Collaborative Algorithm with a Green Touch

Luciana Oliveira, Djamel Hadj Sadok, Glauco Gonçalves,
Renato Abreu, and Judith Kelner

Federal University of Pernambuco (UFPE), Recife, Brazil
{lpo,jamel,geg,rra3,jk}@cin.ufpe.br

Abstract. Discovery and announcement processes are regular tasks in ubiquitous scenarios, since they are important to support devices' self-adaptation. However, devices enable a multitude of protocols by default, including many not always required by users or networks. For example, the NetBIOS protocol is rarely used in some networks, but it is continuously executed by several Desktops. Consequently, today's networks waste energy processing protocol messages that should be turned off. In this paper, we analyze such energy cost, and we propose devices' collaboration and sharing of traffic knowledge to save energy. Firstly, we suggest that devices should view protocols in the same way of users through a social analogy. Skype, MSN, ARP and other protocols are seen as societies with specific languages, which devices can learn and teach, similarly to societies. Finally, simulations showed that our proposal is able to reduce energy consumption for all network's devices.

Keywords: Energy efficiency and awareness, Architectures, systems and applications.

1 Introduction

The ubiquitous scenario brought new opportunities with regard to sharing services between users, providers and autonomous devices. In such environment, providers and users frequently use several protocols to announce their services, but such behavior have been increased the energy consumption of users' devices and providers' network devices. For instance, researchers from Bell Labs found that current networks use 10,000 times more energy than the absolute minimum required to transport data bits across them. For this reason, researchers have been studying solutions to save energy and how efficiently to use energy resources.

According to [1], approaches to reduce consumption span several levels from applications and operating systems to hardware. For example, work in [4] introduces changes to the Bitorrent application to reduce its energy consumption and to encourage P2P and grid networks to save energy in data centers. At the hardware level, [18] introduced optical technologies to save energy. It approached the optical switching to avoid the use of electronic forwarding and switching of IP routers that have high consumption of energy. The research in [5] and [6] are examples of efforts for finding hardware solutions to build devices that consume renewable energy.

Authors in [1] and [19] also point to the network infrastructure as the key issue to control energy consumption. The proposal in [19] identified the consumption of

P. Sénac, M. Ott, and A. Seneviratne (Eds.): MobiQuitous 2010, LNICST 73, pp. 51–62, 2012.

conventional devices in a 3G infrastructure and proposed changing the conventional devices to optical in the access network architecture. Work in [7] presents several factors that should be analyzed to reduce the energy consumption such as collisions at the MAC layer, amount of switching between sending and receiving data, communication modes, computation of transmission schedule and others.

Future networks must consider the use of energy-aware devices, applications and protocols. Such networks, one may call, as green networks [8]. Green networks may have membership policies whereby, only devices that turn off some network protocols, or those that implemented certain operation modes for energy saving, etc., would be able to join the network.

In this work, a communication device is seen as a mean for a user to join in some groups or services which we refer to as societies. Societies are distinguished by their inherent set of protocols and languages (messages). As a result of the above, a green network could define and enforce, within its policies, the sets of societies (protocols) that devices must participate (turn on) or must reject (turn off) in order to joining the network.

As these policies differ between networks, it is important that devices implement a mechanism to recognize societies' information in order to change their behavior to suit the rules of a given network before joining it. To achieve this goal, this work develops a flexible and sharable mechanism for network elements to learn and teach each other knowledge about traffic and consequently allows devices to self-configure while minimizing their energy consumption.

Formally, our approach slices the communication in societies modeled to be recognized by automata. An automaton sharing algorithm is developed for devices to learn and teach others about the identification of societies, providing a flexible and dynamic solution for acquiring knowledge by observing traffic. Using wireless trace data, we show how sharing of automata allows devices to build dynamically a green network and reduce the energy consumption.

The remainder of this paper is organized as follows. Next section introduces the relationship between the standard concepts of societies and network protocols. Section 3 introduces the proposal for a collaborative algorithm to automatically share automata between network devices in order to recognize societies and provide the functionality of management for users and networks. Section 4 presents an evaluation of these algorithms in a scenario of green networks. Finally, a discussion about related works, conclusion of this paper and future works are presented respectively in Sections 5 and 6.

2 Why and How Should We Recognize Societies?

Our work in [2] showed that relationships between machine and humans are towards socially inspired structures, embracing new networking contexts with new business models, services and technologies. We also showed that overlay networks are an interesting model to exemplify societies constituted of elements. Here, we address protocols as societies and protocol messages as the vocabulary for such societies. Thus, we see the vocabulary as a syntactic pattern to extract the society knowledge, i.e., the set of know addresses and messages within a given interval of time.

We suggest that such co-existence among vocabulary, messages and addresses can describe higher level information such as business relationships (representing customers or services of some enterprise) as well as low level or less abstract information such as network protocols (ARP, LLC, IP, NetBIOS and others). Although our proposal is towards modeling syntactically each society with its own vocabulary (pattern to express addresses and messages), this networking case study maps societies to the protocols or low level information that make them up.

We assumed that each society has a specific vocabulary or a syntactic pattern consisting of address information and message symbols. Then automata theory is a promising mathematical model to express a machine to recognize such vocabulary constituted by a set of words that identify a society. Such model is extensively used by compilers and linguistics research to describe the precise and correct vocabulary (formal grammar) of any programming language.

Conceptually, an automaton changes its current state according to read symbols. When the symbols of message are completely consumed the last state indicates if the input were accepted as part of the language. Formally, an automaton $\mathbf{s_i}$ is defined by $\mathbf{s_i} = \{Q, \sum, \delta, q0, F\}$, where Q is a set of states. $\sum$ is a finite set of symbols a.k.a. the alphabet of the language whereas we will refer to as the vocabulary of a society. δ is a function such that δ: Q x $\sum \rightarrow$ Q. q0 is the initial state, that is, the state in which the automaton is when no input has been processed yet, where $q0 \in Q$. Moreover, an automaton can have one or more states represented by F that is a set of states of Q called accept states. Each transition function of δ receives an input symbol and goes to some state or stays in the same state. For example, δ_a or $\delta(q,a)$ represents a transition in state q to q', where δ_a:Q$\rightarrow$Q and $q' \in Q$. Hence, a pair of symbols can be a unique function δ':Q$\rightarrow$Q, for example, δ_{ab}:Q$\rightarrow$Q, or denotes a composition function processed sequentially δ_a and after δ_b, or defined by $\delta_a \circ \delta_b$:Q$\rightarrow$Q.

Consequently, a society is completely known when the automaton $\mathbf{s_i}$ identifies all possible exchanged messages among its elements, i.e., when all combination of vocabulary $\sum^*$ are known. Hence, a society is identified by an automaton $\mathbf{s_i} = \{Q, \sum, \delta, q0, F\}$, when $\mathbf{s_i} = \{w \in \sum^* \mid \delta'(q0,w) \in F\}$. The messages identified by $\mathbf{s_i}$ of a given device constitute its knowledge $\mathbf{k_i}$.

Further the most intrinsic social mechanism, - knowledge sharing by learning and teaching - needs to be executed by networking devices. We suggest that sharing automata, users, service providers and any network may control the traffic.

Before giving details of our proposed algorithm for sharing automata, let us emphasize that the specific creation of automata is outside the scope of this work. We assumed that such task could be manually performed by any individual using any mechanism such as L7[1] or tcpdump/libpcap(T/L)[2] filters support to identify a society. Our focus was on the design of a generic algorithm that saves energy due to automata share. L7-filter deal regular expression that can be derived to translate it into finite automaton and T/L has a specific expression to identify societies.

These tools are able to receive an expression and translate it into a finite automaton. L7-filter is able to translate a file containing a regular expression. For example, Figure 1(a) shows an automaton used to identify the NetBIOS society.

[1] http://l7-filter.sourceforge.net/
[2] http://www.tcpdump.org/

Please note that such filter is limited to identifying societies only in the application layer. Differently, the T/L can identify any society in any layer of the OSI architecture. For comparison, Figure 1(b) shows how to do an automaton to recognize NetBIOS using T/L filter.

`\x81.?.?.[A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P][A-P]`	`eth.dst and eth.src and eth.type == 0x0800 and ip.dst and ip.src and udp.dstport == 137 and nbns.id`
(a)	(b)

Fig. 1. Recognizing NetBIOS by different filters: (a) L7-filter and (b) T/L filter

Due to the use of different mechanisms to identify societies, we generalized our solution to associate the regular expression of L7-filter, or the tcpdump filters or any other new identification mechanism for protocol/societies. This includes the representation of applications that change frequently theirs patterns and require often automata update. For example, we identified that, if a Skype user executes a login, a logout immediately after and it executes again the login after at least 15 minutes, the user machine executes the following steps:

1 Trigger NAT-PMP [20] to define the UDP port used by such protocol;
2 Send several messages using the same port configured by NAT-PMP. We identified that some UDP messages and NAT-PMP was inside ICMP messages;
3 Send DNS messages to find the IP that corresponds to the name containing skype.com and 4publishers.com;
4 Send TCP messages using IPs from Autonomous Systems (AS) associated to messages described in items 1, 2 and 3.

Using such procedure, we recognized more than 90% of Skype traffic executed in three distinct scenarios: a small network with NAT (3 machines); a medium network with firewall and NAT (40 machines); and a machine with direct Internet access. Note that such evaluation is not within the scope of this work. It was described to contextualize the creation of a regular expression to recognize Skype society as can be seen in Figure 2. However, we will need to change *tcpdump/libcap* to filter a list of AS numbers. Additionally, such list and port configured by NAT-PMP should be dynamically updated according new values found by sequence 1-4.

In this context, our central observation and also main contribution is that the use of an automata sharing algorithm can be efficient in the dynamic collaboration of devices to identify and consequently manage societies, enabling power-aware control of its resources. In the following sections the automata sharing algorithm is described.

```
(udp.srcport = port_identified_pmp or udp.srcport =
port_identified_pmp) or (list_ASN contains ip.dst or list_ASN
contains ip.src)
```

Fig. 2. Extension of tcpdump/libpcap to recognize Skype

3 Collaborative Algorithm to Identify and Acquire Knowledge

We assume that the user's device has a set of automata ($S^u = \{s_1, s_2 \ldots s_{n-1}, s_n\}$) used to to identify one or more societies. Thus, the device could then bind the semantic context of the traffic and acquire knowledge (store address information and messages) for the societies and their relationships. Such knowledge provides adequate semantic for users to select and announce routes, store data, block traffic and perform other network management tasks. As result of a lack of contextual knowledge, devices are forced to adopt the extreme behavior: block all traffic or unblock all traffic; cache all or cache nothing, enable all protocols, and so on.

In order to turn on or off such societies and avoid a lack of user control over its own device configuration, a dynamic learning mechanism is used. It is based on automata sharing used by devices to acquire $s_i \notin S^u$ and consequently to obtain new knowledge according to the social interests of each user. First, the following considerations need to be made.

User device (node) traffic can be classified according to two message types as showed in Figure 4: received messages (α) or sent message (β).

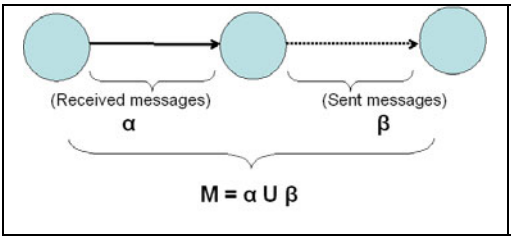

Fig. 3. Flow of user's messages

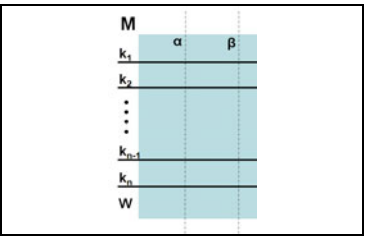

Fig. 4. Set of all user´s messages

The set of all messages received and sent by each node is expressed by M as shown in Figure 5. They are subdivided in two sets: recognized or unrecognized messages, respectively expressed by K and W. Where $K = k_1 \cap k_2 \cap \ldots k_{n-1} \cap k_n$, $|S^u| = n$, and k_i is the set of messages recognized by the used automaton $s_i \in S^u$. W is the set of the messages unrecognized to all automata $s_i \in S^u$. Also, $M = (K \cup W) = (\alpha \cup \beta)$. Additionally, we state that $k_{i,\alpha} = k_i \cap \alpha$ and $W_\alpha = W \cap \alpha$, and β-type messages follow these same conventions.

Each device or node has a knowledge base (B) which can store the following set according to user's interests: S^u (the set of all automata that device has learned) and M (the set of messages stored along the communication). Hence, $B = \langle S^u, M \rangle$, where $S^u = \emptyset$ when node has no automaton; $M = \emptyset$, when the device is configured to discard all the recognized messages or when the algorithm to share automata did not start yet.

Moreover, each device has a mechanism which frequently collects user traffic, generating a set of gathered messages in a given instant t expressed by G^t, where $G^t = \{G^t_1, G^t_2, \ldots G^t_{n-1}, G^t_n, G^t_w\}$. The gathered messages that were recognized by automaton $s_i \in S^u$ is expressed by G^t_i, and the gathered messages that failed to be recognized by any of the $s_i \in S^u$ is defined by G^t_w. Additionally we have: $G^t_{i,\alpha} = G^t_i \cap \alpha$ and $G^t_{w,\alpha} = G^t_{w,} \cap \alpha$. Again, β-type messages follow these same conventions.

Given that the input and output messages for a node are common information to all nodes in any network and that each type of information in Figures 1 and 2 provides a basic and semantic information, even when the node does not have any automata, we propose an algorithm $A_{share\_automata}$ whose behavior depends on recognized and unrecognized, received and sent messages.

By approaching these variables with social semantics, and considering that there is an equilibrium generated by cooperation, we design an algorithm $A_{share\_automata}(M, v_{collect}, v_{store}, \lambda_{learn}, \bigcup_{i}^{|S^u|}\lambda_{i,teach}, k, k_\alpha, k_\beta)$ to share automata as shown in Figure 5. Variables λ_{learn} and $\lambda_{i,teach}$, respectively, express numerically the node's motivation to learn automata and their capability to teach a specific automaton s_i; M is the set of messages already sent and or received that may contain nothing when the algorithm is started; the parameter $v_{collect}$ is a Boolean configuring the node to collect or not messages; the parameter v_{store} configures the node to store or not gathered messages into M and the other parameters (k, k_α, k_β) are values set according to user interests. They establish when a node must learn and/or teach by setting the rate of knowledge, of messages types α and β for each given automaton $s_i \in S^u$.

The algorithm to share automata is subdivided in four steps as seen in Figure 6. The algorithm $A_{collect\_data}(v_{collect})$ is the simplest one, it collects data according to the Boolean variable $v_{collect}$. The algorithm $A_{store}(v_{store}, M, c1, c2)$ is also simple, it stores the gathered messages from G^t into M according to Boolean the flag v_{store} once the algorithms A_{learn} and A_{teach} are finalized (c1 = true and c2 = true).

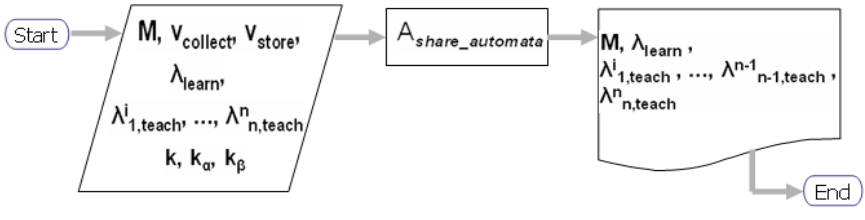

Fig. 5. Flowchart of algorithm to share automata

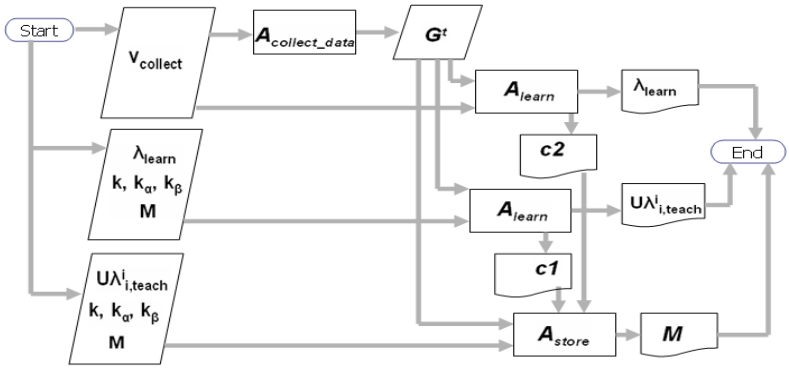

Fig. 6. Flowchart of entire algorithm to share automata

The algorithms A_{learn} (M, λ_{learn}, G^t, k, k_α, k_β) and A_{teach} (M, $\bigcup_{i}^{|S^u|}\lambda_{i,teach}$, G^t, k, k_α, k_β) are the head processes for a node to share an automaton. These were designed following the pseudo-code given in Figures 7 and 8 to increase or decrease the node's capability to learn and teach once the process of message gathering has finalized. We set the learning process to increase the node's motivation to learn (λ_{learn}) when:

1 The node has a low rate of recognized messages;
2 The node receives a high rate of unrecognized messages from the node or to the node;

Moreover, we also encourage that a node increases its capability to teach ($\lambda_{i,teach}$) the automaton s_i to other nodes, when:

1 The node has high rate of recognized messages;
2 The node frequently recognized received and sent messages;

A_{learn} (M, λ_{learn}, G^t, k, k_α, k_β)

if ($\lambda_{learn}=\infty^+$ or $\lambda_{learn}=\infty^-$)
then $\lambda_{learn}=1$
if (|K|=0 or (|K|≠0 and $\|K \cap G^t\|/G^t < k$))
then $\lambda_{learn}=\lambda_{learn}+1$
if ($\|W_\alpha \cap G^t_{w,\alpha}\|/\|G^t_{w,\alpha}\| > (1- k_\alpha)$)
then $\lambda_{learn} = \lambda_{learn} + 1$
if ($\|W_\beta \cap G^t_{w,\beta}\|/\|G^t_{w,\beta}\| > (1 - k_\beta)$)
then $\lambda_{learn} = \lambda_{learn} + 1$
$\lambda_{learn} = \lambda_{learn} - 1$

Fig. 7. Algorithm A_{learn}

A_{teach} (M, $\bigcup_{i}^{\|S^u\|}\lambda_{i,teach}$, G^t, k, k_α, k_β)

if ($\lambda_{i,teach}=\infty^+$ or $\lambda_{i,teach}=\infty^-$)
then $\lambda_{i,teach}=1$
if ($\|K_i \cap G^t_i\| / \|G^t_i\| > k$)
then $\lambda_{i,teach} = \lambda_{i,teach} + 1$
if ($\|K_\alpha \cap G^t_{i,\alpha}\|/\|G^t_{i,\alpha}\| > k_\alpha$)
then $\lambda_{i,teach} = \lambda_{i,teach} +1$
if ($\|K_\beta \cap G^t_{i,\beta}\|/\|G^t_{i,\beta}\| > k_\beta$)
then $\lambda_{i,teach} = \lambda_{i,teach} +$
$\lambda_{i,teach} = \lambda_{i,teach} - 1$

Fig. 8. Algorithm A_{teach}

When theses algorithms finalize, the result is a tuple constituted by the node's capability to learn and teach each automaton of node. When $\lambda_{learn} \in Z^+$, the device will accept to request and receive any automaton, otherwise the device will not accept any, neither could it request any. In same way, when $\lambda_{i,teach} \in Z^+$ then the device will agree to announce and send the automaton to nodes, otherwise the device will not announce it, neither would it send this to another node. However, two nodes n_A and n_B only share a automaton s_i, when λ_{learn} from n_A and $\lambda_{i,teach}$ from n_B are positive values or when λ_{learn} from n_B and $\lambda_{i,teach}$ from n_A are positive values.

4 Evaluation

The objective of this evaluation is to demonstrate qualitatively the benefit of using the proposed algorithm for automata sharing in a scenario where policies may be defined to reduce the energy consumption through the rejection of societies with high levels of energy consumption at their network devices.

The scenario was evaluated in a local area condominium network made up of four buildings and a total of 288 apartments. Along seven hours, we captured packets exchanged among the network devices using the wireshark[3] tool. We identified 418 devices and analyzed the traffic in this environment in terms of dynamicity, number of messages and devices in order to measure the energy consumption in Section 4.1.

For running our simulations, we assumed that the condominium's devices could belong to the same network, where equipments may collaborate and share automata and dynamically turn off some protocols to save energy as detailed in Section 3.

Unlike some existing energy saving approaches that rely on new hardware or its replacing by optical technologies, the present proposal adopts a simple automata based technique. Although there are security and trust issues with the exchange of automata, we leave these outside the scope of the current work. The proposed scheme is scalable as it does not require wire-speed processing as captured packets may be processed subsequently. The algorithm may run continuously or periodically depending on the amount of traffic passing through and the dynamicity of the societies. In a deployment scenario, automata sharing may be policy based as in [9], a proposal to negotiate policies considering a social approach. For simplification, we considered that policies to selectively turn off the recognized societies after the learning and teaching processing, would be enforced by all devices to reduce their energy consumption.

4.1 Analysis of Real Data from Wireless Network

Initially the wireless interface of a fixed notebook was set up in promiscuous mode to capture packets and to identify societies (protocols) and their traffic characteristics including packets size, number of received and sent messages seen here as are important parameters to measure the energy consumption. According to [13], we use the four linear models (a), (b), (c), and (d) to measure energy consumption in terms of broadcast and direct (peer-to-peer – p2p) communication, where size corresponds to the packet size:

$$E_{broadcast\text{-}send} = 1.9 \times size + 266 \quad (a)$$

$$E_{broadcast\text{-}recv} = 0.5 \times size + 56 \quad (b)$$

$$E_{p2p\text{-}send} = 1.9 \times size + 454 \quad (c)$$

$$E_{p2p\text{-}recv} = 0.5 \times size + 356 \quad (d)$$

Packet capture was terminated after seven hours. The collected traffic of three main protocols will be used in our case study. NetBIOS was chosen, because it is very rarely used by a network to support file and printer sharing and it is very insecure [10]. The ARP also was selected for turning off, because devices could use static instead of dynamic ARP in order to protect the network against ARP spoof attacks [11]. Broadcast of ICMP messages is another society selected to turn off, to mitigate ICMP Smurf attacks which can create a Denial of Service in one or more machines in the network. This decision is unlikely to provoke a problem to the network, because it is a setup already used by some operational systems. For instance, a device running Microsoft's Windows-XP with Service Pack2 security software is designed to drop ICMP packets by default [12].

[3] http://www.wireshark.org/

The NetBIOS Name Service (NBNS) and the NetBIOS Session Service (NBSS) protocols of NetBIOS were considered as two different societies in addition to ARP and ICMP, hence reaching a total of four societies. The *libpcap* filter was used to define the automata correspondent to each societies. The ARP society was identified by expression "eth.dst and eth.src and eth.type == 0x0806". The ICMP society was recognized by "eth.dst and eth.src and eth.type == 0x0800 and ip.dst and ip.src and ip.proto == 0x01". The NBNS society was identified by "eth.dst and eth.src and eth.type == 0x0800 and ip.dst and ip.src and udp.dstport == 137 and and nbns.id". The NBSS society was recognized by expression "eth.dst and eth.src and eth.type == 0x0800 and ip.dst and ip.src and udp.dstport == 137 and and nbss.type".

The total energy consumption by these protocols corresponded to 7% of all energy consumed by network along these seven hours. Figure 9(a) details consumption for each society, where the total energy consumption of NBSS was very low and for such reason it is not seen in the graph.

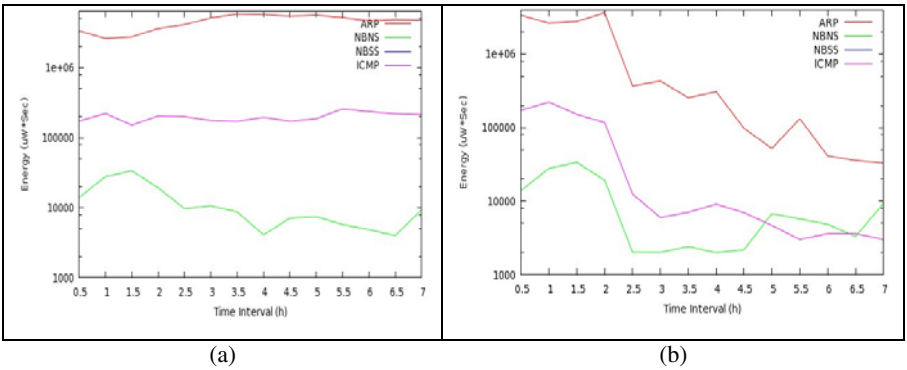

Fig. 9. Energy consumption by protocol: (a) without using automata sharing algorithm; (b) using automata sharing algorithm

4.2 Simulation

We defined a packet as a message in the form <src_address,dst_address,prot>, where src_address and dst_address are the source and destination MAC addresses, and the "prot" field is the protocol/society which can be one of ARP, NBNS, NBSS, or ICMP. Please note that this message structure was chosen for simplicity and that each society must define its message representation.

For this trace-based simulation, we assumed that all devices in the network ran the $A_{share\_automata}$ algorithm with the same parameters k=0.5, k_α=0.5 and k_β=0.5. The set M is initialized without messages. Also, the sampling time of the algorithm was set to 30 minutes. Thus, at each time t the set G^t of each device is populated with messages sent/received by the node, which feed the algorithms A_{learn} and A_{teach} that control the parameters λ_{learn}, $\lambda_{ARP,teach}$, $\lambda_{NBNS,teach}$, $\lambda_{NBSS,teach}$, and $\lambda_{ICMP,teach}$. The initial 196 devices start the simulation with all automata and through the lambda parameters these devices may learn and teach automata to new devices. After learning a new automaton, the device turn off broadcasts messages for such automaton.

Figure 9(b) presents the behavior of energy consumption when the automata sharing algorithm was enabled. Although the energy consumption of society NBSS has not been showed, it uses the same energy than in the previous experiment (5,749.6 uW*sec). That occurred because the number of similar messages for each interval of 30 minutes was not enough for the algorithm increase λ_{learn} and $\lambda_{NBSS,teach}$.

The total energy consumption of the condominium reached 15023237.75 uW*sec. It corresponds to a reduction of 77.58% (51990376.52 uW*sec) without changing the conventional devices with new equipments based on renewable energy or other technologies. So, our approach could be seen as an intermediary solution before investing into new equipments.

5 Related Work

Authors in [14] used the Turing Machine as fundamental model to understand and reduce computation time and identified the similarities between research about how to reduce energy consumption and computation time in a device. However, they did not find any theoretical models to compute the energy complexity and for such reason propose to augment the Turing Machine (a variation of Automata Theory) to determine the energy consumption of algorithms.

Similarly to this approach, we proposed a mechanism also based on theoretical model to reduce energy. However, we did not propose to modify any aspect of Automata Theory, we used it as the mechanism to recognize societies in order to turn off some and consequently to reduce energy consumption. The principal contribution was a mechanism to learn, teach, and the recognition of societies by automata sharing.

Work in [17] dynamically adapts the network based on the exploitation of the resources for power management. For such approach, some policies were presented to configure and manage the operating frequency and voltage of devices maintaining a certain performance. Our proposal focuses on application level, providing a mechanism for users to dynamically detect and control network resources. So, our proposal is a more flexible and extensible algorithm than [3] and [15] that are likely to need additional mechanisms to support dynamic policies. The research in [3] studied the traffic of a cellular network and proposed to temporarily inactivate some cells when the traffic is not high. The work in [15] proposed a specific algorithm to turn off some devices, links, protocols and application of the network.

Our approach is believed to be closer to that in [16], given that both allow hosts to participate directly in network management and traffic engineering. Moreover, the automata sharing and [16] provide mechanisms for devices to distinguish among network applications, deal differently with the traffic of each one independently of port-based filters at routers. The difference to [16] is mainly in the mechanism adopted. The approach in [16] is based on exception handlers - the network administrator defines the combination of condition and actions, while our proposal is based on sharing automata with an explicit cooperation and synchronization between hosts. The first proposal needs predefined static policies (action and conditions) and our proposal is to slice the communication in societies which can be recognized by automata to detect societies' messages (protocol). Instead of having predefined automata, these can be shared and dynamically built by devices that learn and teach these to others including the network management space.

Although [16] and our proposal were evaluated in different scenarios, we believe that our proposal could be evaluated to control the network resources in term of firewall services, load-balance and link failures that are scenarios studied by [16].

6 Conclusions and Future Works

In this paper, we suggest that hosts should be able to control their own traffic in order to accept flexible policies and consequently collaborate with other nodes in the network. Devices were shown to be able to dynamically identify and turn off some protocols in order to save energy's device as well as to reduce the energy consumption of all network.

Core to the proposal are three points: the society's analogy to provide a flexible description of policies, the handling by automata to identify societies and automata sharing algorithm by devices that are able to dynamically obtain knowledge about the network traffic. However, the main objective was to demonstrate the handling by automata sharing in the context of energy saving.

Future work includes the building of automata and applying of policies in terms of high level information (i.e. business model information). The policies should reflect future green SLAs (service level agreement in term of energy) in order to reduce energy and its cost based on learning and teaching of societies.

Other studies and applications of the present solution are planned. Automata sharing will be evaluated in terms of load-balancing work, link failure recovery and new routing algorithms in disconnected networks. Future network devices, making use of our mechanism, will dynamically coordinate themselves to take rapidly decisions such as: store data (as a proxy) or forward data (as routing algorithms) or block traffic (as a firewall) according to its policies.

This work has shown that there is space for intermediary simple solutions that, when introduced into current networks, may offer important energy cost savings without the need for immediate hardware renewal. While our approach does not require wire-speed traffic analysis it suffers from security and trust problems which have been left out of this initial study.

Although the results are promising there is urgent need to conduct further experiments to determine how best the adopted parameters k, k_α and k_β can be tuned. Moreover, we intend to design the control messages of automata sharing algorithm and add such information to be executed and account by our simulation in order to measure the energy consumption of our proposal mechanism to save energy.

References

1. Baldi, M., Ofek, Y.: Time for a "Greener" Internet. In: Proceedings of GreenComm 2009 (2009)
2. Sadok, D., Oliveira, L., Kelner, J.: New Routing Paradigms for the Next Internet. IFIP AICT, vol. 1, pp. 190–201 (2010)
3. Ajmone Marsan, M., Chiaraviglio, L., Ciullo, D., Meo, M.: Optimal Energy Saving in Cellular Access Networks. In: Proceedings of the GreenComm 2009 (2009)

4. Blackburn, J., Christensen, K.: A Simulation Study of a New Green BitTorrent. In: Proceedings of the GreenComm 2009 (2009)
5. Hande, A., Polka, T., Walkera, W., Bhatia, D.: Indoor solar energy harvesting for sensor network router nodes. The Journal of Microprocessors and Microsystems (2006)
6. Chiu, H.-J., Yao, C.-J., Lo, Y.-K.: A DC-DC converter topology for renewable energy systems. The International Journal of Circuit Theory and Applications 37(3), 485–495 (2009)
7. Jones, C.E., Silvalingam, K.M., Agravawal, P., Chen, J.C.: Survey of Energy Efficient Network Protocols for Wireless Networks. Wireless Networks 7(4) (July 2001)
8. The Climate Group Smart (2020), Report: http://www.theclimategroup.org/assets/resources/publications/Smart2020Report_lo_res.pdf
9. Molina, B., Pileggi, S.F., Esteve, M., Palau, C.E.: A social framework for content distribution in mobile transient networks. Journal of Network and Computer Applications 31(5), 1000–1011 (2009)
10. Wool, A.: A quantitative study of firewall configuration errors, vol. 37(6), pp. 62–67. IEEE Computer Society, Los Alamitos (2004)
11. Chomsiri, T.: Architecture and Protocols for Secure LAN. IJCSNS International Journal of Computer Science and Network Security 8(7) (July 2008)
12. Kumar, S., Azad, M., Gomez, O., Valdez, R.: Can Microsoft's Service Pack2 (SP2) Security Software Prevent SMURF Attacks? In: International Conference on Internet and Web Applications and Services/Advanced International Conference, pp. 89–89 (2006)
13. Feeney, L.M., Nilsson, M.: Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In: Proc. IEEE Conf on Computer Communications, Infocom 2001 (2001)
14. Javin, R., Molnar, D., Ramzan, Z.: Towards a Model of Energy Complexity for Algorithms. In: Wireless Communications and Networking Conference. IEEE, Los Alamitos (2005)
15. Chiaraviglio, L., Mellia, M., Neri, F.: Energy-aware Backbone Networks: a Case Study. In: Proceedings of the GreenComm 2009 (2009)
16. Karagiannis, T., Mortier, R., Rowstron, A.: Network Exception Handlers: Host-network Control in Enterprise Networks. SIGCOMM Computer Communication (2008)
17. Bolla, R., Bruschi, R., Davoli, F., Ranieri, A.: Energy-aware performance optimization for next-generation green network equipment. In: Proc. of Workshop on Programmable Routers for Extensible Services of Tomorrow, pp. 49–54 (2009)
18. Harai, H.: Optical Packet & Path Integration for Energy Savings toward New Generation Network. In: Proc. of the International Symposium on Applications and the Internet (2008)
19. Etoh, M., Ohya, T., Nakayama, Y.: Energy Consumption Issues on Mobile Network Systems. In: Proc. of the International Symposium on Applications and the Internet (2008)
20. Cheshire, S., Krochmal, M., Sekar, K. N.: Port Mapping Protocol (NAT-PMP), http://tools.ietf.org/html/draft-cheshire-nat-pmp-03