Green Support for PC-based Software Router: Performance Evaluation and Modeling

Raffaele Bolla, Roberto Bruschi, Andrea Ranieri
Department of Communication, Computer and Systems Science (DIST)
University of Genoa
Via all'Opera Pia 13, 16145, Genoa, Italy
{raffaele.bolla, roberto.bruschi, andrea.ranieri}@unige.it

***Abstract*—We consider a new generation of COTS Software Routers (SRs), able to effectively exploit multi-Core/CPU HW platforms. Our main objective is to evaluate and to model the impact of power saving mechanisms, generally included in today's COTS processors, on the SR networking performance and behavior. To this purpose, we separately characterized the roles of both HW and SW layers through a large set of internal and external experimental measurements, obtained with a heterogeneous set of HW platforms and SR setups. Starting from this detailed measure analysis, we propose a simple model, able to represent the SR performance with a high accuracy level in terms of packet throughput and related power consumption. The proposed model can be effectively applied inside "green optimization" mechanisms in order to minimize power consumption, while maintaining a certain SR performance target.**

Keywords-sowtware routers, green networking, performance measurement.

I. INTRODUCTION

In the last few years, the research field of "green" and energy-efficient networking infrastructures has gained a great interest from both service/network providers and equipment manufacturers. Despite a more widespread sensitivity to ecological issues, such interest springs from heavy and critical economical needs, since both energy cost and network electrical requirements show a continuous growing with an alarming trend over the past years. For example, as shown in [5] and in [6], energy consumption of Telecom Italia network in 2006 has reached more than 2TWh (about the 1% of the total Italian energy demand), increasing by a 7.95% with respect to the 2005, and by a 12.08% to the 2004. Similar trends can be generalized to a large part of the other telecoms and service providers, since they essentially depend on data traffic volume increase and new services being offered.

To support new generation network infrastructures and related services for a rapidly increasing customer population, telecoms and service providers need for a larger and larger number of equipments, with sophisticated architectures able to scalably perform more and more complex operations. In such an environment, the challenge for network equipment manufacturers is, nowadays, to design new architectures able to scale their performance and functionalities preserving a high level of energy efficiency. Despite of some interesting scientific contributions (e.g., [1], [2], [3] and [4] among the others), green networking performance and optimization remain an open and very interesting issue.

Starting from these considerations, we focus on promising new generation architectures for Linux Software Routers (SR) [7][8], based on open source SW and on multi-core COTS hardware. In detail, our work tries to investigate the impact of power management mechanisms on the SR performance. These mechanisms are made available in a large set of general purpose CPUs with the Advanced Configuration and Power Interface (ACPI) technology. Our main objective is to evaluate and to model the tradeoff between power consumption and performance of SR basic functionalities (e.g., packet forwarding), in order to assess the feasibility of "green optimizations" for COTS SR architectures. With this main aim, we performed several benchmarking sessions with a heterogeneous set of HW platforms, in order to carefully characterize the impact of power management mechanisms on SR performance. These measures gave us the chance both understand the relationship between SR internal dynamics and its external/power-related performance, and to derive a simple model to estimate SR behavior with respect to different setups and traffic offered loads. The proposed model could have a key role, thanks to its accuracy and low computational complexity, in developing innovative "green" optimization mechanisms for multi-Core SR.

The paper is organized as follows. Sections II and III introduce the new generation SR architectures, specifically optimized for multi-core HW systems, and the SR standard support for power management, respectively. In Section IV, we propose a large set of benchmarking sessions with the aim of investigating how power management mechanisms impact on SR performance. A simple empirical model, able to effectively estimate SR behavior in terms of throughput and power consumption, is proposed in Section V. Finally, conclusions and future work objectives are exposed in SectionVI.

II. NEW GENERATION ARCHITECTURES FOR LINUX SR

With the aim of evaluating capabilities and performance levels provided by a SR architecture, we decided to use a set of HW and SW enhancements that allow us to effectively and scalably exploit multi-CPU/core PC systems. In detail, these HW and SW enhancements (as already discussed and evaluated in [7] and in [8]), let us go beyond typical performance issues, namely data accessing serialization (i.e., LLTX lock contention) and CPU/core cache coherence, which generally affect SR data-plane operations. The key element of such SR architectural enhancements is the support for "multi-queuing" network adapters, like the Intel PRO 1000 adapters (with

82571 and higher chipsets), which support multiple Tx/Rx Rings and multiple HW interrupts per network interface [9].

Using such innovative network board features, we deployed a new SW architecture/configuration that allows us to optimize SR data-plane performance by reducing memory sharing among CPU/cores and avoiding the LLTX lock contention. In particular, the optimized SR architecture provides three main SW enhancement guidelines: (I) to entirely bind all operations carried out in forwarding a packet to a single CPU (minimizing cache invalidations); (II) to separate working memory regions of different CPU/cores; (III) to equally distribute the computational load among all the processors/cores activated in the system. Further details of such optimized SR architecture can be found in [8] and in [7].

III. The Power Management in Linux SR

Power management is a key feature in today's processors across all market segments. While ACPI provides a well-known standardized interface between the hardware and the software layers, processors use different internal techniques in order to reduce their energy consumption by exploiting the basic idea that systems do not need to run at peak performance all the time. This is usually accomplished by tuning the frequency and/or the voltage of processors, or by throttling the CPU clock (i.e., the clock signal is gated or disabled for some number of cycles at regular intervals). Decreasing the operating frequency and the voltage of a processor or throttling its clock, obviously allows to reduce power consumption and heat production at the price of slower performance.

ACPI technology introduces two main different power saving mechanisms, namely performance and power states (*P* and *C* states), which can be individually employed and tuned for each core in the largest part of today's CPU. Regarding the *C* states, the C_0 power state is an active power state where the CPU executes instructions, while the C_1 through C_n power states are processor sleeping or idle states, where the processor consumes less power and produces less heat.

While in the C_0 state, ACPI allows the performance of the Core to be tuned through *P* state transitions. *P* states allows to modify the operating energy point of a processor/core by altering the working frequency and/or voltage, or throttling its clock. Thus, using *P* states, a core can consume different amounts of power while providing different performance at the C_0 (running) state. At a given *P* state, a Core can transit to higher *C* states in idle conditions. In general, higher the index of *P* and *C* states is, the lesser will be power consumed, and heat dissipated. Each processor model generally provides a fixed number of *C* and *P* states for all included Cores.

IV. Performance Scaling impact on Data Plane

Our main objective in this Section is to evaluate how CPU Performance Scaling Mechanisms (PSMs) impact on SR functionalities and power consumption. With this respect, we have to focus both on HW support and capabilities of PSMs included in different CPU versions, and on how these HW features can affect performance and behavior of main SW functionalities of a SR. Power consumption is clearly dependent both on the efficiency of *P* and *C* states available in the CPU/Core, and on its instantaneous computational load. In fact, the latter directly influences the average time spent in the "active state" C_0, or in idle states (C_i, with i>1). In detail, we can express the average power consumption $\tilde{\Phi}$ of a CPU/core as in the following:

$$\tilde{\Phi} = p_{idle}\Phi_{idle} + (1 - p_{idle})\Phi_{active} \quad (1)$$

where p_{idle} is the probability that the CPU/core is in an idle state, Φ_{idle} is the average power consumption during idle states, and Φ_{active} the one during active states. Both Φ_{idle} and Φ_{active} are respectively functions of the *C* and *P* states, available at the HW layer and activated by the ACPI SW drivers.

Today's ACPI SW drivers manage the CPU power profile through a set of simplified high-level APIs, which allow users to apply a range of "equivalent CPU frequencies". Inside the ACPI driver, each equivalent frequency is mapped into a pre-selected set of feasible *C* and *P* states: as the value of equivalent frequency selected is higher, as the index of feasible *P* states is larger and the *C* states' one lower. This obviously results in a spread of CPU working states with different levels of performance and power saving. In particular, we can re-define Eq. 1 with respect to the selected equivalent frequency:

$$\tilde{\Phi}(f) = p_{idle}\Phi_{idle}(f) + (1 - p_{idle})\Phi_{active}(f), \quad f \in F \quad (2)$$

where *f* is the selected equivalent frequency, and *F* the set of all its possible values.

Note that, while Φ_{idle} and Φ_{active} are fixed by the ACPI HW layer, p_{idle} depends both on the computational load of SW tasks and on the CPU/core computational capacity, which is also a function of the equivalent frequency. Starting from these considerations, we can express p_{idle} as follows:

$$p_{idle}(f, l) = 1 - \frac{l}{C(f)} \quad (3)$$

where l is the SW computational load, and $C(f)$ is the CPU/core computational capacity at the equivalent frequency *f*.

Thus, to analyze PSM performance and its effect on SR architecture, we need to act and to focus on two different layers, by evaluating, on one hand, the ACPI HW capabilities included in different CPU/Cores, and, on the other one, the computational complexity and SW dynamics of a SR.

A. The HW layer

Regarding PSM HW capabilities, we decided to evaluate the performance provided by a heterogeneous set of PC platforms, based on different CPU models and ranging from high-end servers to common workstations and mobile architectures. For each selected platform, we performed two types of power consumption measurements: Φ_{idle} - *Idle measures*: in order to evaluate the average power consumption when all the Cores in the system are idle, and do not perform any tasks; Φ_{active}- P_n *state measures*: to quantify the average power consumption when one or more Cores are running (i.e., performing SW operations for the 100% of their time) at a fixed P_n state.

While the first kind of measurement is useful to evaluate the minimum power consumption for each selected platform, the Φ_{active} measures point out how much energy a CPU/core can absorb when fully active at a certain ACPI P_n state. In this respect, these two measures can be thought, respectively, as the upper and the lower bounds of SR performance when deployed in a real network.

We also decided to consider an additional performance index, useful to evaluate the Core's computational capacity $C(f)$ at the given frequency. In detail, since data-plane functionalities are certainly the most CPU intensive and time consuming operations to be carried out in a SR, we decided to consider the maximum throughput, in terms of forwarded packets per second, as the key performance index in order to evaluate CPU computational capacity.

Figures 1 and 2 show the Φ_{active} and the Φ_{idle} results obtained with a heterogeneous set of HW architectures, which includes high-end servers, workstations and mobile architectures. In detail, we extensively tested seven HW platforms with the following characteristics: *HW platform 1 (X)*: dual CPU Intel XEON Dual Core 5050 "Dempsey", 3.0 GHz, 64 bit, RAM 533 MHz, PCIe 16x; *HW platform 2 (PD)*: Intel Pentium-D 925 Dual Core, 3.0 GHz, RAM 667 MHz, bus PCIe 8x; *HW platform 3 (A)*: AMD Athlon 64 dual Core X2 3800+, 2.0 GHz, RAM 667MHz, PCIe 16x; *HW platform 4 (PM)*: Intel Pentium-M 745 "Dothan", single Core 1.8 GHz, RAM 266 MHz, PCIe 8x; *HW platform 5 (O)*: dual CPU AMD Opteron Dual Core 265, 1.8 GHz, RAM 400 MHz, PCIe 16x; *HW platform 6 (C2)*: Intel Core 2 Duo E2160, 1.8 GHz, RAM 667 MHz, PCIe 8x; *HW platform 7 (A2)*: AMD Athlon 64 3500+, 2.2 GHz, RAM 200 MHz, PCIe 16x.

Figures 1 and 2 highlight the gap, in terms of power consumption and forwarding performance, between the different HW platforms. These differences obviously depend not only on the specific CPU model, but also on the other HW components (e.g., mainboards, etc.). For the sake of simplicity, we drop this additional power consumption contribution in the following of this paper, since it does not affect the obtained results and conclusions. High-end server architectures (i.e., X and O) obviously show both a heavy power consumption (more than 150 Wh), and a high forwarding performance, especially when a large number of Cores is employed.

On the contrary, workstation architectures (PD, A and A2) exhibit an intermediate performance level, but both power consumption and forwarding performance can heavily change, according to the selected HW platform. For example, the PD platform is characterized by a much higher power consumption than the A and the A2 ones, while the A platform, thanks to its high-speed internal busses, shows remarkable values of packet throughput (about 1.5 Mpkt/s). In particular, the forwarding performance of the PD and the A2 platforms are limited by the bandwidth and the efficiency of internal busses, which do not allow to exploit the full computational capacity of CPUs/Cores. Similar remarks can be made also for the mobile architectures (i.e., PM and C2). In detail, we can note how the C2 HW platform provides a very low power consumption, which ranges between about 30 and 50 Wh.

Besides interesting information on different HW platforms' behavior, Figures 1 and 2 show that all the analyzed PSMs (both in AMD and Intel processors) allow to linearly scale the maximum forwarding rates with respect to power consumption. This linear behavior is maintained not only for different equivalent frequency values, when a single core is used for the SR data plane, but also for a rising number of CPUs/Cores.

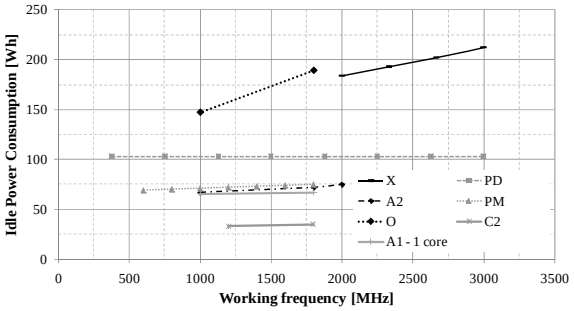

Figure 1. Idle power consumption for all the selected HW platforms.

B. The SW layer contribution

Besides the effectiveness of the PSM included in today's CPUs, we have also to take into account the impact of SW functionalities and dynamics on the HW behavior, and especially on the p_{idle} parameter as suggested by Eq. 3. With this aim, we performed several benchmarking sessions in order to empirically evaluate the relationship between the PSMs and the SR SW layer. As previously outlined, packet forwarding operations are almost the totality of computational load in a SR, since control plane functionalities (i.e., routing protocols, etc.) work at much slower time scales, and their computational weight is quite negligible. Therefore, in order to characterize SW layer behavior, we decided to use a set of performance indexes typically related to the SR data-plane.

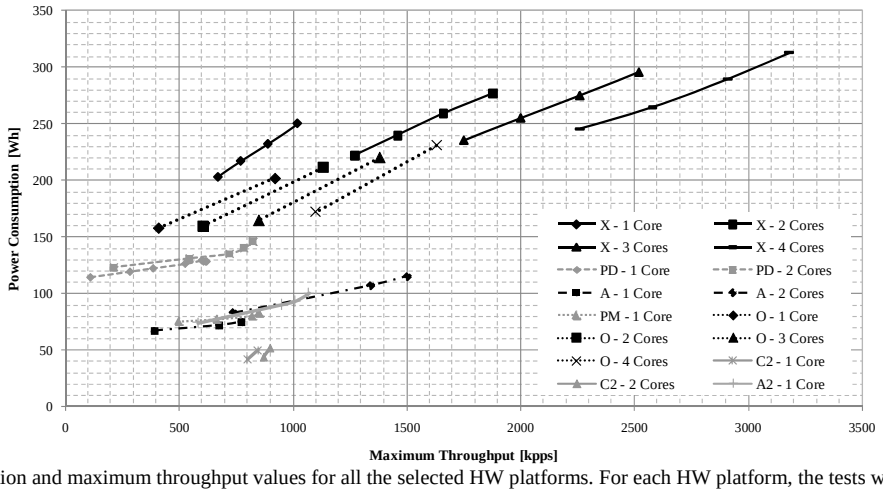

Figure 2. Power consumption and maximum throughput values for all the selected HW platforms. For each HW platform, the tests were performed by changing the CPU/Core *P* state, and with a variable number of active CPUs/Cores.

Starting from these considerations, we analyzed and evaluated the relationship between power consumption and Linux data-plane performance, in terms of throughput and packet forwarding latency. Among all the performed test sessions, Figures 3, 4 and 5 report the results related to the Xeon-based platform. Similar results were obtained for all the other analyzed HW architectures. In detail, these tests were carried out by using only one single Xeon Core to forward a traffic flow, with a constant bit rate and composed of 64B sized IP packets. Figures 3 and 4 show that data-plane performance increases with respect to the operating frequency of the working Core: higher frequency values assure larger maximum throughput values and lower latencies. This obviously depends on the increase of the Core computational capacity $C(f)$, which allows to elaborate a larger number packet headers with smaller processing times. In detail, the maximum latency times, reported in Figure 4 and obtained in identical router configurations (e.g., same buffer sizing) except for the Core frequency, highlight how the processing times of packet headers rise according to the Core frequency (and then to the $C(f)$ parameter) decrease.

However, the forwarding performance gain in terms of maximum throughput and latency, due to the CPU frequency increase, is not exactly linear. In fact, by examining in the detail Figures 3 and 4, we can note how this performance gain, slightly decreases as the Core frequency raises. This effect can be observed in a large number of PC HW architectures, and it is mainly due to a tiny performance decay of other HW elements (i.e., RAM, busses, etc.) at high-speed.

Figure 5 reports the power consumption of the Xeon-based architecture according to different values of Core frequency, and throughput. Note that the throughput values, for each test session, were obtained according to a rising traffic offered load. Thus, as clearly shown in Figure 5, the correlation between power consumption and packet throughput is not linear, all the same it highlights a similar profile according to different frequency values. The cause of such profile similarity, kept both for different frequency values and for HW platforms, can certainly be found in the architecture and dynamics of the Linux networking stack. In fact, as described in [10], the entire framework and dynamics of Linux packet processing architecture are guided by the NAPI mechanism, which allows to scalably perform packet processing operations.

In short, the NAPI's approach consists in handling the network interface requests with an interrupt (IRQ) moderation mechanism. When the traffic offered load rises, it allows to adaptively switch from a classical interrupt management of the interfaces to a polling one. In detail, the NAPI provides that packets' processing operations are performed during priority kernel tasks, namely SoftIRQs, with a maximum number of packets that can be elaborated during each SoftIRQ. When a SoftIRQ ends, if there are any more packets to be processed, a new SoftIRQ is scheduled. On the contrary, when no SoftIRQs are scheduled or active, and new packets are received, SoftIRQs are triggered by network boards' IRQ.

Figure 6 shows a session of internal measures performed with the Xeon based platform at the 3 GHz frequency, obtained using kernel profiling tools [11] and SW counters. These internal measures point out how the NAPI mechanism lowers the number of IRQs and SoftIRQs per forwarded packet, as the traffic offered load rises. In fact, a higher traffic load obviously produces the increase of both the average number of packets processed in a same SoftIRQ, and the SoftIRQ re-scheduling probability. Such decrease of IRQs and SoftIRQs obviously causes an increase of SW efficiency, since it allows to avoid a certain number of "overhead" operations.

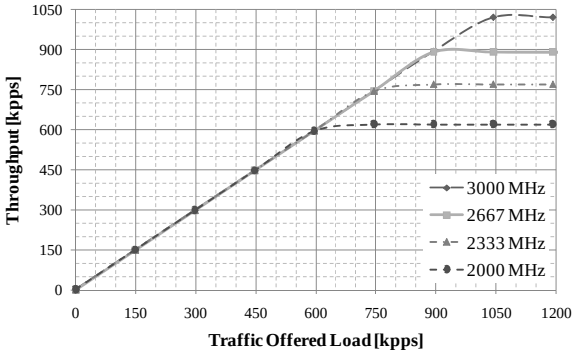

Figure 3. Throughput for the Xeon based architecture for a single forwarding Core, according to different frequencies and traffic offered loads.

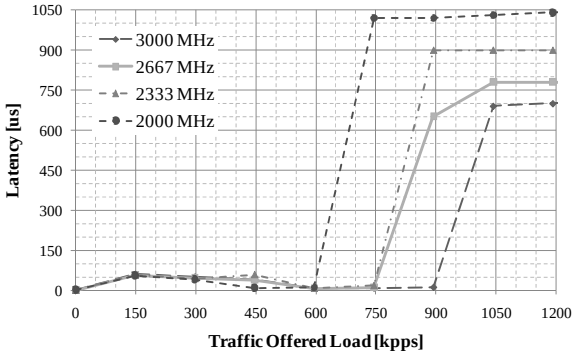

Figure 4. Latencies for the Xeon based platform for a single forwarding Core, according to different frequencies and traffic offered loads.

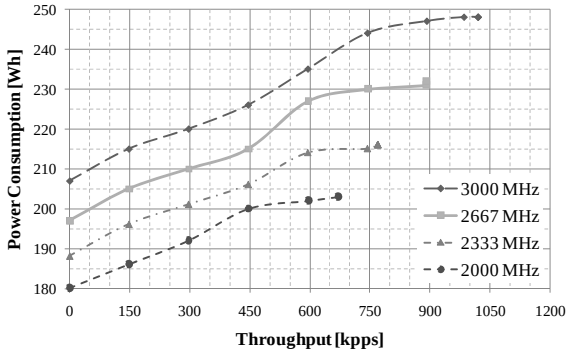

Figure 5. Power consumption and related throghput values for the Xeon based platform, according to different frequencies and traffic loads.

In confirmation of this, both the CPU idle time and the IRQ number get near to zero at about 700 kpps, before the maximum throughput is reached. Then, the throughput rise between 700 and 1,040 kpps, can be only attributed to the SW efficiency increase. Moreover, the IRQs and the CPU idle time zeroing at 700 kpps agrees with the gradient change of the power consumption profile in Figure 5. Thus, while we have an almost linear increase of power consumption up to the IRQ zeroing, for higher throughput values the power consumption achieves more or less its maximum value. This is a clear indication of the relationship between the power consumption profiles in Figure 5 and the SW internal dynamics.

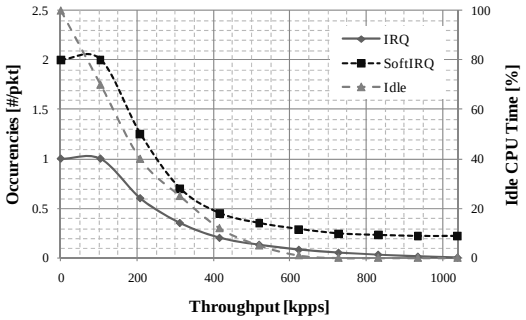

Figure 6. Linux kernel internal measures obtained with SW counters, and the Oprofile SW suite. The IRQ profile is referred to the second y-axis.

V. AN EMPIRICAL MODEL FOR SR POWER CONSUMPTION

Starting from the analysis and the benchmarking results in the previous Section, we propose a simple empirical model for describing the SR behavior in terms of throughput and power consumption. The main aim of the proposed model is to represent the SR performance according to traffic offered load and SR setups, in terms of active Core number and of working frequencies. Such a model obviously represents an indispensable tool to develop optimization mechanisms for SR power saving. To derive such empirical model, we starts by considering the simplest scenario, with only a single "forwarding" Core, in the sub-Section V.A, and then we refine the model in the sub-Section V.B for the multi-Core scenario.

A. The single Core case

As previously sketched, Figure 5 shows a clear dependency between power consumption and packet throughput, which is very similar for all the used Core frequencies. In particular, these tests (and others not reported here) absolutely suggest that the frequency impacts on such correlation in terms both of a scaling factor, and of a shifting one. Thus, we decided to process all the measured samples, collected in the "power consumption versus packet throughput" tests (including the ones shown in Figure 5). In particular, by shifting such curves down of their minimum power consumption (the value when throughput is equal to 0), and by normalizing both the power consumption and the throughput values, we obtained the interesting results shown in Figure 6, which suggest a common form envelope for power consumption according to Core frequencies, thanks to the narrow superposition of normalized measured samples.

In order to represent this profile form, we decided to introduce the normalized form function χ, and to derive it as a polynomial fitting function. In detail, we define χ as a 7th-order[1] polynomial as follows:

$$\chi(t) = \sum_{i=0}^{i=7} a_i t^i \quad (4)$$

Using the normalized values of collected measures for all the selected HW architecture, we calculated the a_i parameters through the method of least squares, and we obtained the following numerical values: a_0=0, a_1=0.47, a_2=15.01, a_3=-89.86, a_4=223.07, a_5=-256.47, a_6=130.47, a_7=-21.70.

[1] The choice of the 7th-order simply depends on the fact that it is the minimum order polynomial, which guarantees an acceptable accuracy level.

Starting from the χ definition, for a single Core, working at the frequency $f \in F$ and forwarding r packets per second, we can express the power consumption in the following way:

$$\Phi(f,r) = \Phi_{idle}(f) + [\Phi_{active}(f) - \Phi_{idle}(f)] \cdot \chi\left(\frac{r}{r_{max}(f)}\right) \quad (5)$$

where $r_{max}(f)$ is the maximum throughput at the frequency *f*.

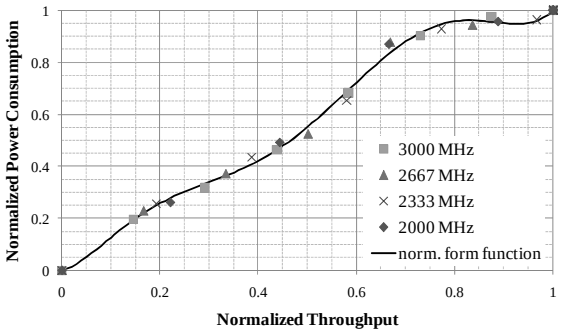

Figure 7. Normalized power consumptions, obtained according to traffic offered loads and to different Core frequencies, and the form function $\chi(\cdot)$.

Observing Figure 3, we decided to approximate the throughput r as the minimum value between the traffic offered load l and its maximum forwarding rate $r_{max}(f)$:

$$r = min\{l, r_{max}(f)\} \quad (6)$$

Note that, in the case of variable throughput values during a $[0,T]$ time period, the mean power consumption can be simply expressed as the $\Phi(f,r)$ contribution, averaged on the $r(t)$ rate values in the same time window $t \in [0,T]$:

$$\widetilde{\Phi} = \frac{1}{T}\int_{t=0}^{T} \Phi\big(f, r(t)\big)\mathrm{d}t \quad (7)$$

By replacing the $\widetilde{\Phi}$ and the $\Phi\big(f, r(t)\big)$ parameters from Eqs. 2 and 5, we obtain the following relationship between the normalized form function and the p_{idle}:

$$p_{idle} = 1 - \frac{1}{T}\int_{t=0}^{T} \chi\left(\frac{r(t)}{r_{max}(f)}\right)\mathrm{d}t \quad (8)$$

From Eq. 8, the χ function clearly represents the probability that the Core is active at a certain working frequency *f* and with a instantaneous throughput equal to *r(t)*.

As indicated in Section IV, the p_{idle} parameter mainly depends both on SW efficiency and feature, taken into account thanks to the χ normalized form function, and on the Core computational capacity, expressed as $r_{max}(f)$, at the working frequency *f*. Regarding the Φ_{idle}, the Φ_{active} and the r_{max} parameters, they obviously depend on the adopted HW architecture. In detail, Φ_{idle} and Φ_{active} and the set of admissible working frequencies *F* are specifications/features of CPU/Cores, usually published by chip manufactures.

On the contrary, the $r_{max}(f)$ parameter depends not only on the features of CPU/Core, but also on the ones of other HW elements, such as I/O busses and RAM. Since the complexity of SR architecture, the $r_{max}(f)$ values cannot be a priori fixed or estimated, but they must be obtained by experimentally measuring the performance of each selected HW architecture. An example of such experimental measures can be found in Figures 1 and 2, which clearly show a linear dependency between the $r_{max}(f)$ and the $\Phi_{active}(f)$ parameters according to different frequency values and HW architectures.

B. The multi-Core case

As explained in Section II, new generation SR architectures exploit multi-Core HW platform in a very effective way, since

they allow to parallelize the forwarding process among the active Cores as much as possible. However, Figure 1 clearly highlights that, as the number of active forwarding Cores increases, the performance gain is not the theoretical one for all the HW platforms. For certain HW platforms (e.g., the Opteron based one), when we increase the number of forwarding Cores from 1 to 2 units, the maximum throughput does not redouble, but it achieves a value near to the 150% of the single Core throughput. On the contrary, other HW platforms (e.g., the Xeon based one) show a performance gain near to the 200%. This variable gain depends on the specific HW platform, and it is caused by performance bottlenecks in other HW components (e.g., I/O bus, RAM, etc.).

To represent this HW-dependent features, and to take into account how the overall performance of the SR increase according to a certain number of active Cores, we introduce the constant γ. Extending Eq. 5 to the multi-Cores' case, we can express the power consumption as in the following:

$$\Phi(f_c, r_c | \forall c \in C) = \sum_{\forall c \in C} \Phi_{idle}(f_c) + \sum_{\forall c \in C} [\Phi_{active}(f_c) + -\Phi_{idle}(f_c)] \cdot \chi\left(\frac{r_c}{\frac{\gamma}{|C|} r_{max}(f_c)}\right) \quad (9)$$

where C is the set of Cores in the SR, and $|C|$ its cardinality; the f_c and the r_c parameters are the working frequency, and the packet throughput of the Core $c \in C$, respectively.

Regarding the SR overall throughput value r, we can simply extend Eq. 6 as in the following:

$$r = \sum_{c \in C} min\left\{l_c, \frac{\gamma}{|C|} r_{max}(f)\right\} \quad (10)$$

where l_c is the traffic offered load for the Core c.

C. A Simple Validation

With the aim of validating the proposed model, we performed further benchmarking sessions by using the HW platform introduced in the sub-Section IV.A. In detail, for each benchmarking session, we collected a set of power consumption measures according to different throughput values and Core number, and, then, we compared these measured values with the estimated ones, obtained with the proposed model. Figure 8 shows the maximum model error, with respect to the measured values, in terms of estimated power consumption for each selected HW architecture. The obtained maximum errors underline that the proposed model allows to describe the SR behavior with a high accuracy level.

VI. CONCLUSIONS

We focused on the "green" capabilities of new generation COTS SRs, able to effectively exploit multi-Core/CPU HW platforms. Our main objective was to analyze, to evaluate and to model the impact of power saving mechanisms, generally included in today's COTS processors, on the SR networking performance and behavior. To this purpose, we understood and separately characterized the roles of both HW and SW layers through a large set of internal (i.e., Linux SW counters and kernel profiling) and external (i.e., throughput, latencies and power consumption) experimental measures, obtained with a heterogeneous set of HW platforms and SR setups. This detailed analysis gave us the chance to derive a simple model, able to accurately represent the SR performance in terms of packet throughput and related power consumption. Our future objective is to extend this contribution by developing a "green optimization" mechanism, based on the model proposed, in order to minimize power consumption while maintaining a certain SR performance target.

REFERENCES

[1] Noguera, J.; Kennedy, I.O., "Power Reduction in Network Equipment Through Adaptive Partial Reconfiguration," *Proc. of the 2007 International Conference on Field Programmable Logic and Applications (FPL 2007)*, Aug. 2007, pp. 240-245.

[2] "Intel is Leading the Way in Designing Energy-Efficient Platforms", Intel White paper.

[3] Chabukswar, R.; "Maximizing Performance and Energy-Efficiency on Intel@ Core™ Microarchitecture using Multi-Threading", Intel White paper.

[4] Miyoshi, A., Lefurgy, C., Van Hensbergen, E., Rajamony, R., and Rajkumar, R., "Critical power slope: understanding the runtime effects of frequency scaling," *Proc. of the 16th international Conference on Supercomputing (ICS'02)*, New York, NY, USA, June 2002.

[5] Bianco, C.; Cucchietti, F.; Griffa, G., "Energy consumption trends in the next generation access network — a telco perspective," *Proc of the 29th International Telecommunications Energy Conference, 2007. (INTELEC 2007)*, Rome, Italy, Sept. 2007, pp.737-742

[6] Telecom Italia Website, "The Environment", URL: http://www.telecomitalia.it/sostenibilita2006/English/B05.html

[7] Bolla, R.; Bruschi, R., "An effective forwarding architecture for SMP Linux routers," *Telecommunication Networking Workshop on QoS in Multiservice IP Networks, 2008. IT-NEWS 2008. 4th International* , Venice, Italy, Feb. 2008, pp.210-216.

[8] Bolla, R.; Bruschi, R., "PC-based Software Routers: High Performance and Application Service Support," *Proc. of the ACM Sigcomm Worshop on Programmable Routers for Extensible Services of Tomorrow* (PRESTO'08), Seattle, WA, USA, Aug. 2008, pp. .

[9] Yi, Z.; Waskiewicz, P.J., "Enabling Linux Network Support of Hardware Multiqueue Devices," *Proc. of the 2007 Linux Symposium*, Ottawa, Canada, June 2007, pp. 305-310.

[10] Bolla, R.; Bruschi, R., "Linux Software Router: Data Plane Optimization and Performance Evaluation," *Journal of Networks (JNW)*, vol. 2, no. 3, Academy Publisher, pp. 6-11, 2007.

[11] Oprofile, http://oprofile.sourceforge.net/.

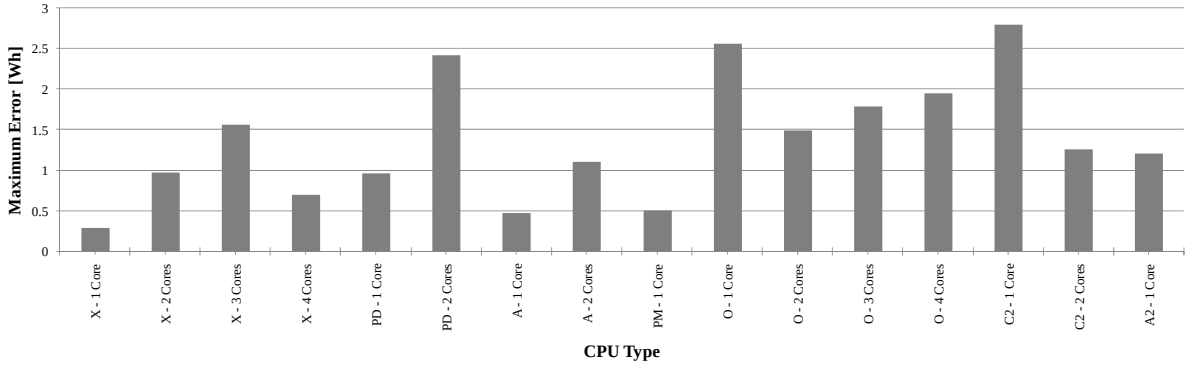

Figure 8. Maximum model error with respect to benchmarking measures and according to different HW platforms and Core number.