A PERFORMANCE MODEL FOR NETWORK PROCESSOR ARCHITECTURES IN PACKET PROCESSING SYSTEM

Mahmood Ahmadi and Stephan Wong
Computer Engineering Laboratory
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
{mahmadi, stephan}@ce.et.tudelft.nl

Abstract

Network processors (NPs) are designed to provide both performance and flexibility through the implementation of both parallel and programmable architectures. Typically, such processors encompass a parallel processor core with several memories and specialized co-processors. A common task performed by such processors is packet processing that is both complex and highly repetitive. Consequently, the challenge is to define an on-chip network processor architecture that is capable of meeting the performance requirements of packet processing. With the current technological advances, it is expected that many (network) processor cores are to be incorporated onto the same chip to perform packet processing for which an efficient configuration must be determined. In this paper, we propose a general framework for analyzing the performance of a network processor that consists of an (on-chip) network of NPs. For this purpose, we utilize queuing theory to model the proposed network processor and analyze it. More specifically, the Jackson network model is utilized to represent our network processor. The simulation results show that the proposed network processor is able to improve the response time and throughput when compared to a more traditional network processor.

Keywords: Network processors, queuing theory, Jackson network, bandwidth limitation

1 Introducion

The bandwidth growth of networks increased almost exponentially in the past couple of years and is expected to continue for years to come. This has been fueled by new emerging technologies that are capable of achieving higher bandwidths. Consequently, new applications are being developed that take advantage of the new capabilities. In turn, more consumers are starting to use these applications and thereby increasing the demand for higher bandwidth. The bandwidth growth and applications variety fueled the need for high performance network processors; network processors combine the flexibility of the general-purpose processors with high performance of the application-specific integrated circuits (ASICs). The type of processing in network processors is different from processors found in servers and workstations. Typically, a network processor comprises a parallel programmable processor core with a number of memory interfaces and special co-processors that are optimized for packet processing [1][2][4].

The packet processing tasks have specific requirements in of response time and throughput. A challenging issue in packet processing is: how to determine whether a certain NP architecture or an array of NP architectures are able to meet requirements. A well-accepted approach to investigate the mentioned issue is to utilize performance modeling and analysis that provide quantitative results. Most NP vendors provide development tool sets (including simulators) that can be used to estimate the performance of NPs. However, most simulators perform the performance analysis after the NP is designed and implemented. It should be noted, designing an NP is time consuming and the simulators are tailored towards specific NPs. To overcome these limitations, we conduct performance modeling and analysis using queuing theory and Jackson network models. There has been no other work in network processor performance modeling that use open queuing theory. Traditionally, queuing network analysis is a valuable tool for modeling and determining the performance and operating characteristics of computing system and communication networks [1][3]. In this paper, we propose to utilize the Jackson network to derive a model for the NP architecture. We investigate and analyze the model, consider it in an on-chip environment, and show that the proposed architecture decreases response time and increases the throughput when compared to traditional network processor architectures.

The paper is organized as follows. Section II presents related work. Section III describes a summary of Jackson models. Sections IV describes the model for NP architectures. Section V explains simulation results of the model. Section VI presents the overall conclusions.

2 Related Work

In this section, we present different efforts by researchers in design space exploration and performance modeling using close queuing in the network processing area. In [7], the StepNP is introduced an exploratory network processor simulation environment for exploring applications, multiprocessor network processing architecture, and system on chip (SOC) tools. The StepNP is modeled at the functional

and transaction level and not at cycle accurate level. In [8], queuing model analysis is presented as a valuable tool for investigation the performance and operating characteristic of communication networks and computer systems. In [6], J. Lu et al., proposed performance analysis of network processor-based application design using a close queuing model and MVA algorithm. Our model describes NP architectures based on open queuing technique and Jackson model and operates at system level.

3 Queuing Network and Jackson Model

In this section, we present the concepts of a network of queues and the Jackson model. In the queuing network analysis, it is possible to determine the performance and operating characteristics of real-world systems, such as communication systems, computer networks, and numerous other applications. A queuing network comprises a collection of interconnected queues that are served by single or multiple servers. Items arriving at the network request a service from one or more servers and may leave the network afterwards [3]. A Jackson network consists of M interconnected queues (with their own servers) - also called nodes - that satisfy the following conditions:

1. Items arrive from outside the system according to the Poisson process with are rate s_i. Items may also arrive from other nodes within the network.

2. The servers act as Poisson processes with exponentially distributed service times with service rate μ_i for node i.

3. Items leaving node i have a probability of p_{ij} to be routed to node j. Therefore, the probability that an item will leave the network is $1 - \sum_{j=1}^{M} p_{ij}$.

4. The utilization of all nodes is less than one.

The arrival rate λ_i at each node i is (from both internal and external sources):

$$\lambda_i = s_i + \sum_{j=1}^{M} p_{ji}\lambda_j, i = 1, ..., M \qquad (1)$$

In this equation, s_i represents the external arrival rate at each node, p_{ji} represents the routing probability between nodes j and i, and λ_j represents the arrival rate at node j. For each network, we have M arrival equations forming a linear system that can be solved. For networks that satisfy the above mentioned conditions, Jackson [5] proved that when in equilibrium the nodes can be treated as $M/M/1$ queuing networks with an arrival rate λ_i and service rate μ_i. In the Jackson network, important parameters are: the mean number of items (mean queue length) and mean resident time (response time) [5][9]. The mean number of items in each server i is:

$$N_i = \frac{\rho_i}{1 - \rho_i} \qquad (2)$$

The total mean number of items in the network is:

$$\bar{N} = \sum_{i=1}^{M} N_i = \sum_{i=1}^{M} \frac{\rho_i}{1 - \rho_i} \qquad (3)$$

The mean resident time T_s (response time) of an item in the network is:

$$T_s = \frac{\bar{N}}{\lambda} = \frac{1}{\lambda}\sum_{i=1}^{M} \frac{\rho_i}{1 - \rho_i} = \frac{1}{\lambda}\sum_{i=1}^{M} \frac{\lambda_i}{\mu_i - \lambda_i} \qquad (4)$$

4 NP Core Architecture

In this section, we present the simple abstract NP model and an NP architecture model. Subsequently, we apply the model to on-chip environments.

4.1 Simple Abstract NP Model

The processing within a network processor can be conceptually subdivided into two planes, namely the data plane and the control plane. They differ in the manner and speed. In the data plane, simple and highly repetitive tasks are being performed. The majority of packets are being processed in this plane of the NP. In the control plane, so-called control packets are handled that require more complex operations in their handling. A simple view of both planes in this simple abstract NP model is depicted in Figure 1(A).

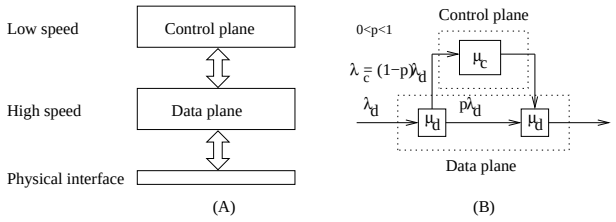

Figure 1. NP model. (A) Simple abstract NP model. (B) Simple abstract NP queuing model.

Based on the abstract NP model, we can derive a queuing model with the mapping of each plane to a separate queue. The resulting queuing model is depicted in Figure 1(B). We call this model the Abstract NP Queuing (ANPQ) model. In this figure, the λ_d and μ_d are the arrival rate and the service rate in the data plane, respectively. λ_c and μ_c are the arrival rate and the service rate in the control plane, respectively. Some packets are sent to the control plane and after the processing are forwarded to the data plane to complete the processing. Therefore, the control plane processing unit receives a flow with arrival rate $(1 - p)\lambda_d$ from data plane processing unit and then sends

it to data plane processing unit. The incoming flow to second data plane processing unit are of two. Using Eq. 4, the response time T_s in the ANPQ model is as follows:

$$T_s = \frac{1}{\lambda}\left(\frac{\lambda_d}{\mu_d - \lambda_d} + \frac{p\lambda_d}{\mu_d - p\lambda_d} + \frac{(1-p)\lambda_d}{\mu_c - (1-p)\lambda_d}\right) = \left(\frac{1}{(\mu_d - \lambda_d)} + \frac{p}{(\mu_d - p\lambda_d)} + \frac{(1-p)}{(\mu_c - (1-p)\lambda_d)}\right) \quad (5)$$

In this equation, the first line is extracted based on the Jackson theorem and λ represents the arrival rate of the overall system which it can be simplified to the second line of the equation. It should be noted that in the ANPQ overall system arrival rate λ is equal to λ_d ($\lambda = \lambda_d$).

4.2 Model of an On-chip NP Architecture

In the previous section, we have introduced the simple abstract queuing model that perceives the processing within an NP being separated into two planes. However, this model is no longer adequate in current NP chips as additional processor cores are being added to further improve processing speed. Therefore, we introduce the general NP architecture model (depicted in Figure 2) to reflect this reality. We intend to use this model to predict and investigate response times, throughput, and other metrics within these modern NP chips.

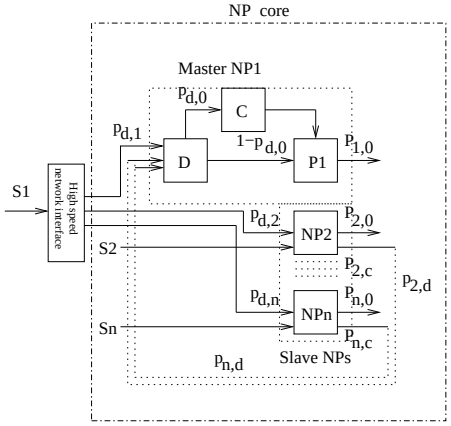

Figure 2. Model of a single chip NP.

The model depicted in Figure 2 comprises several cores that can be perceived as stand-alone NPs by themselves. Therefore, we refer to them in the remainder of this paper as NPs. In this model, we additionally assume that the master NP (NP1) can perform operations from both the data and the control plane. In case it cannot handle the incoming rate in the data plane (when $S1$ is larger than the service of the master NP), slave NPs are to assist in the processing. This decision is made at the 'high-speed network interface'. In Figure 2, the D and P1 elements are data plane processing units and C control plane processing unit within the master-NP. The input packet stream S_1 is distributed over different slave-NPs and the master-NP. p_{di} (with $i \geq 1$) represents the forward routing probabilities inside the model from the network interface and they are independent of internal probabilities p_{d0} and $1 - p_{d0}$ in the control and data planes in the master-NP. p_{id} (with $i \geq 1$) represents the backward routing probabilities. We use the Jackson theorem and write the arrival equations for the model as follows:

$$\begin{array}{ll} \lambda_{p_1} = p_{d0}\lambda_d & \lambda_c = (1-p_{d0})\lambda_d \\ \lambda_d = S_1 p_{d1} + \lambda_2 p_{2d} + \lambda_3 p_{3d} + ... + \lambda_n p_{nd} & \\ \lambda_2 = S_1 p_{d2} + S_2 & \lambda_3 = S_1 p_{d3} + S_3 \\ ..... & \lambda_n = S_1 p_{dn} + S_n \end{array} \quad (6)$$

The simplified form of Eq. 6 is given in the following:

$$\begin{array}{l} \lambda_d = S_1 p_{d1} + \sum_{i=2}^{n} \left(S_1 p_{di} + S_i\right) p_{id} = \\ S_1 \left(p_{d1} + \sum_{i=2}^{n} p_{di} p_{id}\right) + \sum_{i=2}^{n} S_i p_{id} \\ \lambda_i = S_1 p_{di} + S_i,\ i = 2...n \\ \lambda_1 = p_{d0}\lambda_d \quad \lambda_c = (1-p_{d0})\lambda_d \end{array} \quad (7)$$

The mean response time for the model is:

$$T_s = \frac{1}{\lambda}\left(\frac{\lambda_d}{\mu_d - \lambda_d} + \frac{\lambda_c}{\mu_c - \lambda_c} + \frac{\lambda_{p1}}{\mu_d - \lambda_{p1}} + \sum_{i=2}^{n} \frac{\lambda_i}{\mu_i - \lambda_i}\right) \quad (8)$$

In Eq. 7, the value of p_{di} is computed based on proportional allocation using Eq. 9. In proportional allocation, each slave-NP has 'vacant' capacity (v_i), the value of p_{di} is estimated from the unallocated capacity divided by the summation of all unallocated capacities for all the slave-NPs.

$$p_{di} = \frac{v_i}{\sum_{i=1}^{i=n} v_i} \quad (9)$$

It should be noted that the slave-NPs with larger unallocated capacity values are the best candidates. In this model, the value of p_{d0} shows the rate of control plane processing and it indirectly depends on the value of p_{di}.

4.3 Real Environment Exploration

In this section, we investigate the model in real environment. Recently, hardware vendors have begun delivering commodity NPs that again reflect Moore's law-style scaling with the parallelization gains coming from multi-core architecture. Based on model characteristics, we can observe that the model can be applied to an on-chip environment. In this environment, all NPs are integrated using on-chip technology with one NP core operating as the master-NP and others as slave-NPs. In the model, the majority of incoming packets were forwarded to the slave-NPs at the network interface point without processing. The slave-NPs process the received packets and send back the non-handled packets to the master-NP. We want to investigate on-chip NP architecture and therefore have modeled it. The model gives us insights on how the architecture should be implemented.

5 Simulation Results

In this section, we present the simulation results of the proposed model. The simulation results have been generated using Maple v.10.0. Based on the general queuing model, the following assumptions have been made:

1. Each NP is analyzed by (M/M/1) or (M/M/c) queuing model where the incoming packets obeys the Poisson distribution. Additionally, the service time distribution is exponential.

2. The inter-processor communication delay has been ignored in the modeling phase because the communication line bandwidth is large. The related delay may be added after the response time estimation.

3. In many cases, the average service rate is much greater than the average arrival rate, in this case the waiting queue would not grow too long. If the input buffer is reasonably large dropping packet is not an issue.

In this investigation, we assume a large pool of available NPs (64 in this experiment) each having a random service and arrival rate to 'mimic' reality as if they were are already in operation. Furthermore, out of this pool, the master NP is allowed to choose up to 16 NPs as slave NPs to assist itself in the processing of incoming packets. This investigation is not intended to be realistic for current-day processors, but instead we are trying to determine out of two selection mechanisms which one is able to improve the minimum response time and throughput. The model is used as a vessel to achieve this determination as we expect its importance will become evident when increasingly more processor cores will fit on future chips. The first selection mechanism is the first-in and first-out (FIFO) mechanism that list all possible slave NPs and deciding which one to use is solely based on which NP was entered first in the list. This is a simple selection mechanism that should be easy to implement, but will have some adverse effect on the minimum response time and throughput as presented in the following sections. The second selection mechanism first determines for all NPs their largest difference between their arrival rates and service rates and second chooses the one with the largest difference to assist the master NP. This mechanism is more complex as it requires almost continuous monitoring, but it is expected to yield better results.

5.1 Minimum response time investigation

In this section, we investigate the minimum response time of the proposed model depending on the two introduced selection mechanism introduced in the previous section. Using the FIFO mechanism, the resulting response time and arrival rate graphs with different number of slave-NPs are depicted in Figure 3 (A).

We can observe that the best response time is achieved when the number of processors is increased. The best response time is achieved with 16 slave-NPs. N=1 shows the response time of the ANPQ model. The arrival and service rates distribution graph for the 16 selected slave-NPs out of 64 NPs from processors pool is depicted in Figure 3 (B).

A limitation in the FIFO slave-NP selection mechanism is unreliability. In other words, it randomly selects slave-NPs from processors pool and when the difference between service and arrival rates for some processors are low, the overall system response time will increase. Based on the Figure 3 (C), the response time of the model is decreased for N=2, N=4 and is increased for N=8 and N=16. In this case, the graph of service and arrival rates difference based on the selection of 16 slave-NP out of 64 NP in processors pool is depicted in Figure 3 (D). We can observe that the difference between service and arrival rates are low for slave-NPs 7, 8 and 14, therefore, the response time is increased when the number of slave-NPs are 8 or 16. In other words, the FIFO slave-NP selection mechanism is not a very useful mechanism. An alternative to modify the FIFO slave-NPs selection mechanism is utilizing a threshold value in the FIFO mechanism. In this threshold FIFO mechanism, the processors with a lower difference than the threshold value are not selected as slave-NPs. The threshold value can be determined as $threshold = \frac{\mu_d}{k}$. In this equation, k is the number of slave-NPs. The response time and arrival rate graph is depicted in Figure 4.

From the Figure 4 (A), we can observe that the increasing number of processor decrease the response time. The model response time and arrival rate graph for NP with largest difference between service and arrival rates mechanism is depicted in Figure 4 (C). Based on this figure, we can observe that the lowest response time is generated when the number of slave-NPs is 16. We have to note that this number could be higher when more slave-NPs were allowed to be chosen and when 'les active' NPs could still be found. The arrival and service rate distribution based on the 16 selected slave-NP out of 64 NP in processors pool is depicted in Figure 4 (D).

5.2 Throughput investigation

In this section, the throughput of the model for different configurations is presented. The results using the threshold-FIFO mechanism is depicted in Figure 5. The response time for different values of arrival rate and different number of slave-NPs with the backward probability $p_{id} = 0.1$ is depicted in Figure 5 (A). Based on the figure, we can observe that the growth the number of slave-NPs increases the stability of the model. In this case, the master-NP service rate is 10^6 item/sec, therefore, the arrival rate of the ANPQ model can increase to the same value. Using this model, we can see the arrival rate of the system increases up to $4 * 10^6$ items/sec and $8.5 * 10^6$ items/sec

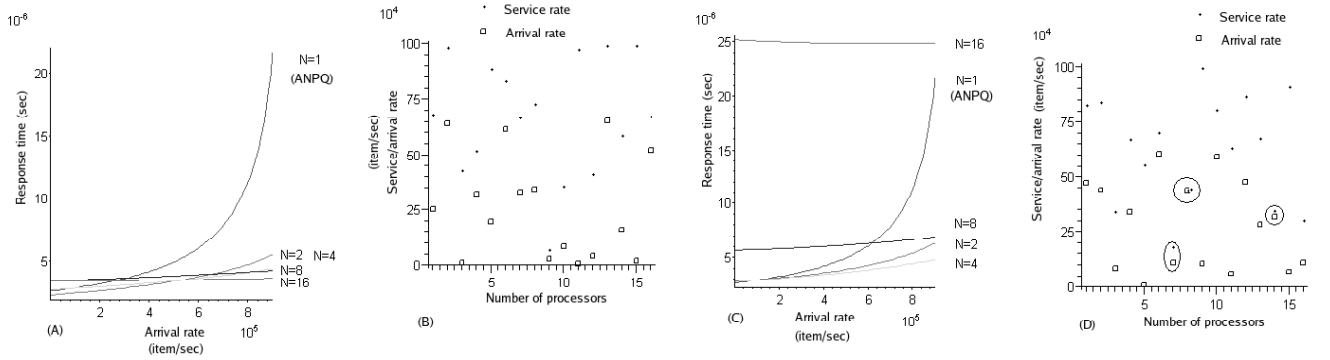

Figure 3. Model response time. (A) Response time for different configuration based on FIFO mechanism. (B) The service and arrival rates distribution. (C) Model response time for different configuration based on FIFO mechanism when the difference between service and arrival rate is low. (D) The service and arrival rates distribution.

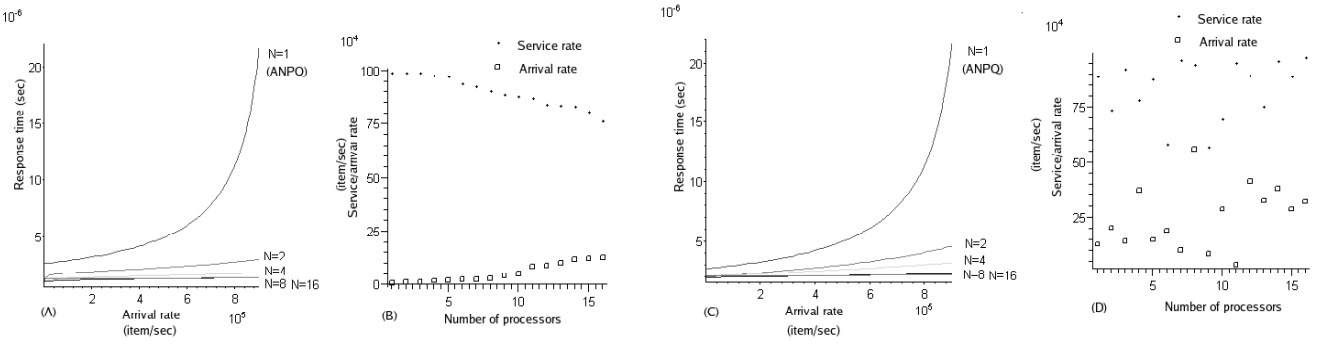

Figure 4. Model response time. (A) Response time for different configuration based on threshold-FIFO mechanism. (B) NPs service and arrival rates distribution. (C) Response time for different configuration based on processor selection based on the largest difference between service and arrival rate mechanism. (D) NPs service and arrival rates distribution.

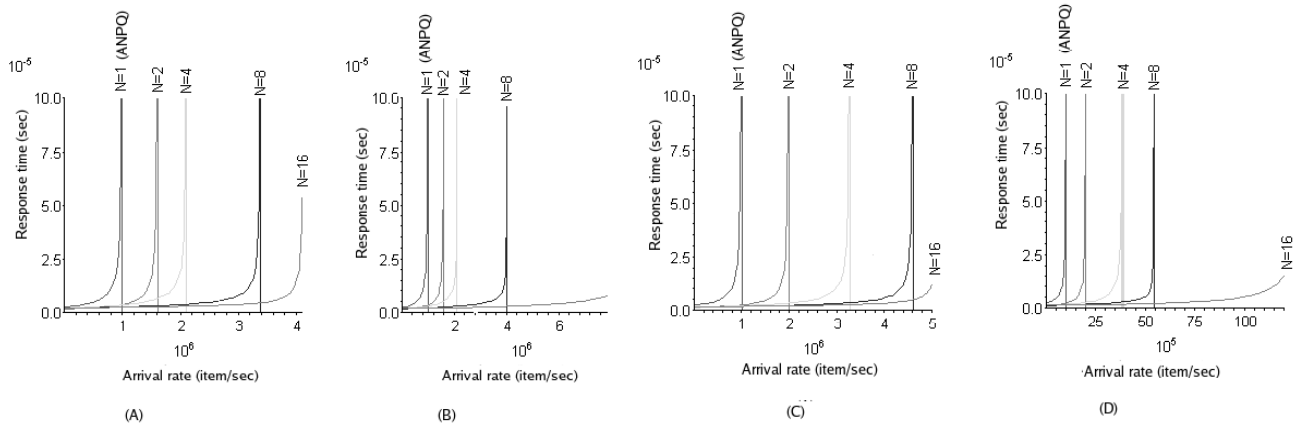

Figure 5. Response time using threshold-FIFO mechanism when the arrival rate (bandwidth) is increased. (A) Response time for different number of slave-NPs with $p_{id} = 0.1$. (B) Response time for different number of slave-NPs with $p_{id} = 0.001$. (C) Response time for different number of slave-NPs with $p_{id} = 0.1$. (D) Response time for different number of slave-NPs with $p_{id} = 0.001$.

when $p_{id} = 0.1$ and $p_{id} = 0.001$, respectively. The response time and arrival rates graph for different number of slave-NPs with the backward probability $p_{id} = 0.001$ is depicted in Figure 5 (B).

The response time for different values of arrival rate and different number of slave-NPs with the backward probability $p_{id} = 0.1$ using processor selection based on the largest difference between service and arrival rates mechanism is depicted in Figure 5 (C). Based on the figure, we can observe that the growth the number of slave-NPs increase the stability of the model. In this case, the master-NP service rate is 10^6 items/sec,therefore, the arrival rate of the ANPQ model can increase to the same value. Using the model, we can observe that the arrival rate of the system increases up to $5.5*10^6$ items/sec and $11*10^6$ items/sec for the cases when the $p_{id} = 0.1$ and $p_{id} = 0.001$, respectively. Based on these figures, we can observe that the processor selection based on the largest difference between service and arrival rates mechanism is more reliable than the threshold-FIFO mechanism against bandwidth variation.

6 Conclusion

In this paper, we proposed an abstract model for network processor as ANPQ using network of queues and open queues. Based on the model, we presented a model for NP architectures. This model operates in parallel to processes packets in network processing environments. Using the Jackson model, we obtained the response time and throughput of the model. We additionally presented two slave-NP selection mechanisms to choose which NPs should corporate with the master-NP as slave-NPs. These mechanisms were called 'FIFO' and 'processor selection based on the largest difference between service and arrival rate'. The model has been tested for FIFO and processor selection based on the largest difference between service and arrival rate slave-NP allocation mechanisms.

References

[1] I. Adan and J. Resing. "Queuing Theory". http:// www.cs.duke.edu/ fishhai/misc/queue.pdf, February 2001.

[2] M. Ahmadi and S. Wong. "Network Processors: Challenges and Trends". In *Proceedings of the 17th Annual Workshop on Circuits, Systems and Signal Processing, ProRisc 2006*, November 2006.

[3] R. O. Baldwin, N. J. Davis, S. F. Midkiff, and J. E. Kobza. "Queueing Network Analysis: Concepts, Terminology, and Methods". *Journal of Systems and Software*, 66(2):99–117, December 2003.

[4] M. A. Franklin and T. Wolf. "A Network Processor Performance and Design Model with Benchmark Parameterizations". In *Proceedings of Network Processor Workshop (HPCA-8)*, pages 63–74, Cambridge, MA, February 2002.

[5] P. G. Harrison and N. M. Patel. "*Performance Modeling of Communication Networks and Computer Architectures*". Addison-Wesley Longman, 1st edition, 1992.

[6] J. Lu and J. Wang. "Analytical Performance Analysis of Network Processor-Based Application Design". In *Proceedings of International Conference on Computer Communications and Networks*, pages 78–86, October 2006.

[7] P. G. Paulin, C. Pilkington, and E. Benisiudane. "StepNP: A System-Level Exploration Platform for Network Processors". *IEE Design and Test of Computers*, 19(6):17–26, November 2002.

[8] C. H. Sauer and K. M. Chandy. "*Computer Systems Performance Modeling*". Prentice Hall, March 1981.

[9] J. Virtamo. "Queueing Course, Complete Lecture Notes". http://www.netlab.hut.fi/ opetus/s383143/ kalvot/english.shtml, 2005.