Power-Aware Parallel Forwarding: An Optimization Study

Weirong Jiang
Juniper Networks Inc.
Sunnyvale, CA, USA
Email: weirongj@acm.org

Yaxuan Qi
Tsinghua University
Beijing, China
Email: yaxuan@tsinghua.edu.cn

***Abstract*—This paper studies the optimization problem of distributing traffic load onto multiple engines in parallel packet forwarding systems. Given that the power dissipation of each engine can be formulated as a function of traffic load going through that engine, we develop a theoretical framework to minimize the overall power consumption while satisfying the throughput demand. We consider two types of power functions, which are different in terms of whether supporting sleep mode or not. Accordingly, the two power functions lead to two different mathematical programming problems: one can be solved via linear programming (LP); the other is modeled as non-linear programming and can be solved via mixed integer programming (MIP). Our simulation using a 18-hour real-life traffic trace shows that our solution can achieve significant power/ energy reduction compared with the traditional parallel forwarding scheme based on load balancing. We also discuss the system design issues and identify the challenges for real implementation. We believe our optimization framework and algorithmic solutions are applicable for load distribution in other parallel systems e.g. data centers and clusters.**

***Keywords*-load distribution; parallel packet forwarding; power-aware design;**

I. Introduction

The primary function of network routers/switches is to forward packets. With the network traffic growing rapidly, the throughput requirement becomes difficult to meet by using a single packet forwarding engine. For example, current backbone link rates have been pushed beyond OC-768 (40 Gbps) rate, which requires a throughput of 125 million packets per second (MPPS) for minimum size (40 bytes) packets. Employing multiple packet forwarding engines has been a standard in today's routers/switches. Most research in this area is on balancing the traffic among these engines to achieve high throughput [1]–[3]. None of them take power or energy consumption into account. On the other hand, as routers achieve aggregate throughputs of trillions of bits per second, power consumption has become an increasingly critical concern in backbone router design [4], [5]. Some recent investigations [6], [7] show that power dissipation has become the major limiting factor for next generation routers and predicts that expensive liquid cooling may be needed in future routers. Recent analysis by researchers from Bell labs [6] reveals that, almost 2/3 of power dissipation inside a core router is due to packet forwarding engines. Various techniques including data structure, hardware, system or network-level optimization [7]–[11] have been proposed to reduce the power consumption of routers. But most of them focus on a single packet forwarding engine. In this paper, we will show that an optimal distribution of traffic load onto parallel forwarding engines can achieve significant power/ energy reduction.

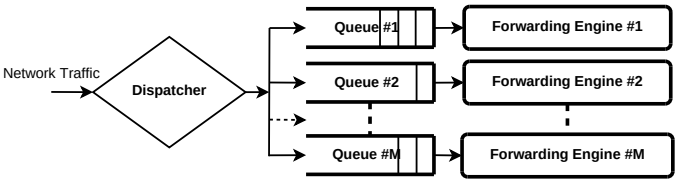

Figure 1. Conceptual model of a parallel packet forwarding system.

We consider the parallel forwarding system shown in Figure 1. Essential to such a system is the dispatcher that distributes incoming traffic load to the multiple forwarding engines. The basic goal of the dispatcher is to fully utilize the multiple engines to maximize the overall throughput. In this paper we also take power / energy consumption into account for designing the dispatcher in parallel forwarding systems. We make following contributions in this paper.

First, we develop a theoretical framework by modeling the power dissipation of a single engine as the function of the traffic load assigned to that engine. Then the power-aware load distribution in parallel forwarding becomes an optimization problem to minimize the overall power consumption while meeting the throughput requirement.

We discuss two types of power functions, both of which are based on linear power models. The first power function is from traditional packet forwarding engines which do not support sleep modes, while the second power function considers the packet forwarding engines with capability to sleep [11].

For each type of power functions, we obtain different variant of the optimization problem and propose using different algorithms to solve them. For packet forwarding engines without sleep mode, the problem becomes a linear programming problem. When packet forwarding engines are enabled to sleep, the programming problem is non-linear and we show that it can be converted into a mixed integer programming problem.

We also examine the system design issues and identify the challenges for real implementation. We hope this initial work can motivate more follow-up research.

We conduct numerical experiments using a 18-hour traffic trace which is obtained by concatenating multiple traces collected in one day from [12]. Experimental results show that our solution achieves up to 9-fold reduction in energy (and average power) consumption compared with traditional power-unaware parallel forwarding schemes.

The rest of the paper is organized as follows. Section II defines the problem and presents our solutions by considering two types of power functions. Section III discusses the system design issues. Section IV shows the experimental results. Section V gives a brief overview of related work. Section VI concludes the paper.

II. Optimization Framework

A. Problem Definition

The problem of distributing traffic load onto multiple forwarding engines to minimize the overall power consumption while satisfying the throughput requirement can be defined as follows.

$$\min \quad \sum_{i=1}^{M} P_i(x_i) \tag{1}$$

$$\text{s.t.} \quad \sum_{i=1}^{M} x_i = T \tag{2}$$

$$0 \le x_i \le B_i,\ i = 1, \ldots, M \tag{3}$$

In the above definition, M denotes the total number of forwarding engines and T the total throughput requirement. For the ith forwarding engine, $i = 1, \ldots, M$, x_i denotes its traffic load, $P_i(.)$ its power function with respect to its traffic load and B_i its throughput upper bound (i.e. the bandwidth).

B. Power Function of Each Engine

To solve the above optimization problem, we should first understand the power function of each engine. The most common are the following two types of power functions.

1) Sleep-disabled: According to [13], [14], the power consumption of most of today's network devices can be modeled as a linear function with respect to the traffic load, as shown in (4):

$$P(x) = ax + b \tag{4}$$

where x denotes the traffic load. a is the coefficient for dynamic power consumption while b the static power consumption. This type of power function assumes that the forwarding engine does not have the sleep mode. As a result, even there is no traffic to be processed, the forwarding engine still dissipates (static) power.

2) Sleep-enabled: Recent work [11] proposes that next generation network devices should support sleep mode. With sleep mode enabled, the forwarding engine will not dissipate any power when there is no traffic load. Thus we have following power function for such kinds of forwarding engines.

$$\begin{aligned} P(x) &= (ax + b)I_x = ax + bI_x \\ &= \begin{cases} 0 & x = 0 \\ ax + b & x > 0 \end{cases} \end{aligned} \tag{5}$$

where I_x is a unit step function:

$$I_x = \begin{cases} 0 & x = 0 \\ 1 & x > 0 \end{cases} \tag{6}$$

C. Specific Case: Homogeneous Engines

We start the discussion from a specific (and simple) case where all the forwarding engines have the same properties including the power function and the bandwidth. In other words, $\forall i$, $i = 1, 2, \ldots, M$, $a_i = a$, $b_i = b$, $B_i = B$, where a, b and B are constants.

1) Considering sleep-disabled forwarding engines, the objective function (1) will be

$$\begin{aligned} \sum_{i=1}^{M} P_i(x_i) &= \sum_{i=1}^{M} [ax_i + b] \\ &= a \sum_{i=1}^{M} x_i + Mb \\ &= aT + Mb \end{aligned} \tag{7}$$

Thus there is no optimization to be done to minimize the overall power consumption.

2) Considering sleep-enabled forwarding engines, the objective function (1) will be

$$\begin{aligned} \sum_{i=1}^{M} P_i(x_i) &= \sum_{i=1}^{M} [ax_i + bI_{x_i}] \\ &= a \sum_{i=1}^{M} x_i + b \sum_{i=1}^{M} I_{x_i} \\ &= aT + b \sum_{i=1}^{M} I_{x_i} \end{aligned} \tag{8}$$

Thus the optimal solution is to minimize the number of active forwarding engines while satisfying the throughput requirement. Since all the fowarding engines have the same bandwidth, the optimal solution for meeting throughput of T is to turn on $\lceil \frac{T}{B} \rceil$ forwarding engines while keeping the rest of forwarding engines to sleep. Then the overall power consumption is minimized to be: $\min \sum_{i=1}^{M} P_i(x_i) = aT + b \lceil \frac{T}{B} \rceil$.

D. General Case: Heterogeneous Engines

In most cases, the properties of forwarding engines differ from each other. In other words, $\exists i, j,\ i, j = 1, 2, \ldots, M$, $a_i \neq a_j$, $b_i \neq b_j$, and $B_i \neq B_j$.

1) Considering sleep-disabled forwarding engines, the objective function (1) will be

$$\sum_{i=1}^{M} P_i(x_i) = \sum_{i=1}^{M} [a_i x_i + b_i] = \sum_{i=1}^{M} a_i x_i + \sum_{i=1}^{M} b_i \quad (9)$$

Then the optimization problem becomes a linear programming (LP) problem which can be solved in polynomial time. Note that $\sum_{i=1}^{M} b_i$ is constant, which will not affect the decision on traffic load distribution. Hence $\sum_{i=1}^{M} b_i$ can be omitted from (9) when solving the linear programming (LP) problem.

2) Considering sleep-enabled forwarding engines, the objective function (1) will be

$$\sum_{i=1}^{M} P_i(x_i) = \sum_{i=1}^{M} [a_i x_i + b_i I_{x_i}] \quad (10)$$

The optimization problem then becomes a non-linear programming problem which is hard to be solved. We convert it into a mixed integer programming (MIP) problem[1] by introducing a penalty parameter K to remove the unit step function. Then we have:

$$\min \quad \sum_{i=1}^{M} [a_i x_i + b_i y_i] \quad (11)$$

$$\text{s.t.} \quad \sum_{i=1}^{M} x_i = T \quad (12)$$

$$0 \leq x_i \leq B_i,\ i = 1, \ldots, M \quad (13)$$

$$y_i <= K * x_i,\ i = 1, \ldots, M \quad (14)$$

$$y_i >= x_i / K,\ i = 1, \ldots, M \quad (15)$$

$$y_i \in \{0, 1\} \quad (16)$$

We can prove that when we set $K >> \max_{i=1}^{M} B_i$, the above problem is identical to (10) with constraints (2)(3). The simple proof is: If $x_i = 0$, then y_i must be 0; otherwise, i.e. $x_i > 0$, then y_i must be 1. Hence $y_i = I_{x_i}$. $i = 1, 2, \ldots, M$.

III. System Design

Although this paper is mainly on theoretical analysis, we discuss briefly in this section the issues of system design and implementation.

[1]Although a mixed integer programming (MIP) problem is *NP-hard*, there exist various computation-efficient MIP solvers.

A. Overall Architecture

To obtain the parameters used in the optimization framework, we need following components in the dispatcher system.

- **Load predication** to predict T in real time.
- **Bandwidth estimation** to estimate B_i $(i = 1, 2, \ldots, M)$ for each forwarding engine if their values are unknown or varying.
- **Power function profiling** to retrieve the parameters a_i, b_i $(i = 1, 2, \ldots, M)$ for each forwarding engine if these parameters cannot be pre-determined.

The kernel of the dispatcher is the linear programming (LP) / mixed integer programming (MIP) solver. Then we have the overall architecture as shown in Figure 2, where 'Q' and 'FE' are the abbreviations of 'Queue' and 'Forwarding Engine', respectively.

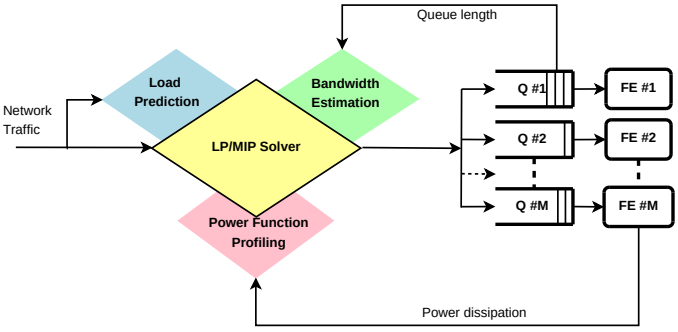

Figure 2. Example of the dispatcher system design.

B. Load Predication

The optimal solution is only possible when we can predict future traffic load. Various load prediction techniques have been proposed in literature [15], [16]. One of the accurate prediction algorithms is Auto-Regressive Moving Average (ARMA) adopted in [16]. We use ARMA for load prediction in our experiments.

C. Bandwidth Estimation

In most cases, the bandwidth of each forwarding engine is pre-known. But in case the forwarding bandwidth of some forwarding engine is unknown or varying, we need to perform real-time estimation using other information. For example, we can monitor the queue length of a forwarding engine. Since the dispatcher keeps track of the traffic load distributed to that forwarding engine, we can infer the forwarding bandwidth based on the queue length of that engine.

D. Power Function Profiling

Usually we can pre-determine the power function of each forwarding engine. If we want to model the power function of a forwarding engine on the fly, we need the real-time

Table I
STATISTICS OF THE 18-HOUR TRAFFIC TRACE

Trace	Date	# of packets	Duration	Max. throughput	Min. throughput
LBNL/ICSI Enterprise Trace	20050107	26325056	17 hours 33 minutes	11083 PPS	0 PPS

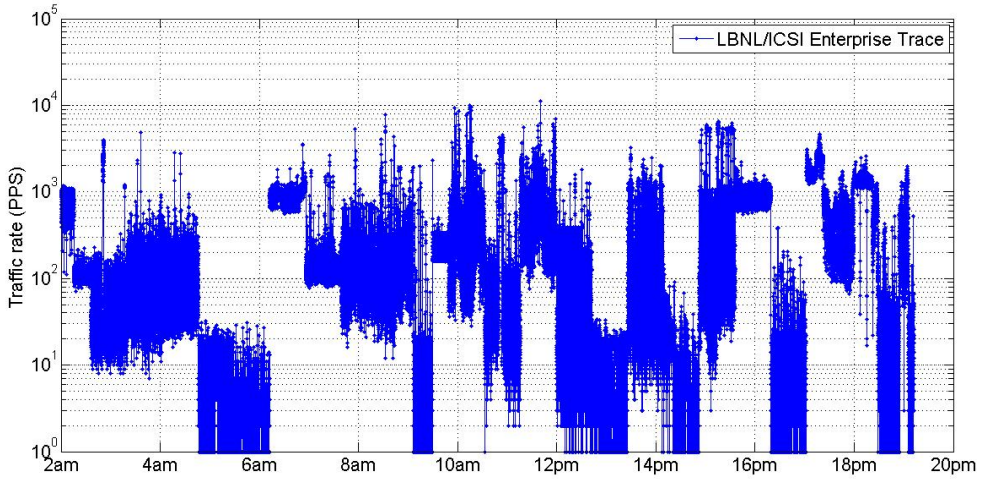

Figure 3. Traffic rate variation of the trace.

information of the power dissipation of the engine. Since the dispatcher contains the traffic load records, we can profile the power functions based on power and load information.

IV. EXPERIMENTAL RESULTS

To evaluate how much power/ energy reduction can be achieved by using the proposed optimization framework, we conducted simulation experiments using real-life traffic traces from LBNL/ICSI Enterprise Tracing Project [12]. We downloaded 25 trace files collected on the same day (2005/01/07) and concatenated them into one trace. Its statistics is shown in Table I where throughput is measured in terms of the number of packets per second (PPS). Figure 3 depicts the traffic rate variation during the entire period.

We consider the general case where the system consists of four parallel heterogeneous forwarding engines. The system configuration for the simulation is summarized in table II. The parameters were set to comply with the power models of network devices observed in [13], [14]. Among the four forwarding engines, the third one (FE #3) represents a high-end engine and is the most power-hungry, while the fourth one (FE #4) represents a low-end engine with the least power consumption.

In following experiments, we consider two scenarios: sleep-disabled and sleep-enabled forwarding engines, respectively. We compared the performance achieved using our optimal (**power-aware**) solution with that using the traditional (**power-unaware**) load balancing -based parallel forwarding scheme.

Table II
SUMMARY OF SIMULATION CONFIGURATIONS

	Parameter	Value
Engines	M	4
FE #1	a_1	0.1
	b_1	300
	B_1	3000
FE #2	a_2	0.05
	b_2	200
	B_2	1500
FE #3	a_3	0.2
	b_3	500
	B_3	6000
FE #4	a_4	0.02
	b_4	100
	B_4	600

A. With Sleep-Disabled Engines

Figure 4 shows that our power-aware scheme achieved lower power consumption than the traditional power-unaware scheme, especially when the traffic load is low. Since the static power consumption cannot be reduced in sleep-disabled forwarding engines, the overall energy reduction was marginal: our solution achieved 3% lower energy consumption than the traditional scheme. However, if we consider the dynamic part only, our solution achieved **4.1**-fold reduction in energy consumption than the traditional scheme.

Figure 5 shows the load distribution on the 4 forwarding engines. We can see that, the forwarding engine with the

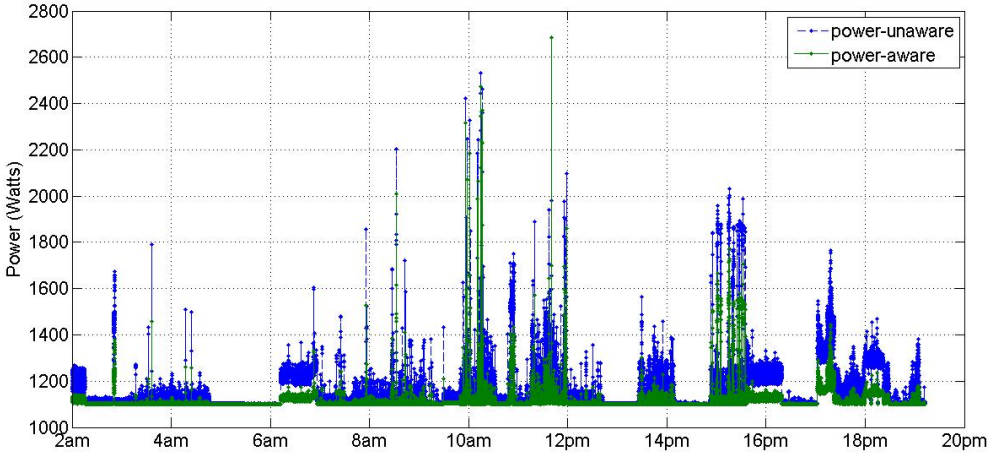

Figure 4. Power consumption of power-aware versus power-unware parallel forwarding schemes (with sleep-disabled forwarding engines).

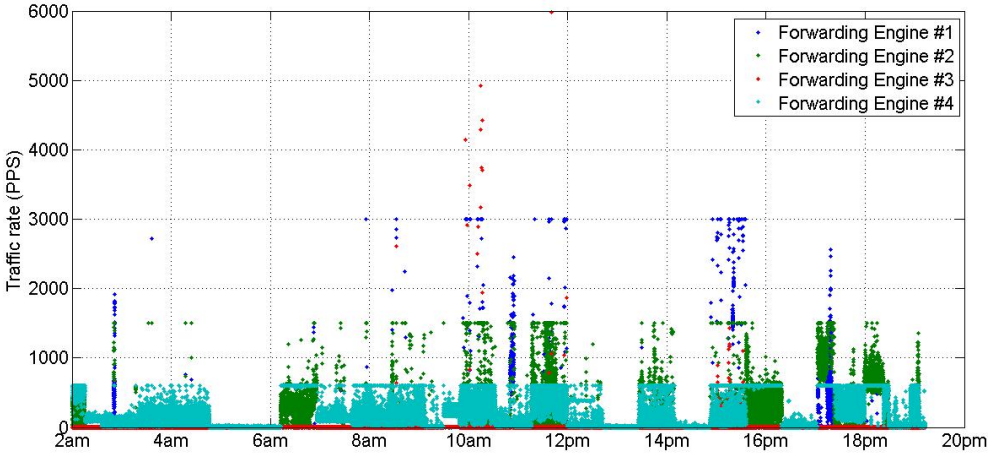

Figure 5. Load distribution on 4 sleep-disabled forwarding engines with power-aware parallel forwarding.

lowest dynamic power consumption usually reached its throughput upper bound (i.e. the bandwidth). The forwarding engines with lower dynamic power consumption tended to receive higher volume of traffic than those with higher dynamic power consumption. As a result, the short-term load distribution among the forwarding engines was not balanced. But the overall throughput requirement was still met.

B. With Sleep-enabled Engines

When forwarding engines have the option to sleep, our power-aware parallel forwarding scheme can achieve significant power/ energy reduction. As shown in Figure 6, our power-aware scheme achieved much lower power consumption than the traditional power-unaware scheme. The fundamental reason is that the traditional parallel forwarding scheme does not exploit the sleep mode of forwarding engines to reduce the static power consumption. Our power-aware parallel forwarding scheme turned on only portion of forwarding engines based on the optimization results. As a result, our solution achieved **9.13**-fold reduction in energy (and average power) consumption than the traditional power-unaware scheme.

Figure 7 shows the load distribution on the 4 forwarding engines. In most situations, the forwarding engines with lower power consumption tended to receive higher volume of traffic than those with higher power consumption, which is similar as Figure 5. When the traffic rate was low, the engine with high power consumption tended to sleep. But when the traffic rate was high, there were some situations where it was better to turn on one engine with high dynamic power consumption than multiple engines with low dynamic power consumption. Thus we observed different load distri-

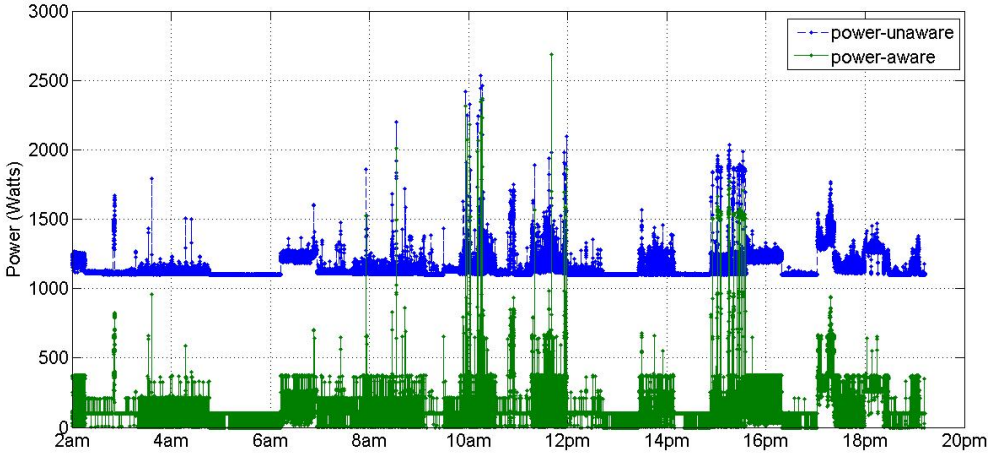

Figure 6. Power consumption of power-aware versus power-unware parallel forwarding schemes (with sleep-enabled forwarding engines).

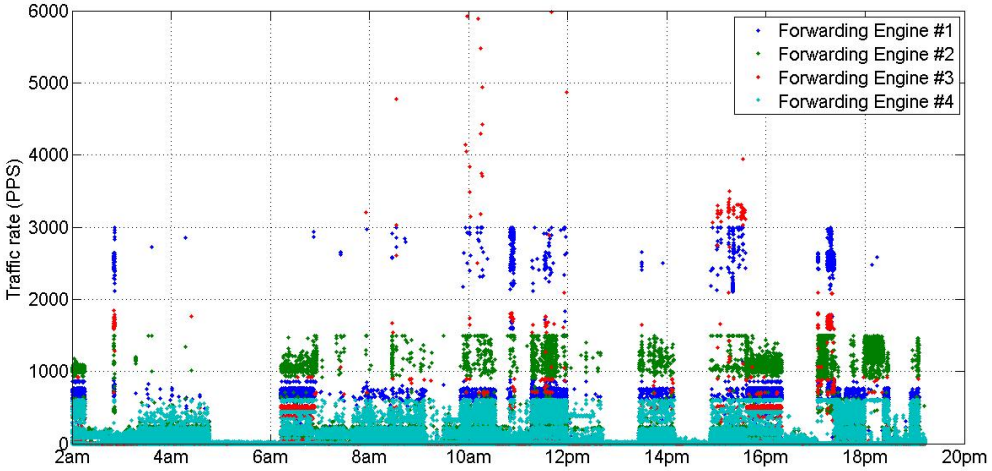

Figure 7. Load distribution on 4 sleep-enabled forwarding engines with power-aware parallel forwarding.

bution between Figure 7 and Figure 5. Again, though the short-term load distribution among the forwarding engines was not balanced, the overall throughput requirement was still met.

V. Related Work

Reducing the power consumption of packet forwarding systems has been a topic of significant interest [5], [7], [11]. Most of the existing work focuses on either single forwarding engines or the system- and network-level optimizations.

In [17] clock gating is used to turn off the clock of unneeded processing engines of multi-core network processors to save dynamic power when there is a low traffic workload. A finer-grained clock gating scheme is proposed in [9] to lower the dynamic power consumption of pipelined IP forwarding engines. In [18] the more aggressive approach of turning off these processing engines is used to reduce both dynamic and static power consumption. Dynamic frequency and voltage scaling are used in [19] and [10], respectively, to reduce the power consumption of the processing engines. In [8] the time-space trade-off of multi-bit tries is revisited from the point view of power consumption. A dynamic programming framework is proposed to determine the optimal strides for building tree-bitmap tries to minimize the worst-case power consumption. Some TCAM-based solutions [20], [21] propose various schemes to partition a routing table into several blocks and perform IP lookup on one of the blocks while "freezing" other blocks.

Chabarek et al. [7] enumerate the power demands of two widely used Cisco routers. The authors further use mixed

integer optimization techniques to determine the optimal configuration at each router in their sample network for a given traffic matrix. Nedevschi et al. [11] assume that the underlying hardware in network equipment supports sleeping and dynamic voltage and frequency scaling. The authors propose to shape the traffic into small bursts at edge routers to facilitate sleeping and rate adaptation.

VI. Conclusion and Future Work

Power consumption has become a major concern in design next generation network infrastructure. This paper represents the first study on power/ energy-aware parallel forwarding in routers/ switches. Given that the power dissipation of each engine can be modeled as a function of traffic load going through that engine, we formulated the optimization problem that minimizes the overall power consumption while satisfying the throughput demand. We discussed two types of power functions, in terms of whether they support sleep mode or not. We solved the two problems via linear programming (LP) and mixed integer programming (MIP), respectively. Our simulation using a 18-hour real-life traffic trace showed that our solution achieved up to **9.13**-fold reduction in energy (and average power) consumption compared with the traditional parallel forwarding scheme based on load balancing. We also discussed the system design issues and identified the challenges for real implementation.

On the other hand, many other issues remain open. For example, we did not consider the power consumption of the queues / buffers in this paper. Some applications require the parallel forwarding system preserve intra-flow packet order, which makes the problem more difficult. We hope our initial work can motivate more follow-up research in this area. We also believe the ideas proposed in this paper can be applied to other parallel computing systems with high power density, such as data centers, server farms, or clusters.

Acknowledgment

Yingtao Ren in University of Southern California, Los Angeles contributed the solution for converting the non-linear programming problem into the mixed integer programming problem. The authors would also like to thank the reviewers for their valuable comments.

References

[1] B. Chen and R. Morris, "Flexible control of parallelism in a multiprocessor pc router," in *Proc. of the General Track: 2001 USENIX Annual Technical Conference*. Berkeley, CA, USA: USENIX Association, 2001, pp. 333–346.

[2] W. Bux, W. E. Denzel, T. Engbersen, A. Herkersdorf, and R. P. Luijten, "Technologies and building blocks for fast packet forwarding," *IEEE Communications*, vol. 39, no. 1, pp. 70–77, 2001.

[3] W. Shi, M. H. MacGregor, and P. Gburzynski, "Load balancing for parallel forwarding," *IEEE/ACM Trans. Netw.*, vol. 13, no. 4, pp. 790–801, 2005.

[4] D. E. Taylor, "Survey and taxonomy of packet classification techniques," *ACM Comput. Surv.*, vol. 37, no. 3, pp. 238–275, 2005.

[5] M. Gupta and S. Singh, "Greening of the Internet," in *Proc. SIGCOMM*, 2003, pp. 19–26.

[6] A. M. Lyons, D. T. Neilson, and T. R. Salamon, "Energy efficient strategies for high density telecom applications," *Princeton University, Supelec, Ecole Centrale Paris and Alcatel-Lucent Bell Labs Workshop on Information, Energy and Environment*, June 2008.

[7] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright, "Power awareness in network design and routing," in *Proc. INFOCOM*, 2008, pp. 457–465.

[8] W. Jiang and V. K. Prasanna, "Architecture-aware data structure optimization for power-efficient ip lookup," in *Proc. HPSR*, 2010.

[9] ——, "Reducing dynamic power dissipation in pipelined forwarding engines," in *Proc. ICCD*, 2009.

[10] M. Mandviwalla and N.-F. Tzeng, "Energy-efficient scheme for multiprocessor-based router linecards," in *Proc. SAINT*, 2006.

[11] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," in *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, 2008, pp. 323–336.

[12] LBNL/ICSI Enterprise Tracing Project, "http://www.icir.org/enterprise-tracing/download.html."

[13] V. Valancius, N. Laoutaris, L. Massoulié, C. Diot, and P. Rodriguez, "Greening the internet with nano data centers," in *CoNEXT '09: Proceedings of the 5th international conference on Emerging networking experiments and technologies*, 2009, pp. 37–48.

[14] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices," in *NETWORKING '09: Proceedings of the 8th International IFIP-TC 6 Networking Conference*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 795–808.

[15] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-aware server provisioning and load dispatching for connection-intensive internet services," in *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, 2008, pp. 337–350.

[16] K. Le, R. Bianchini, M. Martonosi, and T. D. Nguyen, "Cost- and Energy-Aware Load Distribution Across Data Centers," in *Proc. Workshop on Power Aware*

Computing and Systems (HotPower '09), 2009.

[17] Y. Luo, J. Yu, J. Yang, and L. N. Bhuyan, "Conserving network processor power consumption by exploiting traffic variability," *ACM Trans. Archit. Code Optim.*, vol. 4, no. 1, p. 4, 2007.

[18] R. Kokku, U. B. Shevade, N. S. Shah, M. Dahlin, and H. M. Vin, "Energy-Efficient Packet Processing," in *http://www.cs.utexas.edu/users/rkoku/RESEARCH/energy-tech.pdf*, 2004.

[19] A. Kennedy, X. Wang, Z. Liu, and B. Liu, "Low power architecture for high speed packet classification," in *Proc. ANCS*, 2008, pp. 131–140.

[20] F. Zane, G. J. Narlikar, and A. Basu, "CoolCAMs: Power-efficient TCAMs for forwarding engines." in *Proc. INFOCOM*, 2003.

[21] W. Lu and S. Sahni, "Low power tcams for very large forwarding tables," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 948–959, 2010.