Measuring Minimum Transmission Resources in the Internet Access Router: A Sustainable Performance/Energy Trade-Off Approach.

Alfio Lombardo, Carla Panarello, Giovanni Schembra
DIIT - University of Catania - Italy
{lombardo,cpana,schembra}@diit.unict.it

ABSTRACT

The intuition pursued in this paper is to allow access network nodes to adapt the switching and transmission resources to the minimum transmission resources the access node has to provide for guaranteeing the maximum performances the network as a whole can provided to the users. For this purpose this paper propose an autonomous measurement mechanism allowing an access node in the Internet to align its power consumption to either the maximum capacity the user traffic requires, or the minimum available capacity in the network path between source and destination. Indeed the access node has to discover the minimum between the above capacity values, which, in fact, represents a resource allocation trade-off that allows the access node to provide the user with the maximum QoS while minimizing the energy consumption.

Categories and Subject Descriptors

C.2 [**Computer Systems Organization**]: Computer-Communication networks

1. INTRODUCTION

The steadily rising energy cost and the need to reduce the global greenhouse (such as CO_2) gas emission to protect our environment have turned energy efficiency into one of the primary technological challenges of our century. In this context, a number of studies estimate a power consumption related to ICT itself varying from 2% to 10% of the worldwide power consumption. Among the main ICT sectors, 37% of the total ICT emissions are due to the Telecom infrastructures and devices, while data centers and user terminals are responsible for the remaining part. In literature, the study of power-saving network devices has been introduced in the last years, starting from some pioneering work [1, 2], in which for the first time authors face the problem of quantifying and reducing the energy cost of the Internet. In [3, 4], the authors proposed complex algorithms to face the problem of finding the minimum amount of resources that have to be powered on to match the current traffic demand in the core network. Finally, in [6] the authors exploits the idea of exchanging energy profiles among devices to reduce the overall power consumption during routing and traffic-engineering operation.

These proposals aim at reducing the overall network power consumption by turning nodes and links off, and re-routing traffic to save energy. However, these policies can be applied to transit nodes/links only, since at the network access, nodes and links cannot be turned off to avoid disconnecting the users. Notice however that the largest fraction of power consumption of an Internet Service Provider (ISP) network is due to access nodes rather than to core nodes. Therefore, solutions to reduce energy consumption without turning nodes off are needed. The intuition is to allow network nodes to adapt the switching and transmission capacity to the current traffic demand to save energy and reduce power consumption. Indeed, as shown in [2], current nodes have a power consumption that is practically constant and independent from the actual load they face, causing a large waste of energy. Therefore, we support the adoption of both capacity and switching control mechanisms that would allow each node to meet the current traffic demand and save energy. We name this capability *Active Capacity Scaling* (ACS). Notice that, while current nodes do not offer support for capacity scaling, the technology to implement variable capacity electronic devices is readily available, as for example implemented in modern PCs and mobile devices in general. Moreover, novel devices that allow to enter different power states are expected in the near future [7].

Some initial works demonstrate that the rate-adaptation approach, for node with ACS capability, is effective in reducing power consumption. Energy aware solutions involving adaptive switches and software routers have been proposed by [8]. Authors in [9] provide similar performance figures, by presenting actual measurements on software routers that exploit energy features of PCs which can reduce the processing power during under-loaded conditions.

In the work proposed in this paper, we go a step further, by presenting a combined congestion control/rate-adaptation scheme, for Internet access node with ACS capability, which allows to achieve rate-adaptation while at the same time improving network performance. For this purpose, it is required that the access node is able to estimate both the maximum capacity the user traffic requires, and the minimum available capacity in the network path between source and destination. Indeed the access node has to discover the minimum between the above capacity values, which, in fact, represents a resource allocation trade-off that allows the access node to provide the user with the maximum QoS while minimizing the energy consumption.

Estimating the above "trade-off capacity" is not a trivial task in current Internet. This is due to the elastic nature of traffic, and of TCP in particular, so that, if capacity is reduced/increased at the bottleneck link, the instantaneous traffic offered by TCP sources would reduce/increase consequently. The intuition therefore suggests that controlling the trade-off capacity is a multi-constrained problem that must be carefully studied. A similar problem has been studied in the literature to solve the Internet congestion control problem [5]; in this case the user traffic is driven by an implicit or explicit signalling which results from an accurate monitoring and control of the queue length in the network nodes. Similarly, the authors have proposed in [12] two coupled mechanisms to estimate both the current offered traffic and the bottleneck capacity by monitoring and carefully controlling the queue length of transmission links in the access nodes, so that if the queue empties, the transmission capacity is reduced, while it is increased if queue is likely to overflow. In that paper we use the Active Windows Management (AWM) mechanism presented in [10, 11] coupled with an Energy Aware service Rate Tuner Handling (EARTH) mechanism aimed at invoking power management primitives at the hardware level to increase/decrease the node operational rate and the related performance states. In particular EARTH is not a measurement mechanism, that is a mechanism able to precisely quantify the minimum necessary resource from the observation of the queue length; EARTH is a step by step heuristic mechanism looking for the best fit capacity by increasing/decreasing the node operational rate through discrete capacity steps. In this paper we improve the EARTH mechanism with the purpose of proposing a precise measurement scenario able to detect the minimum transmission resources an access router has to provide to maintain the best performance with the minimum energy. For this purpose a new mechanism named C-EARTH (Countinuos EARTH) is proposed which precisely measure the minimum necessary resource from the observation of the queue length. C-EARTH exploits three complementary algoritms: AWM which stabilize the queue lenght despite of the elastic nature of the TCP traffic, Number of loading TCP sources estimation, RTT estimation. All the above mechanisms, including C-EARTH, forms a measurement scenario which we name through the paper "Green Router mechanism" (G-Router).

Let us note that the deployment of the G-Router requires changes at access nodes only, which is a major advantage considering the current and future Internet [10, 11]. Moreover, G-Router mechanisms do not maintain per flow state so avoiding any scalability problem with the amount of managed input traffic, and is compatible with the current routers today existing in Internet.

The paper is organized as follows. Section 2 presents the G-router behavior and describe all the algorithms used: AWM, C-EARTH, Number of loading TCP sources estimation and Round-Trip Time estimation. Then, in Section 3 simulation results are presented.

2. THE G-ROUTER

The goal of the G-router mechanism is to minimize the energy consumption while maximizing the network performance. More specifically, when the traffic load is lower than the G-router service rate, the output link capacity is under-utilized, so a reduction of the node operational rate can be applied in order to reduce the energy waste. On the contrary, when the traffic load increases and more bandwidth is needed to improve performance, an increase of the node operational rate is necessary to improve performance. The actual value of the output capacity of the TCP buffer is chosen in order to guarantee the highest throughput to TCP traffic while minimizing the waste of energy due to over-dimensioned capacities. This value is determined in the G-router by the C-EARTH algorithm. The idea at the base of this algorithm derives from the well known concept of the queueing theory that the instantaneous changing rate of the queue length is given by the difference between the arrival rate and the service rate. Therefore, when the buffer queue length is constant, the service rate equals the arrival rate. So, in order to achieve its goal, the C-EARTH algorithm monitors the output buffer queue length variations and, based on them, determines the service rate variation that should be applied in order to keep the queue length as constant as possible, and meet the arrival rate variations.

However, due to the natural burstiness of the TCP sending rate, the variation registered in the output buffer queue length between two generic time instants should be averaged along a time interval of duration equal to the round-trip time. So the first issue to face is how to estimate the round-trip time of TCP flows crossing the G-router; an additional issue is managing the different round-trip time of all the TCP flows without requiring per-flow state memory. Moreover, the well known sawtooth behavior of the TCP sending rate causes that the buffer queue length suffers of great oscillations which can trigger off frequent service rate transitions that can be lead to instability phenomenons and consequent performance degradations.

For these reasons, we propose to combine the C-EARTH algorithm with the AWM mechanism presented by the same authors in [10, 11], which stabilizes the output buffer queue length around a *target* value. Let us note that the AWM algorithm is able to keep the output buffer queue length of the node close to the *target* value, if the node is the bottleneck along the path between sources and destinations. More specifically, the buffer system can work in one of the following situations:

i)the output buffer input rate is less than the service rate, that is, the G-router is not the bottleneck node. This happens either when sources are constrained by another bottleneck which is not the considered G-Router, or applications generating the traffic require very low bandwidth. In these cases the AWM algorithm tries to increase the TCP source sending rate. However, the TCP sources are not allowed to increase their sending rate because they are driven by either the bottleneck node or the user applications. So, in this case AWM does not influence the G-Router behavior;

ii) the queue is not empty, and therefore the AWM works to maintain its length to the *target*.

The first matter allows to reveal that the service rate is higher than necessary, that is, it is higher than the minimum value of bandwidth necessary to guarantee the maximum allowed throughput without wasting energy. In the second case AWM is able to control TCP sources to achieve a constant queue length equal to the *target*. However, in this case the work of the AWM mechanism does not allow to detect occurrences in which either sources are slowly increasing their offered load or a bottleneck node in the path has increased its available bandwidth. In both cases the AWM node is forcing a bandwidth lower than necessary, that is, a higher QoS could be provided to the users. To avoid this "QoS constriction", the C-EARTH algorithm has to assure that the AWM algorithm increases the sending rate of the TCP sources until their maximum achievable throughput. Since the AWM algorithm increases the TCP sending rate if the queue length is lower than the *target* value, we have defined another target for the C-EARTH algorithm, indicated as $target_{C-EARTH}$, and defined in such a way that $target_{C-EARTH} < target$. In this way, the AWM algorithm drives the TCP sending rate in order to maintain the queue length close to the *target* value, while the C-EARTH algorithm evaluates the output capacity which maintains the queue length between the *target* and $target_{C-EARTH}$ values. Summarizing, the G-router uses the AWM algorithm as a bandwidth waste and a QoS constriction detector, while C-EARTH determines how to react to the condition revealed by AWM by evaluating the desired output capacity. Once C-EARTH has calculated the new value for the output link capacity, power management primitives at the hardware level are invoked to decrease/increase the node operational rate and the related performance states, until AWM regains the control of the TCP source sending rate, and specifically until the buffer service rate becomes the minimum between the source offered load and the forward bottleneck capacity. In other words, inside the G-router, the two algorithms, C-EARTH and AWM, work together implementing a dual control mechanism: AWM guarantees full utilization of the link capacity by driving the TCP sending rate in order to make the arrival rate equal to the service rate, while C-EARTH assures maximum energy saving by making the service rate equal to the maximum throughput achievable by the TCP flow aggregate. The AWM and C-EARTH algorithms will be described in Sections 2.1 and 2.2. Two additional issues arise: 1) the AWM algorithm needs to know the number of TCP flows crossing the G-router, and at the same time 2) the C-EARTH algorithm needs to know the average round trip time of the TCP connections. For this reason, the G-router is provided with two additional algorithms, one for the estimation of the number of flows, and the other for the estimation of the round-trip time. These two algorithms will be presented in Sections 2.3 and 2.4, respectively.

2.1 The AWM algorithm

The goal of AWM is to keep the output buffer queue length of an access bottleneck node close to the *target* value to achieve no loss, while maximizing network utilization. The AWM mechanism acts on ACK packets sent by receivers to the corresponding TCP sources. More specifically, it uses the TCP header field called *Advertised Window* (*awnd*) to control the transmission rate of the TCP sources. To avoid packet losses and maximize link utilization, the AWM algorithm estimates the number of bytes, called *Suggested Window* (*swnd*), that the G-router should receive from each TCP source in order to maintain the output buffer queue length as constant as possible and close to a prefixed *target* value, to be chosen significantly smaller than the buffer size (so as to have no losses), and at the same time higher than zero (to avoid under-utilization). Then the *swnd* value is sent to the TCP sources by using the *Advertised Window* (*awnd*) field of the TCP header in the ACK packets sent by the receivers to the corresponding TCP sources. This is done at each packet arrival and departure in the output buffer, irrespective of the TCP connection the packet belongs to. Obviously, in order not to interfere with the original TCP flow control algorithm, the AWM algorithm will overwrite the value of the *awnd* field with the calculated *swnd*, if and only if *awnd* is greater than *swnd*, whereas in the opposite case the *awnd* filed remains unchanged.

The *swnd* value at the generic updating event k, $swnd_k$, is evaluated on the basis of its previous value $swnd_{k-1}$, by considering two corrective terms DQ_k and DT_k:

$$swnd_k = \max\left(swnd_{k-1} + DQ_k + DT_k, MTU\right) \tag{1}$$

The minimum value of the *swnd* is set to the Maximum Transfer Unit (MTU) to avoid the *Silly Window Syndrome*.

The terms DQ_k and DT_k in ((1)) are defined as follows:

$$DQ_k = \left(q_{k-1} - q_k\right)/N \quad \text{and} \quad DT_k = \frac{\beta \cdot (t_k - t_{k-1})}{N \cdot T} \cdot (T - q_k) \tag{2}$$

where N is an estimate of the number of active TCP flows crossing the G-router and β is a positive parameter to be chosen to guarantee convergence to the *target*. In Section 2.3 we will present the algorithm which allows to estimate the number of flows. In [10] we have presented an analytical fluid model of the AWM algorithm and a system stability study to be used to choose the appropriate values of β to obtain the desired performance.

The term DQ_k has been defined as in ((2)) to make a positive contribution when the instantaneous queue length q_k is lower than its previous value, and a negative contribution in the opposite case. In other words, the AWM algorithm detects the bandwidth availability when the instantaneous queue length decreases, and informs the TCP sources by proportionally increasing the suggested window value for each of them. When, on the contrary, the queue length increases, the AWM algorithm infers incipient congestion and forces TCP sources to reduce their emission rates. If the N TCP sources reduce their transmission window by the term DQ_k, the queue length will go back to the q_{k-1} value. The term DT_k, on the other hand, has been introduced to stabilize the queue length around a given *target* value, T. To this end, DT_k has to make a positive contribution when the queue length is less than T, and a negative contribution in the opposite case.

The design of the AWM algorithm presented in [10] and the relative choice of β, assumes that the buffer drains at a constant rate, whereas the capacity variations, determined by the C-EARTH algorithm in the G-router, change the AWM design conditions. For this reason, some minor revisions related to the value of the β parameter are necessary. In particular, the β value to be used at each updating event t_k is calculated, on the base of its previous value, by taking into account the service rate variation applied by C-EARTH, as $\beta_k = \beta_{k-1} \cdot C_k\, C_{k-1}$, where β_{k-1}, C_{k-1}, β_k and C_k are the β value and the service rate at the time instant t_{k-1} and t_k respectively.

2.2 The C-EARTH algorithm

The goal of the C-EARTH algorithm is to determine the output capacity which equals the TCP maximum achievable throughput. In other words, it has to decrease the output capacity when the traffic load is lower than the service rate, and increase it in the opposite case. The output capacity variation needed to guarantee the highest throughput to TCP sources while minimizing the waste of energy due to over-dimensioned capacities, is determined on the base of the well known concept of the queueing theory that the instantaneous changing rate of the queue length is given by the difference between the arrival rate and the service rate. So , when the buffer queue length is constant, the service rate equals the arrival rate. Therefore, the C-EARTH algorithm determines the desired output capacity by monitoring the output queue length variations and calculating the service rate variation that should be applied to keep the queue length as constant as possible and meet the arrival rate variations. However, due to the natural burstiness of the TCP sending rate, the variation registered in the output buffer queue length between two generic time instants should be averaged along a time interval of duration equal to the round-trip time. Let t_{k-1} and t_k be two generic time instants, and r_{k-1}, q_{k-1}, r_k and q_k the arrival rate and the output buffer queue length at the time instant t_{k-1} and t_k, respectively. Therefore the arrival rate variation $\Delta r_k = r_{k-1} - r_k$ and the consequent queue length variation $\Delta q_k = q_{k-1} - q_k$ are linked together as follows:

$$\Delta r_k = \frac{\Delta q}{RTT_k} \tag{3}$$

where RTT_k is the round trip time estimated at the time instant t_k by the algorithm presented in Section 2.4.

Eq. (3) allows us to evaluate any arrival rate variation producing the queue length variation Δq_k, and therefore the service rate variation ΔC_k necessary to balance the arrival rate. More specifically, the service rate variation ΔC_k to be applied is calculated as:

$$\Delta C_k = |\Delta r_k| \cdot sgn(q_k - T_{C-EARTH}) \cdot g_k \tag{4}$$

Where $|x|$ indicates the absolute value of x, $sgn(x)$ indicates the sign function of x, $T_{C-EARTH} = \gamma \cdot T$ is the C-EARTH *target* value, T is the AWM *target* value, γ is a positive constant lower than 1, and g_k is a weight which value and meaning will be clarified in the following. Let us recall that the $T_{C-EARTH}$ value has been introduced in order to assure that the AWM algorithm increases the sending rate of the TCP sources until their maximum achievable throughput. The service rate variation ΔC_k has been defined as in (4) to increase the G-router service rate when the queue length is greater than $T_{C-EARTH}$, and decrease the G-router service rate when the queue length is lower than $T_{C-EARTH}$. Let us note that, since oscillations around the *target* value are natural and due, for example, to the granularity of the packet size, variations of the queue length are present even in the case the service rate equals the average arrival rate but no variation of the service rate is needed in this case. However, the amplitude of such oscillations is almost constant in stability conditions, that is, when the service rate equals the average arrival rate. Therefore, the variation of the service rate should be more effective when the queue length is far from the *target* and less effective when it is close to the *target*. For this reason, when the ΔC value calculated as in (4) is applied, it is weighted through the term $g_k = |q_k - T|/T$.

2.3 The estimation of the number of flows

In [10] an algorithm for the estimation of the number of TCP flows crossing the G-router has been presented. However, that algorithm was designed assuming a constant buffer service rate: if the service rate changes, this change should be taken into account. Therefore, to assure the correct behavior of the AWM algorithm, the estimation of the number of flows has been here revisited.

In order to estimate the number of flows crossing the G-router, let us recall that the AWM algorithm maintain one *swnd* value for the entire set of connections, so we can consider the output buffer is loaded by a single TCP connection with sending rate $N \cdot swnd/RTT$. Moreover, since the algorithm has been design to maintain the queue length close to the *target* value, we can assume that in the steady state the derivative of the queue length is zero and the round-trip time, given by the sum of the round-trp propagation delay and the queueing delay, is almost constant. Therefore, if we consider two consecutive updating events at the instants t_{k-1} and t_k we have $N_{k-1} \cdot swnd_{k-1}/C_{k-1} = N_k \cdot swnd_k/C_k$, where N_{k-1}, $swnd_{k-1}$, C_{k-1}, N_k, $swnd_k$ and C_k are the number of flows, the suggested window and the service rate at the time instants t_{k-1} and t_k respectively. From the above relationship, we can adaptively evaluate N_k in the following way:

$$N_k = N_{k-1} \cdot \frac{swnd_{k-1} \cdot C_k}{swnd_k \cdot C_{k-1}} \tag{5}$$

2.4 The estimation of the round-trip time

The AWM algorithm presented in 2.1, has been defined in such a way that scalability is not weakened: in order to avoid per-flow state memory, the AWM algorithm maintains one *swnd* value for each network interface and updates are applied to the *Advertised Window* field in ACKs coming from a given interface, irrespective of the particular TCP connection they belong to. The characteristic of maintaining one *swnd* value for the entire set of connections, allows us to consider the output buffer is loaded by a single TCP source with a *Transmission Window* $twnd = N \cdot swnd$, where N is the number of flows crossing the G-router. This feature allows us to express the arrival rate r_k in the G-router output buffer at the generic time instant t_k as:

$$r_k = \frac{N_k \cdot swnd_k}{RTT_k} \tag{6}$$

where N_k, $swnd_k$ and RTT_k are the estimated number of flows crossing the G-router, the AWM suggested window and the aggregate averaged round-trip time, respectively, at the time instant t_k. Since C-EARTH makes the service rate C_k follow the arrival rate r_k, (6) allows us to estimate the round-trip time of the aggregate as $RTT_k = N_k \cdot swnd_k/C_k$. Let us note that the estimation of the round-trip time RTT_k, that has been obtained as a function of the service rate C_k, cannot be used for the calculus of C_k itself in (4). However, since in the steady state AWM keeps the buffer queue length close to the *target* value, the buffer queue length and, consequently, the round-trip time do not suffer appreciable variation and we can approximate the value of RTT_k at the time instant t_k with the value of RTT_{k-1} at the time instant t_{k-1}. Therefore the value of the round-trip time to be used in (4), can be obtained as:

$$RTT_k = \frac{N_{k-1} \cdot swnd_{k-1}}{C_{k-1}} \tag{7}$$

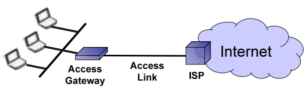

Figure 1: The simulated network topology: Access Gateway implements AWM/C-EARTH.

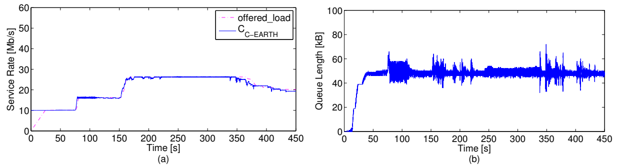

Figure 2: Case Study 1. (a) G-router output capacity determined by the C-EARTH algorithm - (b) Output Buffer Queue length in the G-router.

3. NUMERICAL RESULTS

In this section we will show that the G-router is able to minimize the energy consumption while maximizing the network performance. More specifically, we will see that the C-EARTH algorithm is able to determine the minimum bandwidth needed to assure the maximum achievable throughput, while AWM stabilizes the queue length around the *target* value.

We have conducted extensive simulations with the ns-2.30 simulator,which we integrated with AWM and C-EARTH modules. The network topology used is the one depicted in Fig. 1. The considered scenario is typical of home users accessing Internet via DSL technology and sending data to users located anywhere in the Internet. Therefore, for our simulations we consider the source down-link rate is 20Mb/s and the up-link rate is 1Mb/s. The number of source nodes is variable during each simulation and the applications running on each node can generate both FTP-like and Web-like traffic, so assuring the traffic is highly dynamic. In particular we modeled the web workload alternating requests for web files and idle times, with distribution and correlation properties of real web users. More specifically, we assume that short flows arrive according to a Poisson process with an average of 5 new web file requests per second, with Pareto-distributed file sizes with an average of 200 packets and shape 1.35. The average number of FTP-like sources is 20. The round-trip propagation delay is 200ms. The MTU length is set to 1000 bytes. Sources use unmodified TCP NewReno. Let us observe that although we consider TCP NewReno, the G-router performance are not affected by the TCP version we use; this is

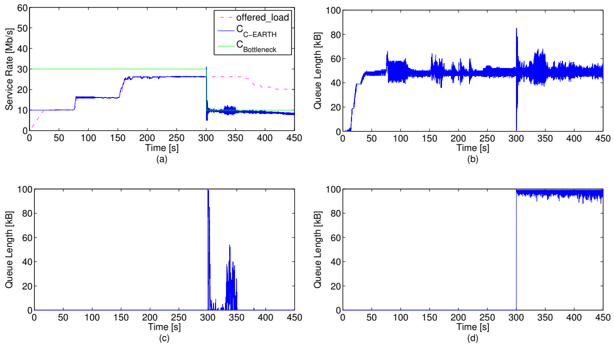

Figure 3: Case Study 2. (a) G-router output capacity determined by the C-EARTH algorithm - (b) Output Buffer Queue length in the G-router - Output Buffer Queue Length in the Internet node when the access node (c) implements the AWM/C-EARTH mechanism; (d) is Not Controlled (NC).

because AWM by-passes the TCP control mechanism, which is the key element making the difference between TCP versions. The buffer size is set to 100 packets. The β and *target* values for the AWM algorithm are set to 10kB/s and half of the buffer size, respectively. The $T_{C-EARTH}$ parameter for the C-EARTH algorithm is set to 95% of the AWM *target* value.

Let us first investigate performance of the AWM/C-EARTH mechanism in the case the offered load is lower than the network capacity (Case Study 1); for this reason both the maximum available access link capacity and the capacity of any link in Internet are initially large enough to be considered unlimited. Simulation results are shown in Fig. 2.a. The dot-dashed line shows the throughput achieved by the source aggregate in over-provision condition that is when the network capacity is higher than the offered load. The bandwidth demand is about 10Mb/s for the first 75 seconds of simulation and then rapidly increases to about 15Mb/s. This traffic load is maintained for about 75 seconds more. Then for 200 seconds it becomes about 25Mb/s and after that it decreases gradually until about 20Mb/s at the end of the simulation. The solid line in Fig. 2.a is the service rate of the G-router forced by the C-EARTH algorithm. As can be seen form the figure, the C-EARTH algorithm is able to make the service rate of the G-router follow any variation of the traffic load. Fig. 2.b shows the queue length in the access node output buffer. The AWM algorithm stabilizes the queue length around the *target* value with small oscillation, so assuring high performance. Indeed, thanks to AWM, losses caused by the saturation of the output buffer of the G-router are avoided while the utilization of the link capacity is high. Simulation results show that the average link utilization in the G-router is 98.97% whereas the throughput achieved by the aggregate of TCP sources when the G-router is 99.17% of the throughput achieved by the aggregate in over-provisioning conditions. These results demonstrate that the G-router is able to assures full utilization of the link capacity so allowing reduction of the energy waste typical of over-provisioned networks, without reducing the source performances.

Let us suppose now changes in the traffic loading a forward node occur and make that node a bottleneck along the path between sources and destinations (Case Study 2). In such a conditions, any value of the access node service rate higher than the available bandwidth on the bottleneck node results in a waste of resource. Fig. 3 shows the simulation results obtained in the case a generic node in the Internet path (called "Internet node" in the following) becomes a bottleneck after 300 seconds of simulation. In order to demonstrate that AWM/C-EARTH always determines the minimum service rate between the source offered load and the bottleneck capacity, we expect that during the first 300 seconds of simulation the service rate set by the AWM/C-EARTH algorithm follows the offered load; then for the following 300 seconds, corresponding to the presence of a bottleneck, it follows the bottleneck capacity. In Fig. 3.a the service rate determined by the AWM/C-EARTH algorithm in the access node (solid line) is compared with both the offered load (dash-dotted line) and the Internet node capacity (dashed line): it is evident that the AWM/C-EARTH mechanism captures the requirements of adapting the access node service rate as expected. Fig. 3.b shows the queue length in the access node output buffer. Again, the AWM algorithm stabilizes the queue length around the *target* value so assuring high performance. To give more insight, Fig. 3.c and Fig. 3.d show the output buffer queue length of the Internet node when the access node implements the AWM/C-EARTH algorithm (Fig. 3.c) or when it does not implement any energy-aware service rate control mechanism (Fig. 3.d). In the latter case, as soon as the Internet node available bandwidth decreases, its output buffer overflows and remains full until its service rate becomes higher than the offered load. Therefore, for all the duration of this time interval many packets are lost. On the contrary, when the access node implements AWM/C-EARTH mechanism, the Internet node output buffer remains full and causes losses for the very short period the AWM/C-EARTH mechanism needs for capacity adaptation. The above considerations are confirmed by simulation results: when AWM/C-EARTH is implemented, the number of lost packets is about 97% lower than the not controlled case.

4. ACKNOWLEDGEMENTS

This work was partially supported by the Italian Ministry of University and Research (MIUR), with the Grant PRIN-project "Sorpasso" and by the EU-funded PROVIDEO Project.

5. REFERENCES

[1] M. Gupta, S. Singh, "Greening of the Internet," *Proc. of ACM SIGCOMM*, Karlsruhe, Germany, August 2003.

[2] P. Barford, J. Chabarek, C. Estan, J. Sommers, D. Tsiang, S. Wright, "Power Awareness in Network Design and Routing," *Proc. of IEEE INFOCOM 2008*, Phoenix, USA, April 2008.

[3] L. Chiaraviglio, M. Mellia, F. Neri, "Reducing Power Consumption in Backbone Networks," *Proc. of ICC 2009*, Dresden, Germany.

[4] L. Chiaraviglio, M. Mellia, F. Neri, "Energy-Aware Backbone Networks: a Case-Study," *Proc. of GreenComm'09*, Germany.

[5] V. Jacobson, "Congestion Avoidance and Control," *Proc. ACM SIGCOMM '88*, 1988, Stanford, USA, August 1988.

[6] J. Restrepo, C. Gruber, C. Machoca, "Energy Profile Aware Routing," *Proc. of GreenComm'09*, Dresden, Germany.

[7] *Cisco Energy Wise* , http://www.cisco.com/.

[8] G. Ananthanarayanan, R. H. Katz, "Greening the Switch," *Proc. of HotPower '08*, San Diego, USA, December 2008.

[9] R. Bolla, R. Bruschi, A. Ranieri, "Green Support for PC-based Software Router: Performance Evaluation and Modeling," *Proc. of ICC 2009*, Dresden, Germany, June 2009.

[10] M. Barbera, A. Lombardo, C. Panarello, G. Schembra, "Active Window Management: an efficient gateway mechanism for TCP traffic control," *Proc. of IEEE ICC 2007*, Glasgow, UK, June 2007.

[11] M. Barbera, M. Gerla, A. Lombardo, C. Panarello, M. Y. Sanadidi, G. Schembra, "Active Window Management: Performance Assessment through an Extensive Comparison with XCP," *Proc. of IFIP Networking 2008*, Singapore, May 2008.

[12] C. Panarello, A. Lombardo, G. Schembra, L. Chiaraviglio, M. Mellia, "Energy Saving and Network Performance: a Trade-off Approach," *Proc. of e-Energy 2010*, Germany, April 13-15, 2010.