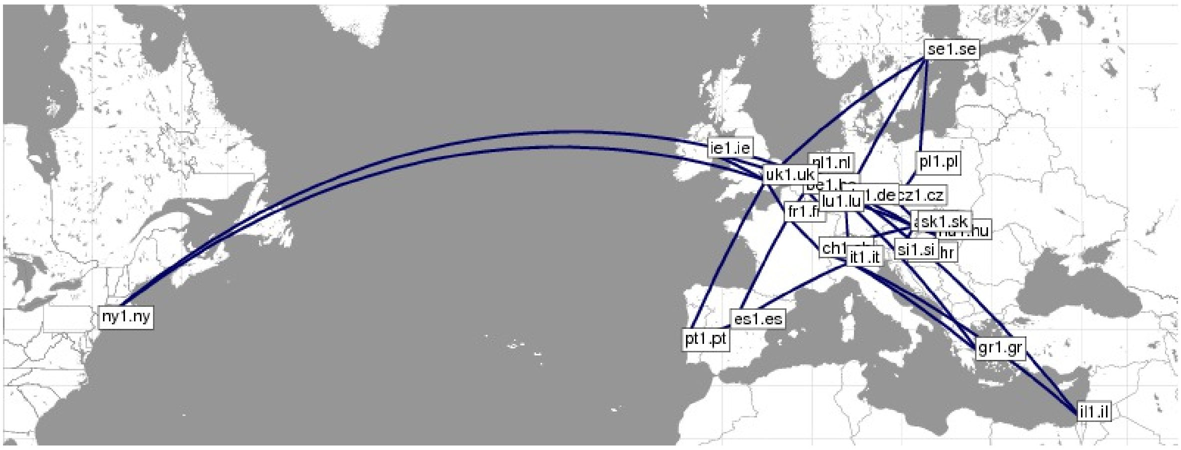
The demand values in the original TMs [9] are low with respect to the capacities of the current WDM systems. In order to provide comparable load to both networks, we scale the TM used for the design of the SBN D_{SBN} so that the total demand per node $d_{|V|} = \sum_{(a,b)\in V\times V} d^{ab}_{SBN}/|V|$ is equal to 300 Gbps per second per node. We introduce the unit 'Gpn' to abbreviate the 'Gbps per node'. We use the same scaling factor as at the SBN design for scaling the TMs used when evaluating energy savings with EWA.

Concerning the design of the SBN design, we consider as T_{past} a time period equal to one month. In particular, we select the period between 2004-07-01 and 2004-07-31 for Abilene (Fig. 2(a)[1]) and between 2005-05-05 and 2005-06-04 for Géant (Fig. 2(b)[1]), and allow the maximum utilization of the logical links equal to 0.5 at the SBN design. The TM of the period 2005-05-27 at 17:45 (Géant) in the original data [9] is higher by at least two orders of magnitude than other TMs. We assume that this is a measurement error, and scale all the values in this TM by the factor 0.00001. Moreover, we point out

[1]Zoom to see details of Fig. 2 and Fig. 3.



(a) Abilene

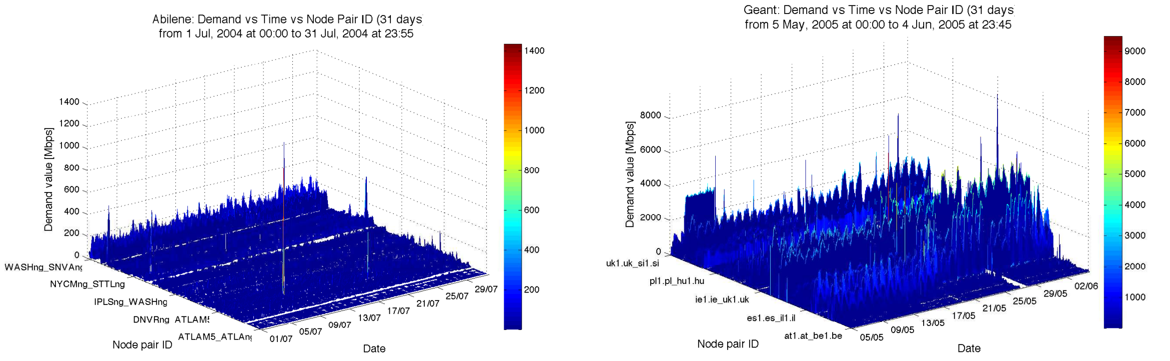


(b) Géant

Figure 1: Physical supply network topologies [9]

that the traffic demand between the nodes CHINng and LOSAng (in both directions) in the Abilene network is significantly higher than other demands. Therefore we do not plot it in Fig. 2.

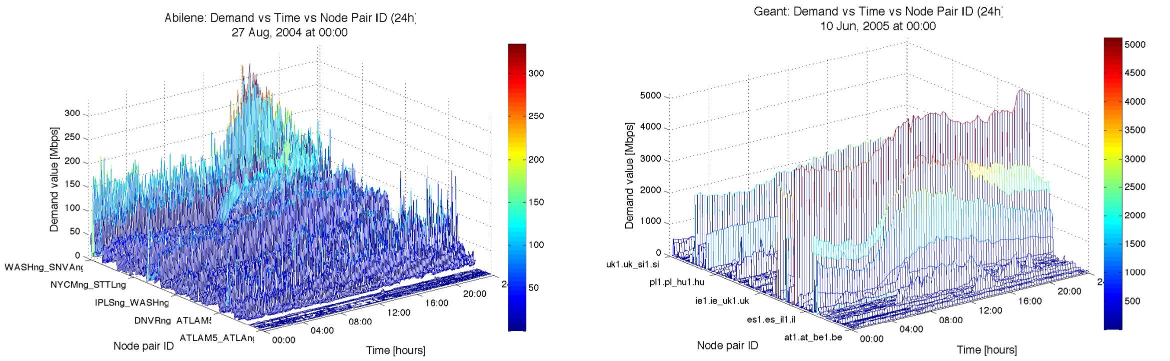
The energy savings achieved with EWA are evaluated using traffic data measured on 2004-08-27 and 2005-06-10 for Abilene and Géant respectively. We set Δt equal to 15 minutes. This is the original granularity for the Géant TMs available at [9]. The original granularity of the Abilene TMs available at [9] is 5 minutes. Therefore we create the TMs of granularity equal to Δt by selecting the maximum values out of the three corresponding original TMs. The original data is depicted in Fig. 3,[1] and shows that the traffic in the Géant network is smoother than in the Abilene. Variation of the total demand (sum of all elements of a TM, $\sum_{(a,b)\in V\times V} d^{ab}(t)$ for each $t \in T_{fut}$) of the actual (scaled) data used for the evaluation of energy savings achieved with EWA is shown in Fig. 4 for both Abilene and Géant.



(a) Abilene between 2004-07-01 and 2004-07-31 GMT (without the traffic demands between CHINng and LOSAng in both directions)

(b) Géant between 2005-05-05 and 2005-06-04 GMT (without 2005-05-27 at 17:45)

Figure 2: Demands between all node pairs over time for the traffic data used for the SBN design (original data from [9])



(a) Abilene on 2004-08-27 GMT

(b) Géant on 2005-06-10 GMT

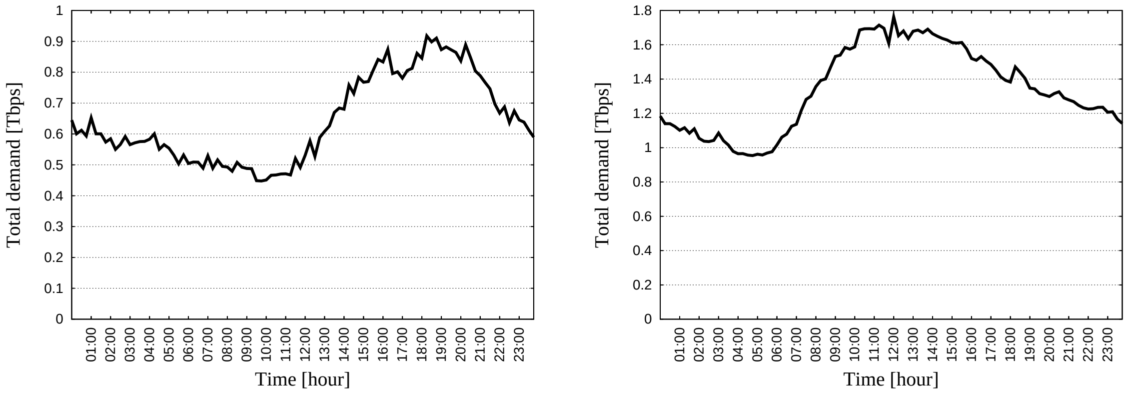
Figure 3: Demands between all node pairs over time for the traffic data used for EWA evaluation of energy savings (original data from [9])

4.2 Power and CapEx

We select the Cisco CRS-1 router [10] as reference for the IP router parameters. In particular, the power consumption values have been taken from [11], while we refer to [6] for the CapEx normalized values that have been used for the design of the SBNs.

Detailing the values, we have that a LCS consumes $\mathcal{P}^{LCS}$ = 2920 W and it cost 16.67 units, the power consumption of a FCS is $\mathcal{P}^{FCS}$ = 9100 W and its cost is equal to 53.35, while a LC consumes $\mathcal{P}^{LC}$ = 500 W and its cost is equal to 13.37 units.

Each LC is constituted by the “Cisco CRS-1 1-Port OC-768c/STM-256c Tunable WDMPOS (Wavelength Division Multiplexing Packet over SONET/SDH) Interface Module” and the “Cisco CRS-1 Modular Service Card”, operating at a bit rate equal to 40 Gbps. Thus, each lightpath has capacity C equal to 40 Gbps.



(a) Abilene on 2004-08-27 GMT (b) Géant on 2005-06-10 GMT

Figure 4: Total demand over time for the traffic data used for EWA evaluation of energy savings (actual data used in our study)

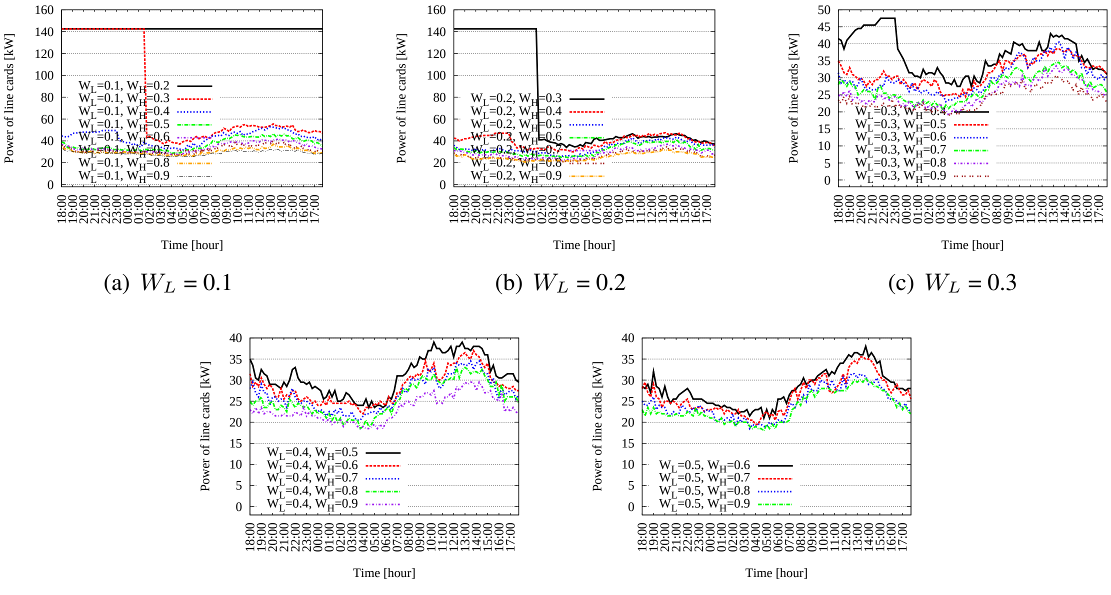
5 Results

We run EWA on a personal computer equipped with a Dual Core CPU at 2.4 GHz and 2 GB of RAM using a Java framework. We consider different networks (Abilene and Géant), different algorithm parameters (W_L and W_H, and $\psi = W_H$), and performance metrics (reconfigured traffic and overload).

5.1 Power consumption of Line Cards (LCs)

We first look at the power consumed by LCs, since we assume that it is easier to switch off this type of devices rather than to switch off LCSs and FCSs. Fig. 5 reports the total variation of power consumed by LCs over time in the Abilene network [2], for different values of W_L and W_H. Fig. 5(a) reports the results for W_L = 0.1 and different values of W_H. When both the thresholds are low, the active LCs tend to consume more power. This is due to the fact that with this setting the algorithm rarely tries to switch off lightpaths (low W_L), and many devices are required to be powered on due to overprovisioning introduced by low W_H and ψ. In particular, power is constant for W_L = 0.1 and W_H = 0.2 (Fig. 5(a)), since the maximum utilization of the last lightpath on a logical link ψ is equal to the W_H in our study, and it is impossible to keep the utilization of the last lightpath below 0.2. In this case EWA keeps all the LCs active in order to reduce the violation of ψ. However, violation of ψ does not frequently occur with increasing W_H and ψ, and the algorithm adapts the network configuration to the traffic evolving over time. Two pairs (W_L; W_H) are of particular notice, namely (0.1; 0.3) in Fig. 5(a) and (0.2; 0.3) in Fig. 5(b). Both of them share the same value of W_H and ψ. EWA cannot adapt the SBN in the beginning of its run with these two settings. It is impossible to establish new lightpaths in the first and the second stage of EWA (lines 1 and 3 of Alg. 1) and reduce the violation of W_H, because all the LCs are active in the SBN. Consequently, it is impossible to switch off any LCs in the final stage of EWA (lines 4 of Alg. 1), because it could increase the violation of both W_H and ψ. However, once the traffic demands get lower (more precisely at 01:45 am MDT, see both Fig. 5(a) and Fig. 5(b)) EWA manages to find a network configuration, where ψ is exceeded on none of the logical links, and consequently manages to switch off line cards in the network. Please note that

[2]We assume that the timestamps of the original TMs available at [9] are GMT, but use Mountain Daylight Time (MDT) for Abilene in the plots in order to reflect the day-night pattern.



(a) $W_L = 0.1$ (b) $W_L = 0.2$ (c) $W_L = 0.3$

(d) $W_L = 0.4$ (e) $W_L = 0.5$

Figure 5: Power consumption of LCs in the Abilene network.

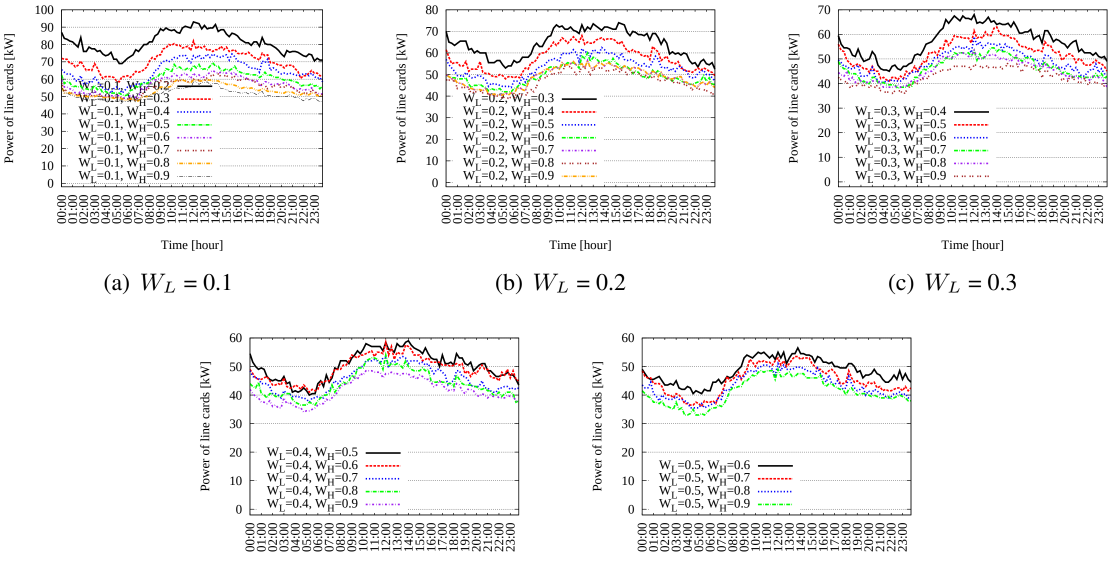
violation of ψ at a single logical link is sufficient to prevent EWA from switching off any lightpaths. It is interesting to note that EWA manages to adaptively change the network configuration so that the violation of ψ does not happen any more when the traffic raises again (around 2 pm MDT).

To give more insight we extend our analysis considering $W_L = 0.3$ (Fig. 5(c)), $W_L = 0.4$ (Fig. 5(d)) and $W_L = 0.5$ (Fig. 5(e)). The power consumption is lower as W_L increases, since the algorithm becomes more aggressive in switching-off lightpaths. Moreover, the increase of W_H further reduces the power consumption.

We then extend our analysis by considering the Géant network. Fig. 6 reports the variation of power for different W_L and W_H. Differently from the Abilene network, in this case the power follows the traffic trend even for $W_L = 0.1$ and $W_H = 0.2$ (reported in Fig. 6(a)). This is due to the fact that we start our evaluation at low demand hour (12:00 am GMT), and therefore the switch off attempts performed at the original configuration of the SBN does not trigger the violation of ψ. EWA then manages to adapt the network configuration to the increasing traffic demands, even during the peak hours. Similar observation as for Abilene holds also for Géant, that the power consumed by LCs decreases with increasing values of W_L and W_H.

5.2 Break down of power over IP components

We consider the case in which also LCSs and FCSs can be switched off in the periods of low traffic. Fig. 7 details the power consumption of active LCs, LCSs and FCSs in the network for different values of W_L and W_H. Activation and deactivation of FCSs and LCSs occurs less frequently than activation and deactivation of LCs. However, the impact of switching a shelf on or off on the total power consumption of the network is significant due to high values of power consumption of single components (see Section 4.2). These effects can be observed in Fig. 7(c) starting from 06:00 am MDT. Especially high jumps can be seen for FCSs (blue dotted line). This is due to the fact that a FCS is the most power hungry component (recall from Section 4.2 that $\mathcal{P}^{FCS}$ is equal to 9100 W). The frequent



(a) $W_L = 0.1$ (b) $W_L = 0.2$ (c) $W_L = 0.3$

(d) $W_L = 0.4$ (e) $W_L = 0.5$

Figure 6: Power consumption of LCs in the Géant network.

changes of the power consumed by FCSs in Fig. 7(a) can be explained by the fact that as the traffic in the network increases, FCSs need to be activated in several nodes of the network.

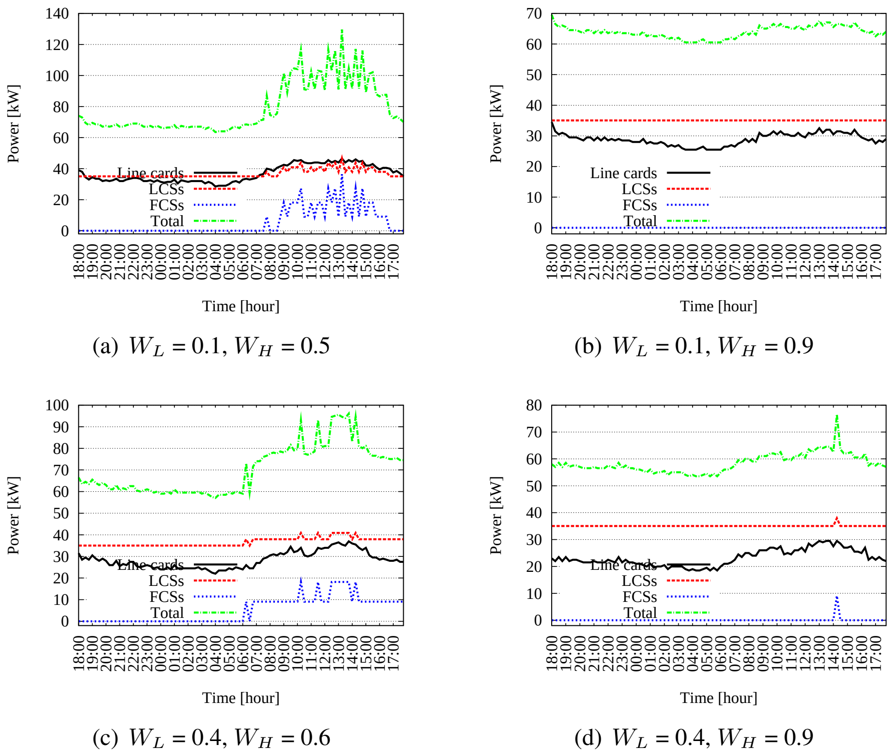
The power consumption of LCs is always higher than the power consumption of FCSs and comparable with the power consumption of LCSs, even though a single LC consumes significantly less power than a single LCS or FCS. This is due to the fact that there are much more LCs in the network than LCSs and FCSs.

Regarding the Géant network, similar observations can be made in Fig. 8. However, power consumed by LCSs and FCSs changes less frequently due to smoother traffic, and power of LCSs is consistently slightly higher than power of LCs.

5.3 Costs of energy savings

Finally, we evaluate the performance of EWA considering the reconfigured traffic and the overload before and after reconfiguration (Eq. (9), Eq. (10) and Eq. (11), respectively).

Fig. 9 reports the results for the Abilene network. Interestingly, the reconfigured traffic is relatively small, i.e., typically lower than 0.2. In particular, the reconfigured traffic is reduced for $W_L = 0.1$. This is due to the fact that, with this configuration, the network is less aggressive in deleting lightpaths. On the contrary, as W_L increases, the reconfigured traffic tends to increase, suggesting that there exists a trade-off between reducing power consumption (high W_L) and reducing the reconfigured traffic (low W_L). Moreover, the reconfigured traffic does not consistently depend on the values of W_H. Fig. 9 reports also the overload before (pre-dropped) and after the reconfiguration process occurs (post-dropped). Interestingly, the pre-dropped traffic fraction tends to increase with W_H. This is due to the fact that, as W_H is close to 1, utilization of lightpaths is higher, and the probability that an overload before reconfiguration occurs is also higher in the case of traffic increase. However, the post-dropped traffic fraction is always zero. This suggests that the EWA can adapt the network to the changing traffic. However, the granularity of the TMs assumed in this study ($\Delta t = 15 min.$) seems to be too

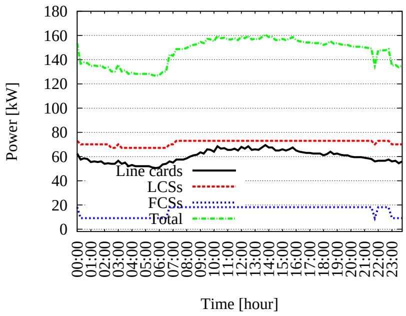


(a) $W_L = 0.1, W_H = 0.5$
(b) $W_L = 0.1, W_H = 0.9$
(c) $W_L = 0.4, W_H = 0.6$
(d) $W_L = 0.4, W_H = 0.9$

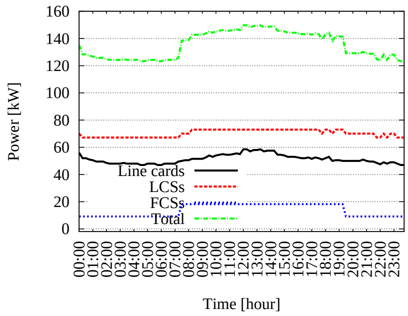
Figure 7: Power consumption in the Abilene network.

low to avoid traffic drops before the network is adapted to the changing load. In real operation the choice of the observation period does not depend on the available set of traffic data and its granularity Δt. Therefore the observation period (corresponding to Δt) should be carefully chosen in order to capture the changes of traffic, but also avoid unnecessary network reconfigurations. Usage of ψ can be useful to reduce the unnecessary reconfiguration in the network.

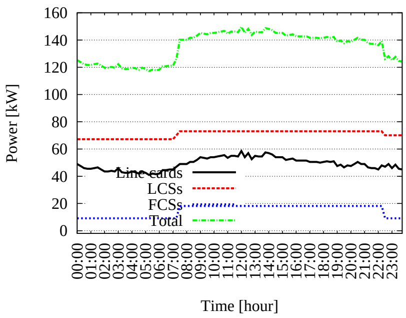
Both the reconfigured traffic fraction, and the pre-dropped traffic fraction are lower in the Géant network than in the Abilene network (compare Fig. 9 and Fig. 10). This can be again explained by the fact that the traffic in the European backbone network is smoother than in the American one. We stress again the fact the no traffic is dropped after the reconfiguration, which means that sufficiently quick reaction to changing traffic conditions together with correct setting of the parameters W_H and ψ should eliminate traffic drops in the normal operation of a real backbone network.



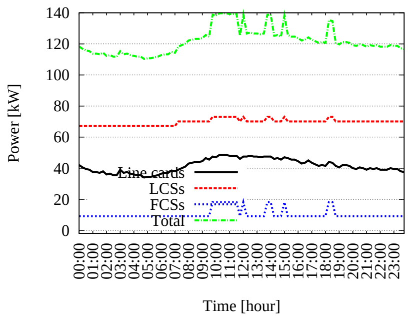
(a) $W_L = 0.1$, $W_H = 0.5$



(b) $W_L = 0.1$, $W_H = 0.9$

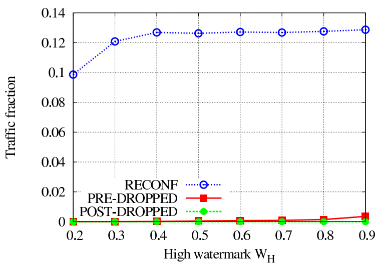


(c) $W_L = 0.4$, $W_H = 0.6$

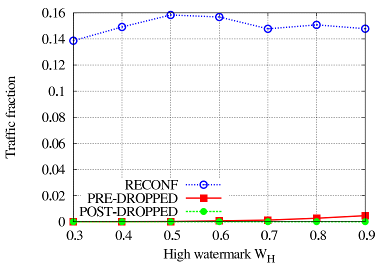


(d) $W_L = 0.4$, $W_H = 0.9$

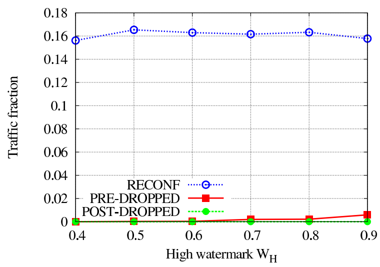
Figure 8: Power consumption in the Géant network.



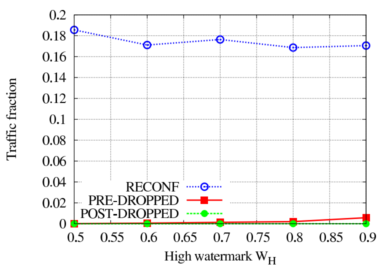
(a) $W_L = 0.1$



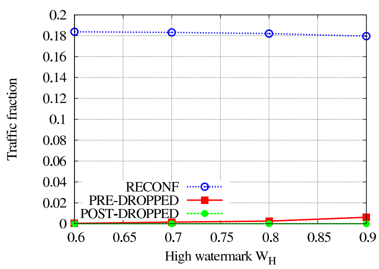
(b) $W_L = 0.2$



(c) $W_L = 0.3$

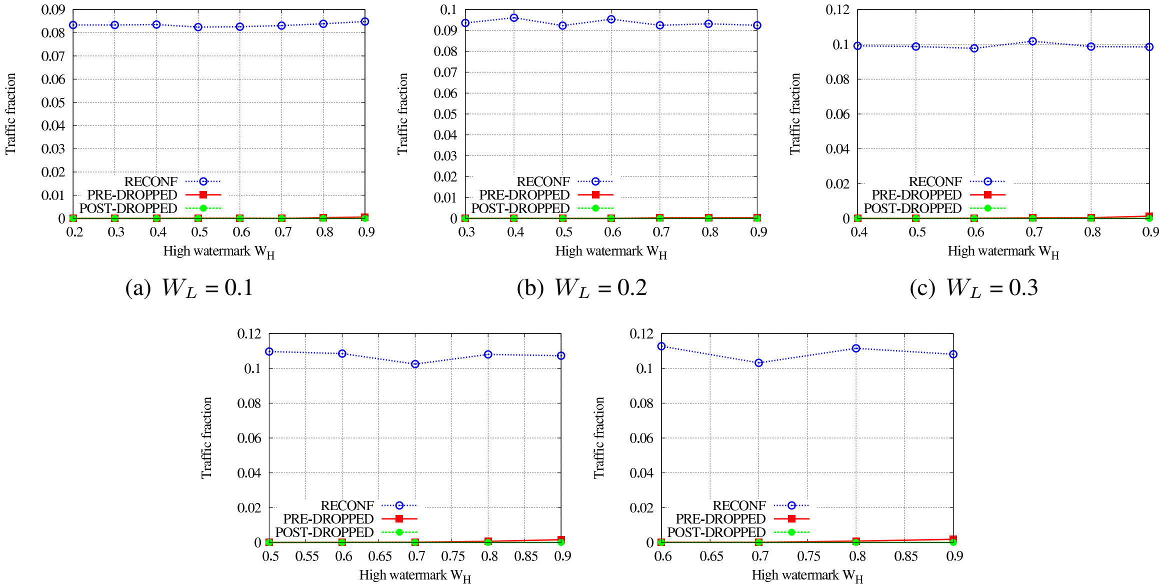


(d) $W_L = 0.4$



(e) $W_L = 0.5$

Figure 9: Reconfigured traffic and dropped traffic before (PRE-DROPPED) and after (POST-DROPPED) reconfiguration as fraction of the total traffic in the Abilene network.



(a) $W_L = 0.1$ (b) $W_L = 0.2$ (c) $W_L = 0.3$

(d) $W_L = 0.4$ (e) $W_L = 0.5$

Figure 10: Reconfiguerd traffic and dropped traffic before (PRE-DROPPED) and after (POST-DROPPED) reconfiguration as fraction of the total traffic in the Géant network.

6 Conclusion

We introduced an adaptive algorithm which can dynamically configure IP-over-WDM networks according to current traffic demands. Our goal is to reduce the total power consumption by switching off idle line cards in the periods of low traffic, while keeping the constraints on QoS and limiting as much as possible the reconfigurations of the network.

The algorithm has been detailed and several simulations have been performed using traffic profiles measured over real networks. Moreover, a sensitivity analysis has been performed in order to choose the best values for the algorithm parameters, namely the watermarks.

Results show that it is possible to switch off several network devices, and, in the meanwhile, the algorithm is able to limit the amount of reconfigured, and dropped traffic, both before and after the reconfiguration of the network. High values of both watermarks result in high power savings. However low value of the low watermark W_L still allows to save significant amount of power, while limiting the amount of reconfigured traffic. High watermark W_H and the maximum utilization of the last logical link ψ do not influence consistently the amount of reconfigured traffic. With all the considered sets of parameters EWA manages to eliminate overload after reconfiguration. However the values of W_H and ψ in the range between 0.5 and 0.7 allowed to limit also the overload before the reconfiguration.

The proposed algorithm calculates the network configuration that is needed for a given time period. The control mechanisms and migration from the old network configuration to a new one is an open issue left for future work. We point out that since EWA considers first adding necessary lightpaths and then deleting of unnecessary ones, it is feasible to migrate from the old network configuration to a new one. However, modeling physical layer constraints such as the physical length of the lightpath, capacity of a fiber or wavelength assignment (optimally originating from a real SBN) would improve the estimations of power that can be saved with EWA.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 257740 (Network of Excellence "TREND").

References

[1] M. Webb. SMART 2020: Enabling the Low Carbon Economy in the Information Age. The Climate Group. London, June 2008.

[2] Towards Real Energy-efficient Network Design (TREND). http://www.fp7-trend.eu, 2010.

[3] J. Baliga, R. Ayre, K. Hinton, W.V. Sorin, and R.S. Tucker. Energy consumption in optical IP networks. *Journal of Lightwave Technology*, 27(13):2391–2403, July 2009.

[4] F. Idzikowski, S. Orlowski, C. Raack, H. Woesner, and A. Wolisz. Dynamic routing at different layers in IP-over-WDM networks – maximizing energy savings. *Optical Switching and Networking, Special Issue on Green Communications*, 8(3):181–200, July 2011.

[5] A. Gençata and B. Mukherjee. Virtual-topology adaptation for WDM mesh networks under dynamic traffic. *IEEE/ACM Transactions on Networking*, 11(2):236–247, 2003.

[6] R. Hülsermann, M. Gunkel, C. Meusburger, and D. A. Schupke. Cost modeling and evaluation of capital expenditures in optical multilayer networks. *Journal of Optical Networking*, 7(9):814–833, 2008.

[7] A. Ahmad, A. Bianco, E. Bonetto, D. Cuda, G. Gavilanes Castillo, and F. Neri. Power-aware logical topology design heuristics in wavelength-routing networks. In *Proc. of the ONDM, Bologna, Italy*, February 2011.

[8] P. N. Tran and U. Killat. Dynamic reconfiguration of logical topology for WDM networks under traffic changes. In *Proc. of the IEEE Network Operations and Management Symposium (NOMS), Salvador, Brazil*, pages 279–286, April 2008.

[9] Sndlib: library of test instances for survivable fixed telecommunication network design. http://sndlib.zib.de/home.action, February 2012.

[10] Cisco CRS-1 Production Brochure. http://www.cisco.com/en/US/prod/collateral/routers/ps5763/prod_brochure0900aecd800f8118.pdf, October 2008.

[11] F. Idzikowski. Power consumption of network elements in IP over WDM networks. Technical Report TKN-09-006, Technical University of Berlin, Telecommunication Networks Group, July 2009.