8-2017

PaSE : Parallel Speedup Estimation Framework for Network-on-Chip Based Multi-core Systems

Ghassan Bachay Dharb
gbd7047@rit.edu

PaSE : Parallel Speedup Estimation Framework for Network-on-Chip Based Multi-core Systems

Ghassan Bachay Dharb

PaSE : Parallel Speedup Estimation Framework for Network-on-Chip Based Multi-core Systems

Ghassan Bachay Dharb
August, 2017

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science
in
Electrical Engineering

Supervised by
Dr. Amlan Ganguly
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
August 2017

R·I·T | Kate Gleason *College of* ENGINEERING

Department of Electrical and Microelectronic Engineering

PaSE : Parallel Speedup Estimation Framework for Network-on-Chip Based Multi-core Systems

Ghassan Bachay Dharb

Committee Approval:

Dr. Amlan Ganguly *Advisor* Date
Associate Professor - R.I.T. Dept. of Computer Engineering

Dr. Sildomar Monteiro Date
Assistant Professor - R.I.T. Dept. of Electrical and Microelectronic Engineering

---

Dr. Andres Kwasinski Date
Associate Professor - R.I.T. Dept. of Computer Engineering

Dr. Sohail Dianat Date
Department Head - Professor - R.I.T. Dept. of Electrical and Microelectronic Engineering

Acknowledgments

I would like to express my sincere appreciation and gratitude for my advisor Dr. Amlan Ganguly for his constructive remarks, optimal guidance, devoted time, and contributions throughout the duration my research and for the completion of this work. I would Also like to thank Dr. Sildomar Monteiro and Dr. Andres Kwasinski for being as thesis committee members and for their time, effort in supporting my research, and providing pointers to further improve my work.

I would like to thank the Higher Committee of Education Development in Iraq (HCED) for supporting me and my family to complete my thesis and my Master degree. Without their support. My research would not have led up to this thesis.

Finally, I am grateful to the members of the multi-core systems lab at Rochester Institute of Technology especially Naseef Mansoor for their encouragement and support during the work on this thesis.

To my beloved parents, brothers and sisters especially my brothers Raad and Auday who support me and without whom, my dreams of obtaining my Master's degree would not have came into a reality.

To my lovely wife who stands by my side and being a constant source of support toward achieving my goals. Thanks for being part in my life.

To my adorable daughter who motivates me and gives me the hope for the future.

Abstract

The massive integration of cores in multi-core system has enabled chip designer to design systems while meeting the power-performance demands of the applications. However, full-system simulations traditionally used to evaluate the speedup with these systems are computationally expensive and time consuming. On the other hand, analytical speedup models such as Amdahl's law are powerful and fast ways to calculate the achievable speedup of these systems. However, Amdahl's Law disregards the communication among the cores that play a vital role in defining the achievable speedup with the multi-core systems. To bridge this gap, in this work, we present PaSE a parallel speedup estimation framework for multi-core systems that considers the latency of the Network-on-Chip (NoC). To accurately capture the latency of the NoC, we propose a queuing theory based analytical model.

Using our proposed PaSE framework, We conduct a speedup analysis for multi-core system with real application based traffic i.g. Matrix Multiplication. the multiplication of two [32x32] matrices is considered in NoC based multi-core system with respect to three different cases ideal-core case, integer case (i.g. matrix elements are integer number), and denormal case (i.g. matrix elements are denormal number, NaNs, or infinity). From this analysis, we show how the system size, Network-on-Chip NoC architecture, and the computation to communication (C-to-C) ratio effect the achievable speedup.

To sum up, instead of the simulation based performance estimation, our PaSE framework can be utilized as a design guideline i.g. it is possible to use it to understand the optimal multi-core system-size for certain applications. Thus, this model can reduce the design time and effort of such NoC based multi-core systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

As the computational complexities have been increasing, there is a demand to produce powerful computers Such computers used in the fields of astrophysics, bioinformatics, weather forecasting. The processors that have used to run these computers have been developing within time. As per Moore's law, the number of transistors has been increasing in a single chip on a massive scale [1]. The traditional way to run theses processors and achieve better performance is carried out by increasing the operation clock frequency. However, this has reached to a limited point due to the increasing in the power dissipation because the power dissipation is proportional to the operation frequency. As a result, chip designers have shifted to multi-cores systems. With the scaling of CMOS technology, multi-core systems have become the de facto design choice to meet the power-performance demands of the applications. Although, these systems now include as many as eight cores (e.g. Xeon Processors) to hundreds of cores [2], this number is predicted to scale up to 1000 of cores in near future [3].

To study the achievable speedup with small multi-core systems (e.g. tens of cores), full-system simulation frameworks are traditionally deployed. The full-system simulations require detailed modelling of the multi-core architecture such as core archi-

tecture, memory hierarchy and coherency schemes [4]. Furthermore, to capture the interaction between the cores that execute the application in parallel, these simulators also model the interconnection fabric between the cores.The conventional approach to tackle data communication within multi-core system is the global crossbars or the shared bus. This approach scales unsteady With the increasing number of cores in multi-core system. Therefore, the interconnection between the increased cores drives the need to have a modular-efficient interconnection network-on-Chip (NoC) [5]. NoC architectures can be used to provide the communication requirement for the future increasing of cores in which data have routed across these networks by using switches.

Although a bus based interconnect is traditionally used for small scale multi-cores, due to the scalability issues, Network-on-Chip (NoC) architecture are used to provide the communication infrastructure in large multi-core systems [6]. Hence, the full-system simulators account for the latency of the NoC while evaluating the speedup with multi-core systems. However, such full-system simulation is time consuming and computationally expensive. The situation exacerbates as the system size increases to a few hundreds of cores [7]. On the other hand, analytical speedup models such as Amdahl's law [8], and Hill and Marty model [9] are fast and powerful ways for evaluating the achievable speedup without any detailed system level modeling. Unlike the full system simulators, analytical models like Amdahl's law [8] neglect the communication overheads. However, the NoC plays a vital role in defining the performance, power and area of the multi-core systems [7]. Hence, to accurately capture the achievable speedup for the multi-core systems, analytical models should consider the NoC latency while determining the execution time or speedup of the parallel applications. For this reason, in this thesis, we present a Parallel Speedup Estimation (PaSE) framework to analytically model the achievable speedup for parallel systems like multi-core processors while accounting for the NoC latency.

An overview of the proposed PaSE framework is shown in Figure 1.1. The pro-

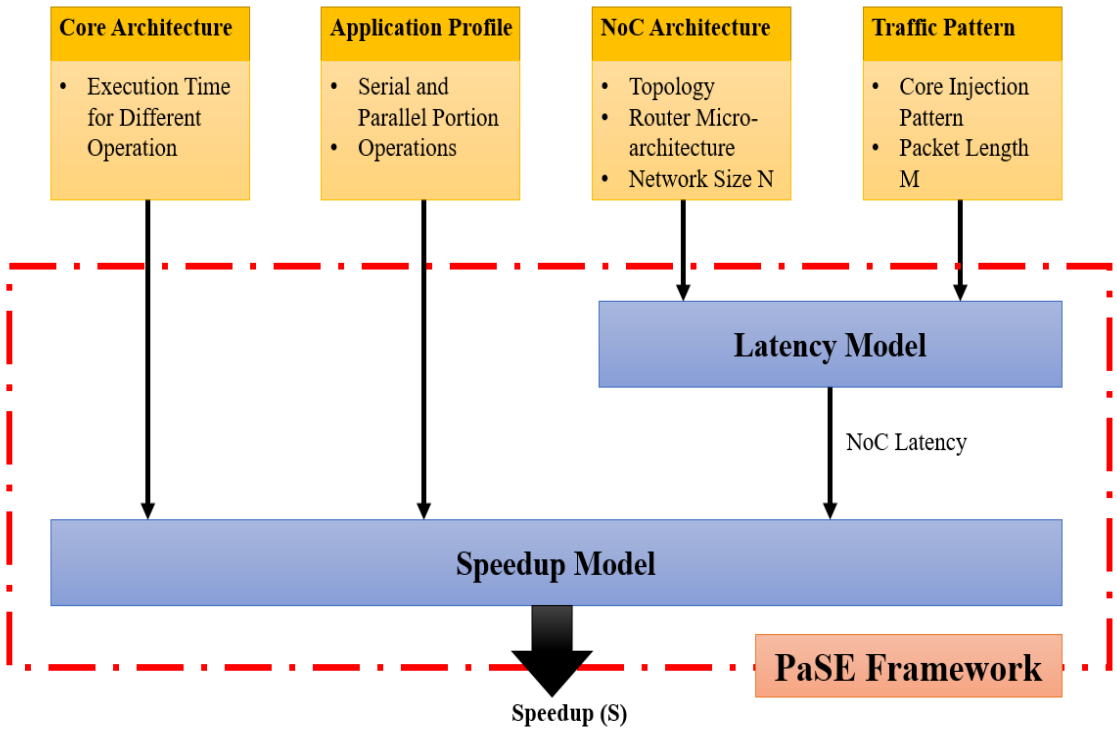


Figure 1.1: Overview of the proposed PaSE framework for Speedup evaluation.

posed PaSE framework contains two models: the network latency model and the speedup model. The models take the application profile, the NoC architecture, the traffic pattern and the core architecture of the multi-core processor as inputs and computes the speedup (S) for the NoC based multi-core system. The latency model determines the NoC latency based on the NoC architecture (i.e. topology, router microarchitecture, and interconnect technology, and network size), and the traffic pattern. In this work, to analytically determine the latency of the NoC, we propose a queuing theory based NoC latency model. To validate the proposed latency model, we compare the latency values from the model with that of a cycle accurate NoC simulator for different NoC architectures and traffic patterns. We then propose an analytical model to compute the speedup of parallel applications on a NoC based multi-core system. As a case study, we present the speedup for a data parallel applications of matrix multiplication. From this study, we domesticate how the speedup is dependent on the NoC topology, System Size, and Computation-to-Communication

$(C - to - C)$ ratio of the application running on multi-core system.

1.2 Thesis Contribution

The following points summarize contributions made during the work of this thesis:

- Proposed network latency model using queuing theory. (M/M/1/K queue model has been used to estimate the value of waiting time at each switch. Then, the total latency of the network has evaluated with the help of calculating the effective hop counts that the message will go through in its path from the source node to the destination node. After that, we explore the proposed latency model with different network topology (Mesh, Folded Torus, 3D Mesh, and Small-world), different traffic pattern (Uniform, Hotspot, and Matrix Multiplication), varying network size, and varying the injection rate of flits per core per cycle.)

- Investigate the validation of the proposed NoC latency Model. (We have run a cycle accurate NoC simulation to validate the results of the proposed model. We compared the simulation and model results with different traffic injection pattern, varying the injection rate of flits per core per cycle, and different NoC topologies.)

- Present a comparison with other NoC latency models. (We compared our proposed latency model with the other NoC latency model that were also derived using queuing. From this comparison, we found that our proposed latency model shows better NoC latency estimation.)

- Present the novel of Parallel Speedup Estimation (PaSE) framework.(PaSE can analytically model the achievable speedup for parallel system like multi-core processors taken into account the network latency which also estimated in the thesis using M/M/1/K queue model.)

- Present a case study using our PaSE framework. (We present a case study of multi-core speedup with data parallel application i.g. Matrix Multiplication. From this study, we shows how the network size, NoC architecture, and computation-to-communication effect the total achievable speedup.)

- Present a comparison with other proposed speedup model. (We compared our PaSE framework with other proposed speedup model. From this comparison, PaSE shows a better results and reassembly closed to the simulation results.)

1.3 Organization of The Thesis

This thesis is organized in five chapters. A brief information about each chapter is mentioned below:

- **Chapter 1. Introduction :** introduces the motivation behind the work that has been performed toward completing this thesis. Then, it states the contribution of this thesis and thesis organization.

- **Chapter 2. Background and Related Work :** explains the trend from single chip to multiple cores systems and Network on-Chips .Then, it describes the previous works that have been done to estimate the speedup after parallelization. It reports most of the works that have been done to extend Amdahl's law. Thus, it eventually states the reason why the work of this thesis is valuable and beneficial in the field of estimating the speed up of the parallel system. Also, this chapter shows list of the previous work that have done to estimate network latency.

- **Chapter 3. The Proposed Latency Model :** describes proposed method to estimate the latency of the network using queuing theory. Then, it shows the analysis for the proposed NoC latency and validation for this analysis. Also, in

this chapter, we show a comparison between our NoC latency model with other NoC latency models that were previously published which were also proposed based on using queuing theory.

- **Chapter 4. The PaSE Framework :** introduces the proposed Parallel Speedup Estimation PaSE along with accounting the network latency. Also, it presents a speedup analysis for multi-core system with real application based traffic i.e. Matrix Multiplication. From this analysis, we show how the network size, NoC architecture, and computation-to-communication effect the speedup.

- **Chapter 5. Conclusion and Future Works :** mentions a brief summary about the work of this thesis and the obtained results. It also point out the future work.

Chapter 2

Background Information and Related Works

This chapter has two sections. The first section describes in details the reasons of emerging Network-on-chip in the multi-core system-on-chip (SoC). The second section has a brief clarification about the related works that have been done to estimate the speedup for parallel systems such as Amdahl's law which is the base for our proposed Parallel Speedup Estimation PaSE. Also, it shows a summary for other works that have been done to estimated NoC latency based on using queuing theory.

2.1 Background Information

2.1.1 Multi-core System On-Chip (SoC).

With the technology's scalability, the number of transistors fabricated on a single chip is going to increase. This is in accordance with Moore's law as it states that the number of transistors in integrated circuits have been doubling every eighteen months [10]. Figure 2.1 shows the increase in the transistors counts with time per Moore's law. Traditionally, a better performance of the processor can be achieved by increasing the operating clock frequency. Based on power formula $Power = V_s^2 FC$ where V_s is the voltage supply, C is the effective switched capacitance, and F is the operation frequency, it is certainly obvious that clock speed is directly proportional to the power consumption. Thus, any increase in the clock frequency rate will lead to

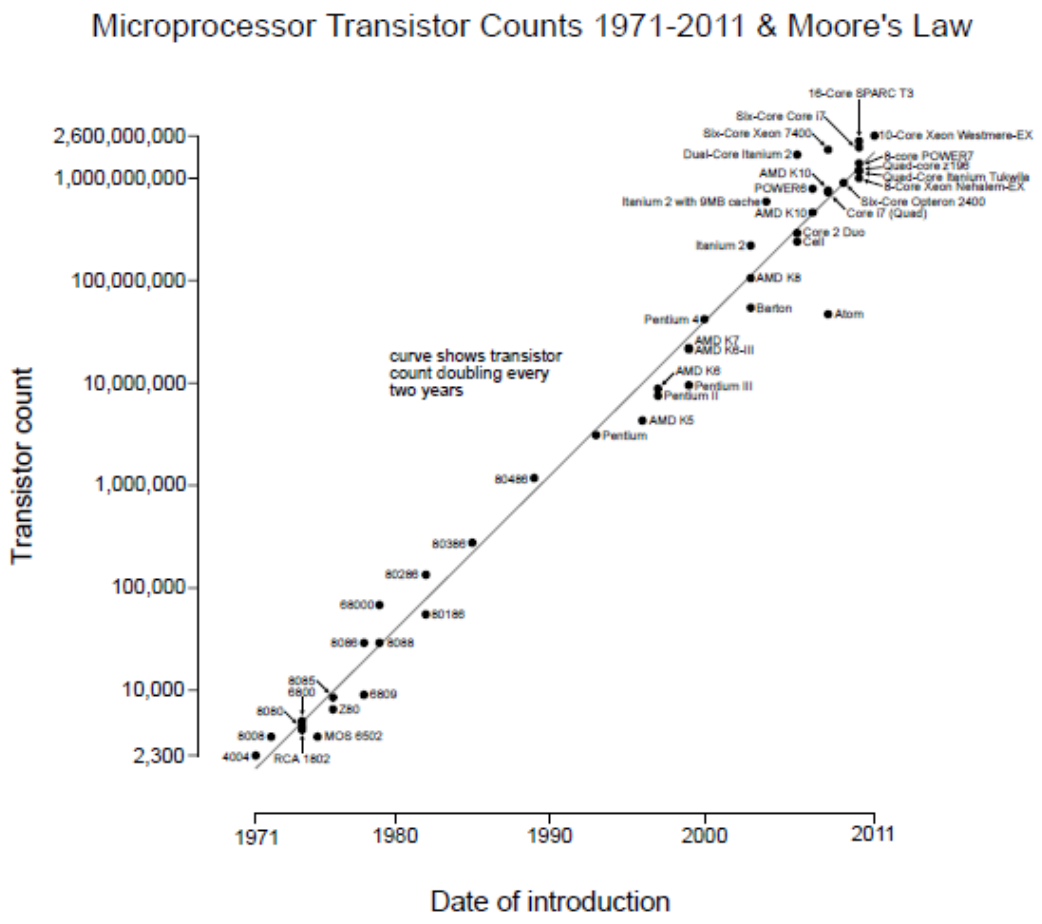


Figure 2.1: Transistors counts against dates of introduction in accordance with Moore's law [1]. The number of transistors in an integrated circuit is doubling every eighteen months.

a massive increasing in power dissipation. Therefore, the increase in operation clock frequency speed to achieve better performance has reached to its boundary limit. As a result, chip designers have examined a promising alternative method that can be used to get better performance with a agreeable power dissipation. These have approved to be obtained by using multi-core chips instead of single core system. The reason for that is because the capabilities of the parallel computing which make multi-core faster while operating at the same clock speed. Furthermore, comparatively, achieving better performance at moderate clock speed make multi-core chips ideal to use with lower power dissipation.

The Traditional architecture to interconnect multiple cores on-chip and achieve

better performance is the shared medium arbitrated bus [11]. There is a couple of bus interconnect architectures that can be used to integrate IP blocks such as ARM AMBA [12], Wishbone [13],and IBM CoreConnect [14]. Unfortunately, these shared bus interconnects does not scale as the system size increases due to the increase in the parasitic resistance and capacitance. Furthermore, adding more IP blocks to the bus can cause in increasing the parasitic capacitance thereby increasing the propagation delay. Therefore, using shared bus in the interconnection between IP blocks is eventually limiting the system scalability [15]. With the high demand for the modern systems which integrate hundreds of IP blocks while maintaining the delay of the global communication, chips designer have explored an alternative way for global interconnection across a chip. Network-on-Chip interconnects paradigm was found to be the promising architecture for global communication within IP blocks.

2.1.2 Network-on-Chip (NoC)

The influential disadvantage of using multi-core chips is the not scalability of the global wire delays. Global wires are used to carry the signals in multi-core chips. These wires does not scale in length as the technology scale, and their delay increases significantly [16]. Their delay can exceed multiple clock cycles even after insertion repeaters. It clearly demonstrated that, in ultra-deep submicron processes, 80 percent of delays of the critical paths are because of interconnections [17], [18].

As the future generation of multi-core system will contain hundreds or even thousands of IP cores integrated on a single chip, the length of the global wire to connects those cores will significantly increase. This has resulted in massive increasing interconnect communication delay and thereby limits the system scalability. Therefore, there is a need to have a scalable interconnection network that can be utilized to integrate more and more IP block cores without limiting system scalability. This can be achieved by using on-chip interconnection network. With this interconnection,

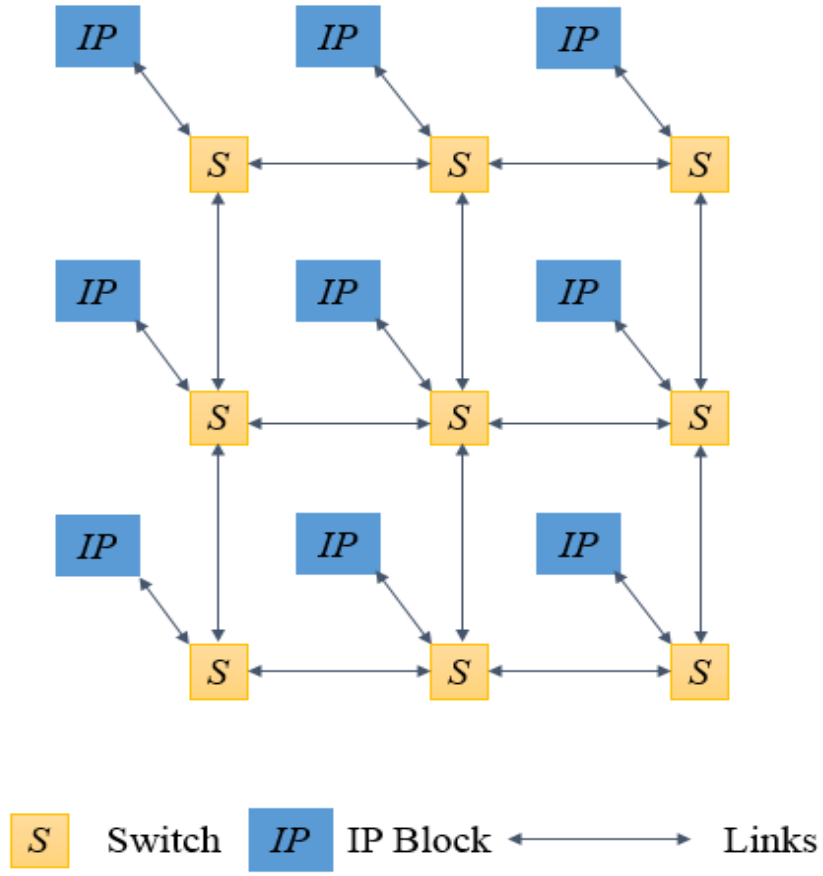


Figure 2.2: Typical Network-on-Chip NoC architecture with Mesh topology.

the IP block cores could separate from the communication network which provides a plug-and-play system. The new paradigm of designing a scalable interconnection network is called Network-on-Chip (NoC) [5].

Figure 2.2 shows a simple NoC architecture in which it can be seen that it consists of the following components: IP cores, switches, and inter-switch links. With NoC system, global wires have replaced with a logic network, and the data is routing across the network through switches and links. For routing theses data, there are different types of switching, namely Circuit Switching, Packet Switching, and Wormhole Switching [19].

Among these switching technique, Wormhole switching has been adapted in this thesis due to the smaller network area and more efficient of network utilization. In

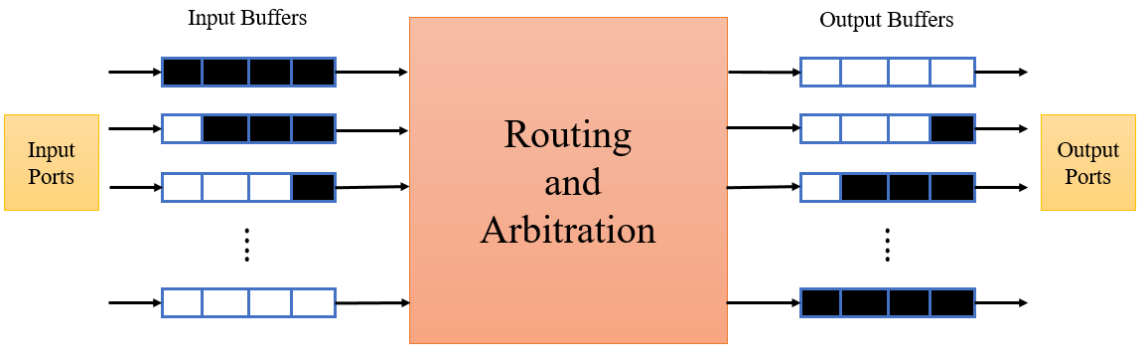


Figure 2.3: Overview of Network-on-Chip NoC switch architecture.

wormhole switching, the packet has broken into smaller units called flits which are header flit and body flit. The size of each flit is determined so that they can traverse between any two adjacent switches within certain clock cycle. The first flit of the packet which is the header flit contains routing information used to establish the path of the entire packet from the source node to destination node. The rest of the flits follows the header flit to their final destination in a pipeline fashion [19]. The path setting up by the header flit may block by other communication of other nodes e.g. the path could be reserved by a particular packet till it is completely transmitted. To overcome this problem, virtual channel buffers have introduced. NoC switches have virtual channel buffers to store the flits until the path is available to send them. Figure 2.3 shows an overview of the structure of NoC switch with input and output virtual channels.

2.1.3 Network-on-Chip NoC Topology

Network topology determines how many hops i.g. routers in the network that the packet must traverse to reach the final destination as well as the interconnect lengths within these routers. Thus, Selecting NoC topology plays a significant role in determining the system performance. In last few decades, many NoC topology has been explored. They can be classified as direct or indirect topologies. In direct

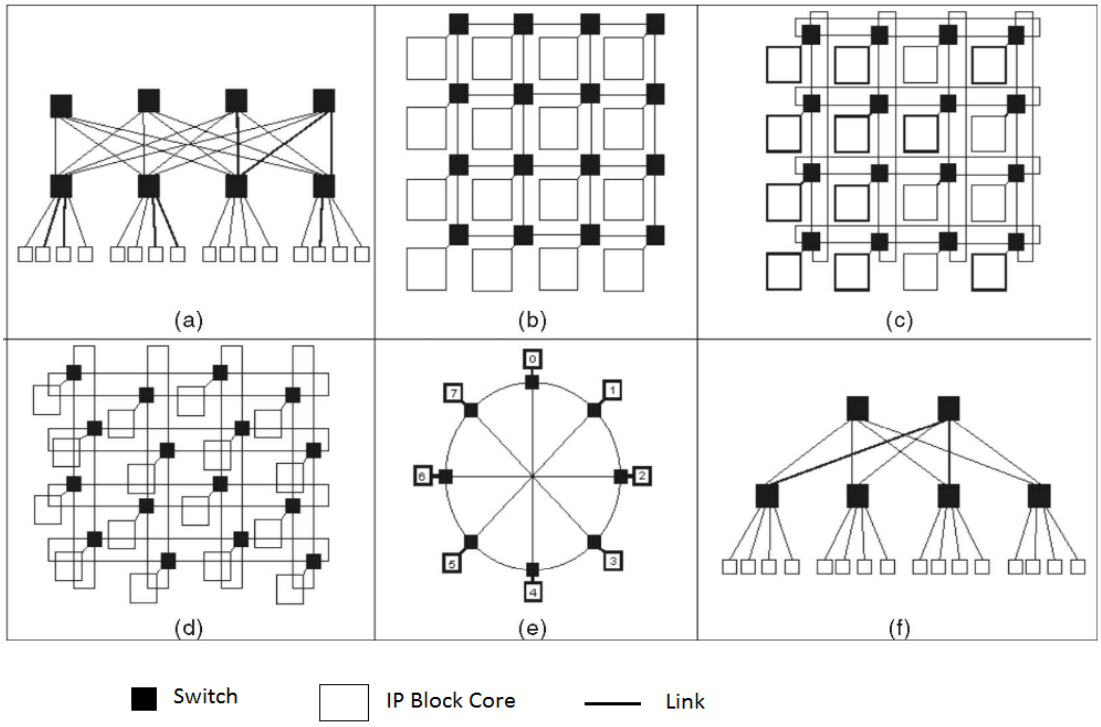


Figure 2.4: Network on-Chip (NoC) topology architectures [54] where (a) and (d) are examples of indirect topologies while (b), (c), (d) and (e) are examples of direct topologies.

topology, every router or switch is directly connected to end node which is the source or destination of packets. In indirect topology, nodes could serve as a direct IP core with an associated router or intermediate node help transferring the packet across terminal nodes. Figure 2.4 shows direct and indirect NoC topology.

Among these topologies, Mesh and Folded Torus topologies have been considered in this thesis for experimentation due to their widespread use in NoCs. Mesh topology has been proposed in [20] as NoC architecture in which this design consists of m x n switches interconnecting IP blocks cores. The number of switches in this topology is equal to the number of IP blocks. Every switch is connecting to four neighboring switches, and one connects local IP block except the ones at the edge. Furthermore, the links between switches or between switch and IP block consist of two unidirectional interconnects. Another topology has been considered in this thesis is Folded Torus topology [21] as NoC architecture. Folded Torus topology is similar to the Mesh

topology expect that it has a wrap around links in the switches at the edge. This has resulted in having five ports for every switch, four to connect with the neighboring switches and one to the local IP blocks. Mesh and Folded Torus both are example of NoC regular topology. In addition to these regular topologies, we considered in this work irregular NoC topology such as Small-world networks in [22].

2.1.4 Emerging Interconnects

As the technology continues to scale down, the interconnects wire are getting thinner and thinner resulting in increasing its resistance. The increase in resistance can lead to significant increase in power dissipation and latency. According to the International Technology Roadmap for Semiconductor (ITRS), interconnects are the major bottlenecks to overcome the power-performance barrier in the future generation. Increasing in power consumption can lead to a high on-chip temperature which in turn compromises the performance and reliability of the chip [23]. Hence, it is clear that the challenges facing future chips are the scalability of on-chip interconnects.

Because of that, different interconnect technologies has been proposed to improve the performance of the traditional NoC interconnects such as three-dimensional integration, photonic interconnects, and multi-band RF (wireless) interconnects [24], [25]. These new approaches are considerd the promising paradigms that are capable of improving the power dissipation and the performance of the NoC design.

In three-dimensional integration interconnects, multiple active layers have integrated onto a single chip. The main advantage of this interconnects is that it reduced the hop counts due to the reduction in length and number of global interconnects. Another benefit of 3 D integration of chips is that two different technologies can be connecting with each other. However, due to the small footprint, the power density of 3-D interconnects would be high thereby high-temperature dissipation [26] which in turn requires cooling mechanisim [27]. Furthermore, fabricating 3D chips has sets

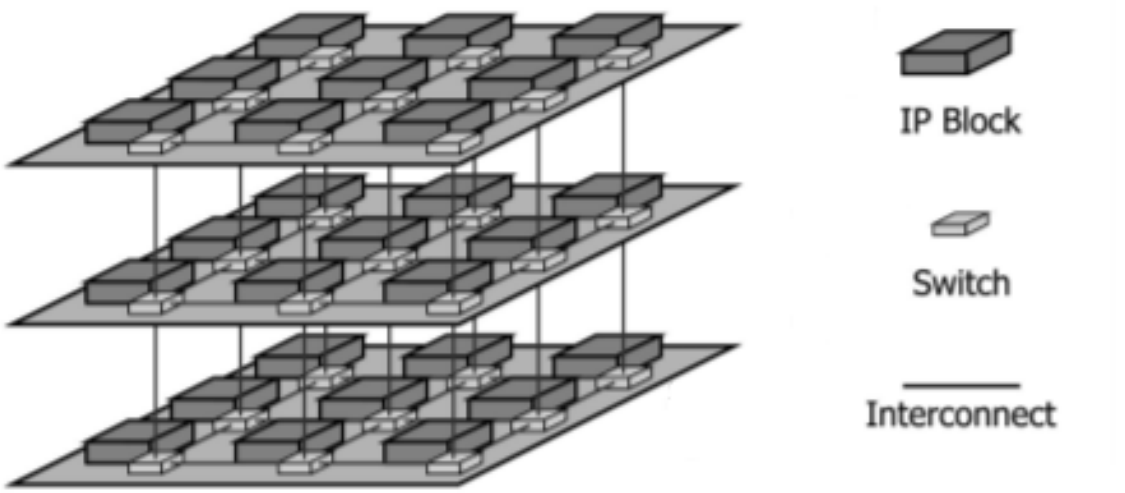


Figure 2.5: Network-on-Chip NoC 3D Mesh based architecture [34].

of issues with inter-layer alignment, bonding, and inter-layer patterning [26] resulting in high risk of manufacturing defects.

In photonic interconnect technology, optical interconnects are used instead of traditional global interconnect to transmit data within switches [28]. The transmission of data is carrying at the speed of light. Therefore, photonic interconnects has an advantage of having low latency with a high bandwidth. Also, due to the extremely data loss, optical interconnects is considered a reliable data transmission. The disadvantage of this interconnects is that it is hard to integrate photonic interconnects with silicon devices.

In wireless interconnects, switches are equipping with a wireless link that contains antenna and transceiver [29]. Using this interconnect, different NoC architectures has been proposed [30], [31].This has lead to transmit the traditional multi-hop interconnects with a single-hop long distance shortcut. Several recent studies have found that wireless links can significantly improve chips performance and better power consumption [32], [33]. The issue of wireless interconnects is that the fabrication of carbon nano tube antennas faces sets of problems much higher that CMOS process.

As an example of using an emerging topology, in this thesis, we have considered

3D Mesh topology [34]. With this topology, every switch has seven ports four to the coordinated direction e.g. for North, South, East, and West, one to the local core, and two to the above and below connection with other cores. 3D Mesh topology can be seen in Figure 2.5.

2.1.5 Network-on-Chip NoC Traffic Pattern

NoC traffic pattern plays a significant role in evaluating NoC performance because it indicates the destination core for the message in the network. In this thesis, we have consider two type of traffic patterns. The first traffic pattern is the most frequently used in NoC network which is Uniform distribution of packets [35]. In this traffic, any node in the network can send a packets to all other network with equal probability i.g. any core in the network have equal probability for being the destination for the packets sending from node i. This probability does not include the probability for any node in the network that sending packets to them self due to the fact that if that happened, packets transfer will not use the network thereby it does not have any effect on the network latency. Secondly, we have considered two nonuniform traffic pattern Hotspot [36] and Matrix Multiplication. In Hotspot traffic pattern, all nodes in the network send packets to one single node i.g. there is one master core in the network who receives packets from all other core. Sometimes the hotspot core receives X% of packets from other cores while the remaining portion of packets distributed uniformly with the other network cores. With Matrix Multiplication Traffic pattern, packets follow the behavior of the normal multiplication algorithm of two matrices in their choosing the destination core in the network.

2.2 Related Works

When designing a certain NoCs, there is a trade-off between multiple competing metrics such as lowering power consumption, minimizing packets transferring latency,

and reducing area overhead [37]. These NoC parameters have typically evaluated from doing simulations based analysis. Also, to evaluate the achievable speedup, full system simulation is typically executed which takes into account the latency of the NoC. Getting the performance of NoCs and the speedup based on simulation analysis is considered a time consuming especially when the size of the network is increasing. Therefore, it is more significant to have an approach that can be used to know the design metrics earlier before the simulations have undertaken.

Analytical based models are considered as an alternative approach to simulation-based analysis to estimate NoCs performance metrics and the performance gain. In general, any analytical model is built based on particular theoretical assumption. The validity of these models are relied on how this theoretical assumption have taken from the real world. Therefore, by understanding NoCs topology scenarios, an efficient and reliable analytical model can be created to calculate the performance metric. This can save chip designers a considerable time to reasonably estimate the overall performance metrics. Thus, chip designers could focus more and more to come up with feasible designs.

In the next subsection, we will state some of the previous works that has been done to estimate the achievable speedup and the NoC latency models.

2.2.1 Amdahl's Law

In 1967, Gene Amdahl defined his law to predict the theoretical speedup of the performance improvement in parallel computing [8]. Amdahl's law states that if the portion of any execution of a specified task P can be parallelized by a factor N, and the remaining part can not be improved and remain entirely sequential, Then, the part that is not getting changed of the task would dominate the overall performance. Even after excessive improvement of the reform part of the computation task, the effect of this amelioration will have a little effect and have resulted in the sequential

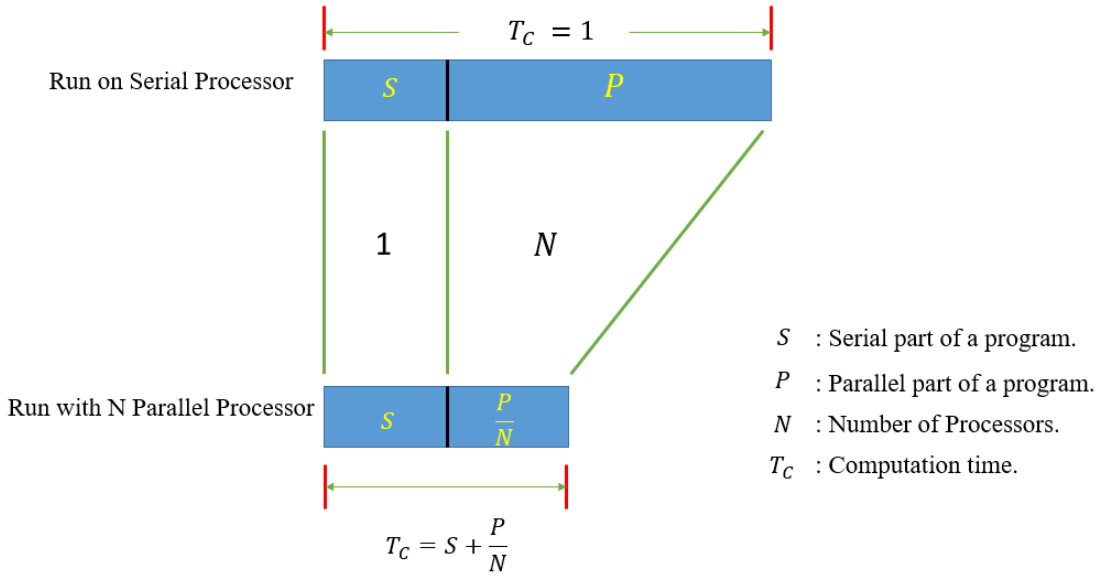


Figure 2.6: Amdahl's law clarification in parallel computing.

part dominance of the gain performance.

Amdahl's law is established as a particular case of using N processors in parallel instead of single-processor utilization for achieving large-scale computing capabilities. Assuming that P be the portion of the task that can be parallelized, N is the number of processors, and S is the part that remains entirely sequential, Then the processing speedup by Amdahl's law on N processors declares in equation (2.1). This equation resulted from the definition of the speedup of a task execution in parallel computing since it is defined as the sequential execution time over the parallel execution time. Figure 2.6 shows a clarification of Amdahl's law.

$$Speedup = \frac{1}{S + \frac{P}{N}} \tag{2.1}$$

Several significant facts can be revealed from Amdahl's law. One fact is that when the portion of the task that can be executed in parallel is small, optimization will have a little impact on the speedup. Furthermore, the sequential part of the execution is limiting the speedup even if the number of the processors approach

infinity. This means that the increase in the speedup of any implementation of a task will be bounded by the $1/S$. Amdahl's law is playing a fundamental role and serves as a guideline in the parallel computing. However, it is not suitable to apply it for the modern Network-on-Chip NoCs architectures. This is because the analysis of the network communication has not involved in Amdahl's law. If it is incorporated with this model, it become more precisely to predict the speedup on multi-core system.

2.2.2 Extension of Amdahl's Law in Multi-core System

Amdahl's law has been further analyzed in [38] regarding the scalability of multi-core architectures. Authors come up with a three speed up models, fixed-size model, fixed-time model, and memory-bounded model. These models analyzed based on the symmetric multi-core architecture in which all cores are identical. Although Amdahl's law state that speedup metric will be fixed by the sequential part of the architecture, the theoretical analysis of their models demonstrates that multi-core architecture is scalable for the large-scale parallel processing. This from the scaled computing view can provide an insight into the designing of a large-scale multi-core architecture and break down the view of limited scalability of multi-core architectures in the industry. However, the three speed up models are not considered the detail of the interconnection between cores.

Another extension for Amdahl's law studied by Gustafson in [39]. In this work, the author investigated the assumption of fixed size problem which was assumed by Amdahl's law and argued that it is a run-time problem. This hypothesis is because of in almost all application, the need for more cores is required to solve large complex calculation. Thus, the parallel fraction of total computation of the tasks will get increased and scale linearly with the number of cores. In [40], authors formulated a Generalized Scaled Speedup Equation (GSSE) which incorporates both of Amdahl's and Gustafson law. Then, Authors applied both of Amdahl's law, Gustafson law,

and GSSE to the area performance model developed by Hill and Marty in [9] with symmetric, asymetric, and dynamic multicore architectures and found that different results obtained. However, authors assumed that the performance has limited by the area. Therefore, many constraints can be incorporated into to extend this model such as power, workload behavior, and how the parallel fraction scaled with problem size.

Hill and Marty in [9] extend Amdahl's law by constructing a simple model that can be utilized to calculate the overall speed up for different multi-core chip architectures: symmetric core, asymmetric core, and distributed core systems to the use of one single core. This extension has been fulfilled by adopting Amdahl's law to generate a distinct model employed for each system of multi-core. The proposed models have been playing an important role in the parallel computing and multi-core area. However, the models that Hill and Marty come up with have some weaknesses. This is because the communication latency resulting from the developing NoCs network is not considered in those models. In spite of the limitation in those models, it certainly clear that this work has successfully encouraged multi-core designer to investigate the entire performance of the chip instead of focusing on the core abilities.

Hill's and Marty's work in [9] is used over the years by researchers as the basis for modeling new equations to predict the performance of future multi-core architectures. For example, Yao, Bao, Tan, and Chen in [41] have theoretically investigated Hill and Marty work to come up with a general framework models. These comprehensive models can be used by multi-core designers to determine the overall performance architecture. Despite this enormous effort, Authors have not derived such a powerful model because many performance factors have been removed from the model including the communication cost by the system.

In [42], authors investigated Amdahl's law to obtain an analytical expressions for the optimum frequency, voltage supply, and energy. However, none of these proposed model consider the communication latency between the cores in a multi-core system.

In [43], authors proposed a speedup model considering the communication latency in a multi-core system. However, authors considered a simple latency model which neglects the contention delay in the NoC routers. Hence, in order to capture the effect of the NoC in multi-core speedup it is necessary to accurately estimate the NoC latency. Therefore, we proposed PaSE framework to analytically estimate the speedup for parallel systems such as multi-core taken into consideration the accurate estimation of the network latency.

2.2.3 Network-on-Chip NoC Latency Models

Table 2.1: Comparison between the proposed latency model and lists of the previous latency models that derived based on using queuing theory.

Queue type	**Arrival**	**Service**	**Buffer size**	**Reference**
M/M/1	Poisson	Exponential	Infinite	[44]
M/G/1	Poisson	General	Infinite	[45]
G/G/1	General	General	Infinite	[46]
M/G/1/K	Poisson	General	Finite	[47]
G/G/1/K	General	General	Finite	[48]
M/M/1/K	Poisson	Exponential	Finite	Proposed in this work

Many latency models have been proposed in recent years to estimate the latency of NoC architectures. In [49] authors have proposed a machine learning technique based NoC latency model called SVR-NoC. Although, such work is unique and shows promising results, the large training set required to precisely calculate the latency for different NoC architectures and traffic patterns is difficult to generate. Consequently, many of the NoC latency models are based on queuing theory. These works can be broadly classified in two categories: i) infinite buffer capacity queuing systems and ii) finite buffer capacity queuing systems. For example, in [44], [45], and [46], authors proposed NoC latency models considering M/M/1, M/G/1, and G/G/1 queues re-

spectively with infinite buffer capacity. However, the number of virtual channels in a NoC router is limited (i.e. finite buffer capacity) due to power and area constraints. On the other hand, authors in [47] and [48] proposed M/G/1/K queue and G/G/1/K queue based latency model with finite buffer capacity. However, in the NoC routers, the arrival and departure of packets follows a Poisson distribution [50]. Consequently, the service time in the NoC routers should follow an exponential distribution as the time interval between Poisson events are characterized to be exponential. Hence, assuming general distribution for the service time will result in limiting the accuracy of the latency model. Therefore, to accurately capture the NoC latency, in this paper, we propose an analytical model based on M/M/1/K queuing systems. A comparative difference between the latency model proposed in this work and existing works is presented in Table 2.1. Our proposed latency model is formulated based on the following: Poisson distribution of the arrival rate of flits, exponential distribution of router service rate, and finite buffer size. Then, using this latency model we propose a framework for evaluating the speedup of multi-core systems.

Chapter 3

The Proposed Latency Model

The proposed latency model utilized to analytically determine the average latency for the NoC based multi-core systems. This model has been estimated using queuing theory. Since the queuing theory is the mathematical study for the queues in which it can be utilized to estimate the waiting time in the queuing systems as well as the queue length , there are some notations and assumption should take into consideration to tackle NoC latency with this theories. Therefore, In this chapter, we started by discussing the basic assumption and an introduction to queuing theory along with the proposed latency model in details. After that, we used our model to estimate the packet latency for different NoC topologies, system sizes, and injection rate of flits per core per cycle. Then, we validate our result with those from running cycle accurate NoC simulator. We also show a comparison with the recent NoC latency works that have been proposed using queuing theory.

3.1 Basic Assumptions and Notations

In this work, we consider multi-core system to be symmetric (i.e. all the cores are identical). Each core is considered to be connected with a NoC router through a network interface. The NoC routers implement shortest path based deterministic routing algorithm [51] along with virtual channel (VC) based wormhole flow control mechanism [19].

The communication among the cores occurs in form of packets where the packets are divided into multiple flow control units (flits). The header flit (i.e. first flit of the packet) contains the routing information and the body flits simply follows the path set by the header flit. The number of bits in a flit (i.e. flit size) is considered such that they can be transmitted between two adjacent routers in one clock cycle. Moreover, we consider a constant packet size represented as M in Table 3.1 along with other notations used in this thesis.

3.2 Queuing Theory

Queuing system can be described as customers are arriving to server to get service. They might spend some time waiting at the queue if the server are busy with serving other customers. Then, customers are leaving the queue after getting service. Queuing theory is a mathematical description for such systems in which we can use it to solve the queue problem and get system performance. Thus, It can be used to predict the queue length (i.e. how many customers are waiting in the queue to get service) and the waiting time in the queue.

To describe any queuing system, Kendall's Notations [52] have been utilized. With this notations, any queuing system is specified based on six parameters in the form A / B / m / K / n / D. The definition for each single notation is as following

- A : the distribution of the arrival rate to the queuing system.
- B : the distribution of the service rate of the servers.
- m : number of servers.
- K : system capacity, the maximum number of customer that the queuing system can acquire.

Table 3.1: List of notations used in this thesis.

Parameter	**Explanation**
L_t	Average Network Latency.
L_h	Average latency of the header flits.
L_b	Average latency of the body flits.
M	Message Length.
C	Topology of the NoC.
N	Network size.
N	Expected steady-state number of packets in the buffers.
K	Buffer size.
ρ	Traffic intensity for each router.
λ	Arrival rate of packet.
μ	Service router rate.
P_n	The probability that n flits in the buffers.
$T_{(i,j)}$	Traffic injection pattern.
h_{ij}	Length of the path from node i to node j.
S	Serial subtask portion of the program.
P	Parallel subtask portion of the program.
T_c	Computation time of the task.
R_p	Router pipeline stage.
O_I	Number of instruction I.
C_I	Cycles for the instruction I.

- n : population size, it states whether the system service infinite or finite number of customer.
- D : service discipline, it shows the order or the rule that next customer will receive service. The most common service discipline is First in First out or FIFO.

Queuing theory has been utilized in this thesis to estimate the waiting time W_t for the packet in the input virtual channels VCs including the service time by the router. This will be explained in detail in the calculation of the NoC latency in next section.

3.3 Latency Model

Based on the NoC architecture and the traffic pattern, the latency model determines the average time elapsed for the packets to traverse from the NoC router connected to the source core to the NoC router connected to the destination core (i.e. packet latency). As packets are divided into header and body flits, the packet latency contains both the delay for the header flit and body flits. The packet latency L_t can be calculated based on.

$$L_t = L_h + L_b \tag{3.1}$$

Where, L_h is the total time for the header flit to traverse from the NoC router connected to the source core to the NoC router connected to the destination core, which includes the time spent at the all intermediate routers in the path. This is alternatively known as path-discovery latency. Therefore, to determine the average time for the header flit, both the waiting time at the routers as well as the effective number of hops H_e in network is required. The effective number of hops can be determined from the NoC topology as shown in next subsection. Therefore, the

latency of the header flits can be calculated by the following equation.

$$L_h = (W_t + R_p) * H_e \tag{3.2}$$

W_t is defined as the service time for the header flit including the waiting time in the queue (i. g. it includes both of the waiting time at the buffer and the service time by the routers). Also, because of the R_p-stage pipelined router design, n cycles have been added to the W_t for fully service the header flit. Due to wormhole switching, once the header flit is transmitted, the body flits will follow it in a pipeline fashion. Therefore, the latency for the body flits with a packet size M is given by

$$L_b = M - 1 \tag{3.3}$$

As a result, the only unknown parameters that required to calculate the total average network latency for the message are W_t and H_e. Both are estimated in the following subsection.

3.3.1 Determination of Expected Waiting Time

The expected waiting time W_t for the header flit is the time that it must wait in the input VCs before it is routed to the output VC of a NoC router. To estimate this time, we have used Queuing Theory.

To estimate this, we model each router as a queue with finite buffer capacity. The total buffer capacity K_{total} is equals to the summation of all the input VCs sizes K in all ports. We adopt an M/M/1/K queuing model [53] that assumes Poisson distribution for the arrival rate of packets and exponential distribution for the service rate. According to the merging property of the Poisson distribution, the total number of arrival rate of packets λ_{total} is equal to the summation of all the incoming packets from all input router ports. The representation of the NoC router to the M/M/1/K

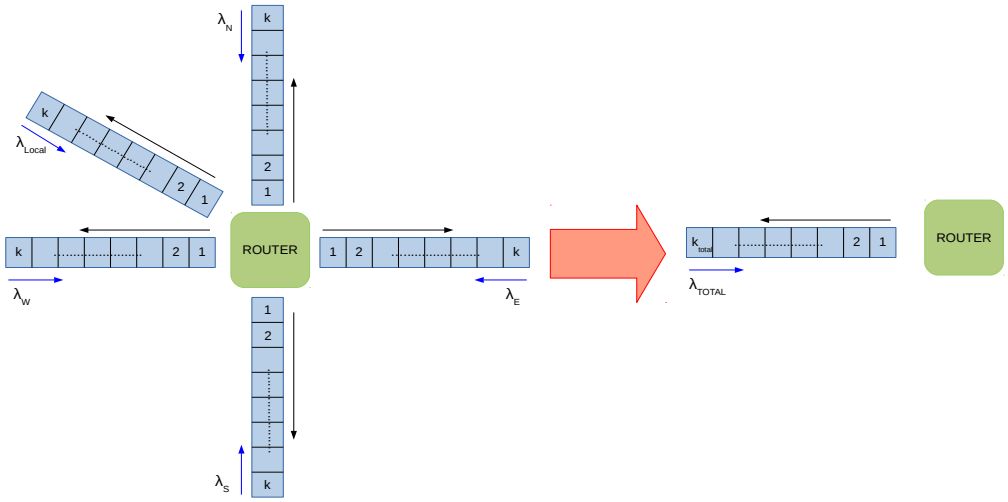


Figure 3.1: The simplification of the NoC router to the M/M/1/K queuing model.

queuing model is shown in Figure 3.1.

As shown in [50], both the arrival and departure of packets follows a Poisson distribution. Hence, our assumption of Poisson arrival for the header flits is valid. On the other hand, the service time between two Poisson events are characterized to be exponential. Hence, considering an exponential service time for the queuing system can accurately model the NoC router behavior.

Using the M/M/1/K queuing model, with λ Poisson arrival rate of the header flits and μ exponential service rate, the expected waiting time in the queue with size K can be calculated using the following equation

$$W_t = \frac{\bar{N}}{\lambda(1 - P_K)} \tag{3.4}$$

Where, $\bar{N}$ is the expected steady state number of the header flits in the queue and P_K is the blocking probability (i.e. steady state probability of having K header

flits in the queue). Hence, the waiting time is dependent on the flits already waiting in the queue and the arrival rate of the header flits. The steady state number of the header flits in the queue, $\bar{N}$ can be calculated using the following equation

$$\bar{N} = \sum_{k=1}^{K} k\rho^k P_0 \tag{3.5}$$

Where, K is the number of buffers in the queue, ρ is the traffic intensity and P_0 is the probability that the queue being empty. The probability of having n flits in the queue can be calculated using following equation,

$$P_n = \begin{cases} \frac{(1-\rho)\rho^n}{1-\rho^{K+1}}, & \text{if } \lambda \neq \mu. \\ \\ \frac{1}{K+1}, & \text{if } \lambda = \mu. \end{cases} \tag{3.6}$$

Where, the value of n is any whole number between *0* to Ke.g. the probability of having *0* header flits in the queue to K e.g. the probability for the queue being full. On the other hand, the traffic intensity ρ can be calculated using the following equation,

$$\rho = \frac{\lambda}{\mu} \tag{3.7}$$

It can be seen from the equation (3.7) that the traffic intensity depends on the arrival rate and the service rate. The service rate, μ is a function of the NoC router and can be obtained from the NoC router specification [45]. Alternatively, the arrival rate λ depends on the application traffic injection pattern. Given traffic injection pattern of the application, we can got $T_(i, j)$ as

$$T_{(i,j)} = \begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,3} & \dots & t_{1,N} \\ t_{2,1} & t_{2,2} & t_{2,3} & \dots & t_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{N,1} & t_{N,2} & t_{N,3} & \dots & t_{N,N} \end{bmatrix}$$

Where, $t_{(i,j)}$ is the injection rate in packets/cycle from core i addressed to core j. Hence, $\sum_{i=1}^{N} t_{i,j}$ is the total rate at which packets addressed to core j are injected in the network from all cores. Therefore, λj is the arrival rate at which packets can arrive at core j. For any traffic pattern, we calculate the arrival rate, for all routers. Then, the overall arrival rate is calculated as the average of all arrival rates and is given by,

$$\lambda = \frac{\sum_{j=1}^{N} \lambda_j}{N} \tag{3.8}$$

Where, N is the number of NoC routers in the network. Hence, in order to determine the waiting time we need the router service time which can be found in router specification and the arrival rate which is a function of the traffic pattern. In the next subsection, we discuss about the determination of the effective number of hops that is used to calculate the latency of the header flit following equation (3.2).

3.3.2 Determination of Effective Number of Hops

Effective number of hops in a NoC is the expected number of intermediate router that a packet has to travel through during its way from the source to destination as it can be seen in Figure 3.2. Therefore, the effective number of hops H_e can be calculated using following equation for any NoC architecture

$$H_e = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} h_{i,j}}{N^2 - N} \tag{3.9}$$

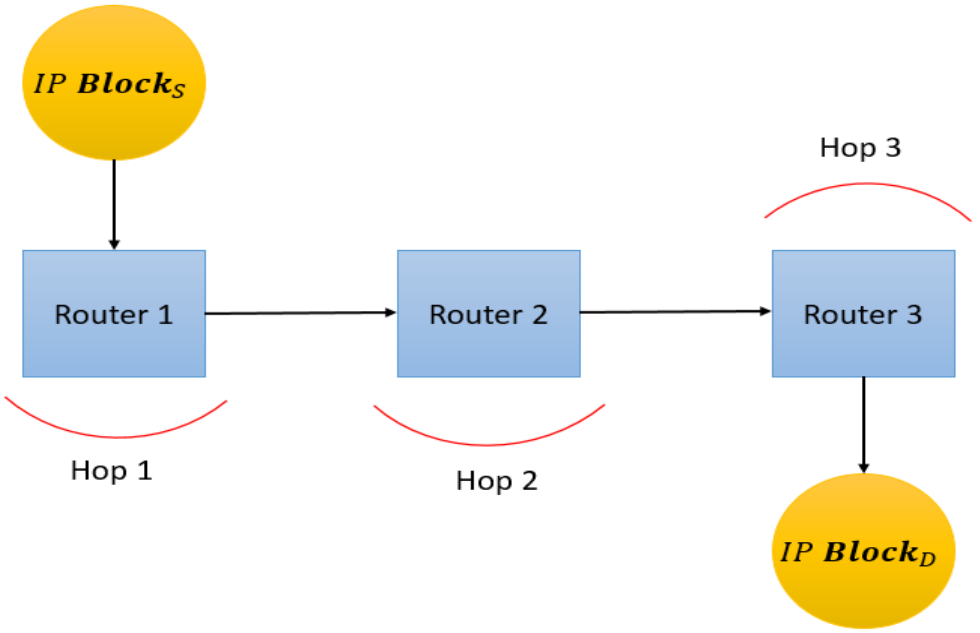


Figure 3.2: Effective number of Hops from the source node to the destination node. This figure shows that packets need to travel three Hops (routers) from the source node($IPBlock_S$) to the destination node($IPBlock_D$).

Where, N is the number of cores in the system and h_{ij} is the number of hops in the shortest path between core i and core j. This equation is used to calculate the effective number of hops with any regular topology (in our case, we use it with mesh and Folded Torus topologies). For 3D Mesh topology, we have the equation (3.10) from [34] as below.

$$H_e = \frac{n_1 n_2 n_3 (n_1 + n_2 + n_3) - n_3(n_1 + n_2) - n_1 n_2}{3(n_1 n_2 n_3 - 1)} \tag{3.10}$$

It can be seen from the H_e equations that the effective number of hops is different for different network topology with same system size due to different in h_{ij}. Also for the same network topology, as the number of cores increases, the effective number of hops also increases yielding a higher packet latency following equation (3.2). The outcome of this latency model will be used in the speedup model PaSE which will be discussed in next chapter.

3.4 Results and Analysis for the Proposed Latency Model

Using the proposed M/M/1/K queuing based latency model, we evaluate latency for regular NoC topologies like Mesh, 3D Mesh, and Folder Torus as well as for irregular topologies like small world networks. The small world topology design methodology is adopted from [22] which is characterized by many short and a few long distance links. We also consider different system sizes and injection rates in our latency evaluation. Both uniform random spatial traffic pattern and non-uniform traffic pattern such as hotspot and matrix multiplication are used in these evaluations. In the uniform random traffic pattern, each core can address packets to any other core with equal probability. On the other hand, for hotspot traffic pattern, one core (core 1) is assumed to be a hotspot core and 10% traffic from all cores are addressed to the hotspot core. The rest of the traffic is uniformly distributed among all cores. For matrix multiplication traffic pattern, packets follow the behavior of the normal multiplication algorithm of two matrices in their distributing and choosing the destination core in the network. We adopt the three stage pipelined NoC router architecture from [54] with wormhole switching [19]. Each port of the NoC router is considered to have 2 VCs (one input and one output) with buffer depth of 8 flits which is same as the size of a packet. We also consider the width of the interconnect to be same as the size of the flit and one flit can be transferred between two adjacent routers in one cycle. Although the routing logic in NoCs is typically dependent upon the topology here we assume the shortest path routing as most deterministic routing strategies converge to the shortest path routing regardless of topology.

To validate the latency model, we compare the latency result from our model with a cycle accurate NoC simulator. The NoC simulator characterizes the NoC architecture and models the progress of the flits over the NoC routers and links per cycle accounting for those flits that reach the destination as well as those that are stalled.

For this comparison, in the simulations we considered the same NoC architecture configuration in terms of NoC router pipeline, switching mechanism, link latency, bandwidth and topology as in the model. Ten thousand iterations were performed eliminating transients in the first thousand iterations to capture steady-state characteristics modeled in our framework. Also, multiple simulations are performed in order to collect relevant averages for the comparison.

the analysis for the proposed Latency model and the validation for the result will be presented in the next subsection.

3.4.1 Analysis of the NoC Latency Model

In this section, we evaluate the packet latency for different NoC topologies (i.e. Mesh, Folded Torus, 3D Mesh and Small World) using the proposed latency model. We evaluate these topologies for system size of 2 x 2 (4) core system, 4 x 4 (16) core system, 8 x 8 (64) core system, 16 x 16 (256) core system, and 32 x 32 (1024) core system. It can be seen from Figure 3.3 which is with uniform traffic pattern, Figure 3.4 which is for hotspot traffic pattern, and Figure 3.5 which is for matrix multiplication traffic pattern that for all NoC topologies the latency increases with increasing system size. This because with increasing system size the average effective number of hops that the packets pass through also increases. Hence, a shorter path between the source and destination cores will result in better latency as in the case with smaller system sizes. This can be further verified when we compare the latency for different NoC topologies with same system size. Among the different NoC topologies the small-world has the lowest effective number of hops due to the long-range shortcuts (i.e. a property of small world networks). Due to this reduced effective number of hops, the small-world has the lowest latency for all system sizes. For example, for a system size of 16 x 16 (256) core system, the effective number of hops for the small-world topology is 6.05 compared to 6.52549 hops in a 3D Mesh. Due to this, reduction in

effective number of hops, the latency of the small-world network is lower than that of the 3D Mesh and all other topologies compared in this paper. Due to this also the small-world network saturates at much higher injection load for the same system size. The injection load at which the NoCs saturate can be found where the latency plots start increasing drastically.

3.4.2 Validation of the Proposed Latency Model

In this subsection, we validate our proposed latency model with simulation based latencies. We compare the latency of an 8x8 (64) core system with uniform, hotspot, and matrix multiplication traffic pattern with varying injection load. The latency for different NoC architectures with these traffic patterns are shown in Figure 3.6, Figure 3.7, and and Figure 3.8. It can be seen from the figures that, for all NoC topologies, the latency model closely resembles the simulation latencies with negligible error of less than 2% with low injection rates. One interesting observation from those plots is that, after saturation, when the latency increases drastically, the latency obtained from the model latency is lower than that from the simulation. Beyond this level of injection load, the NoC is saturated which means that the NoC switches experience high levels of congestion resulting in sharp increase in latency.

However, the throughput of all the NoCs given by our model is same as that of simulation results. The maximum injection rate that can be sustained by the NoC or its maximum throughput, can be derived from the latency plots by observing the injection load at which the latency has a sharp increase. It is interesting to note that the throughput of all the NoCs given by our model is same as that of the simulation results. So, the error in throughput of our model with respect to simulations is negligible.

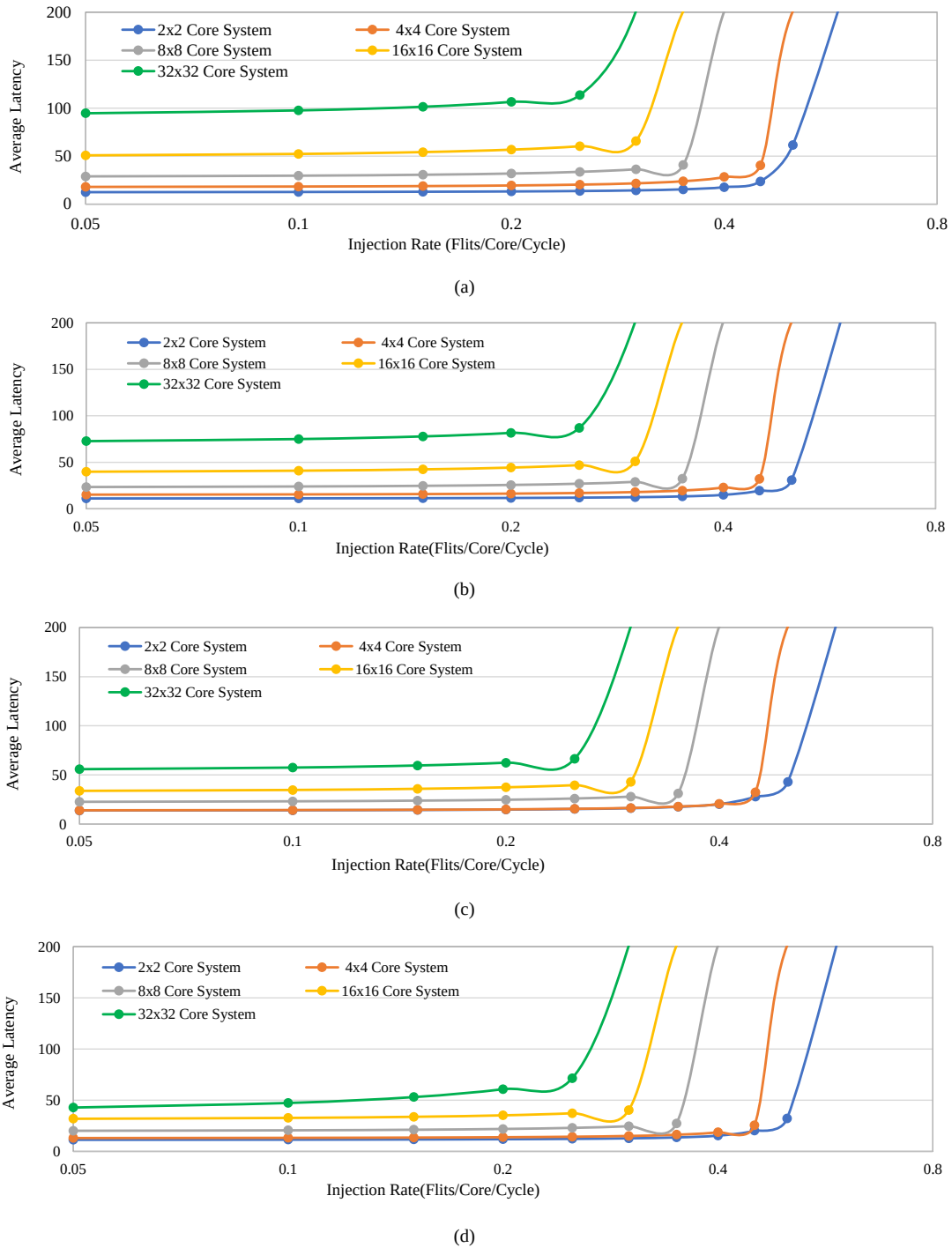


Figure 3.3: Packet latencies for (a) mesh, (b) folded torus, (c) 3D mesh, and (d) small-world topologies with uniform traffic pattern.

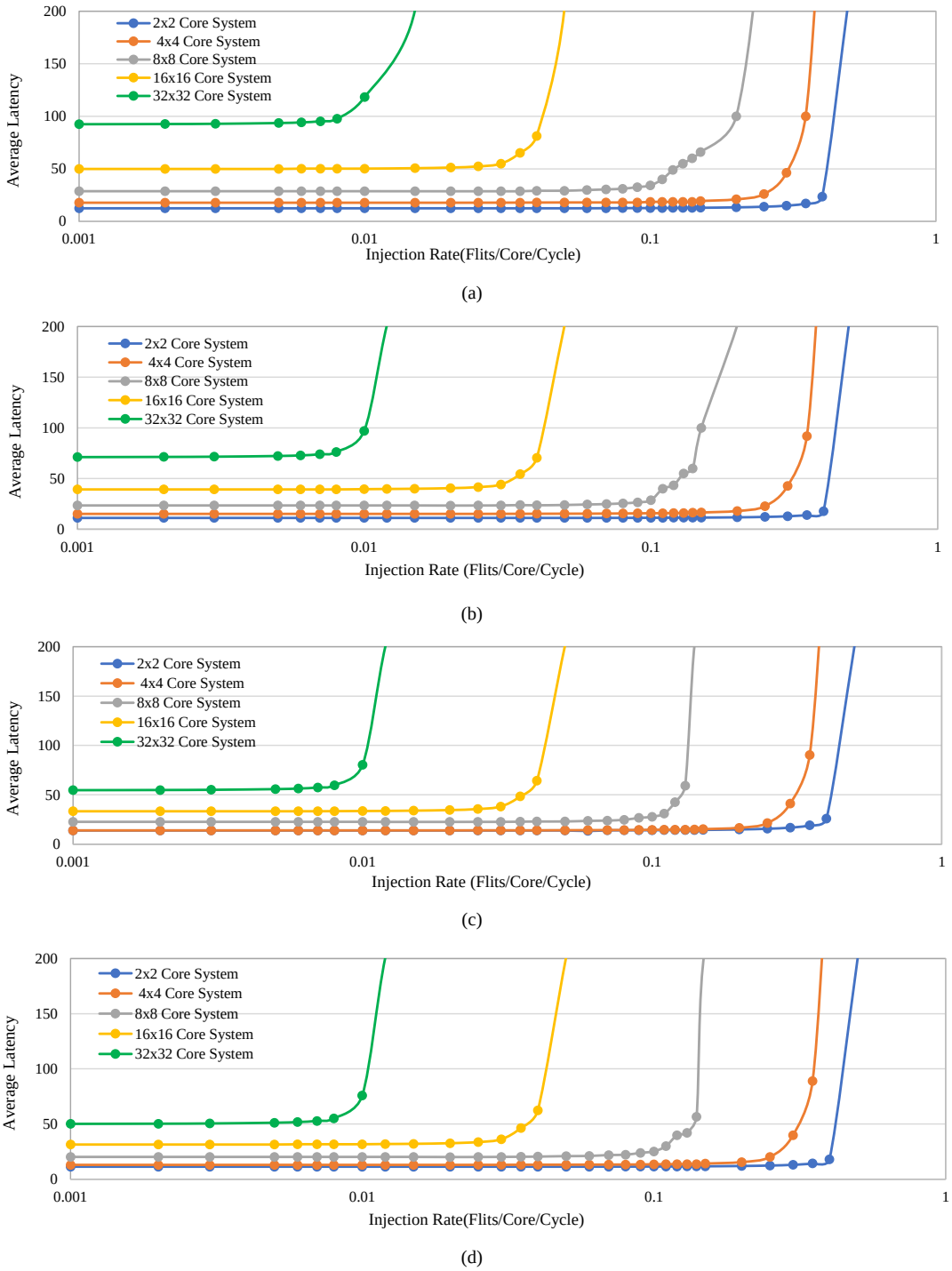


Figure 3.4: Packet latencies for (a) mesh, (b) folded torus, (c) 3D mesh, and (d) small-world topologies with hotspot traffic pattern.

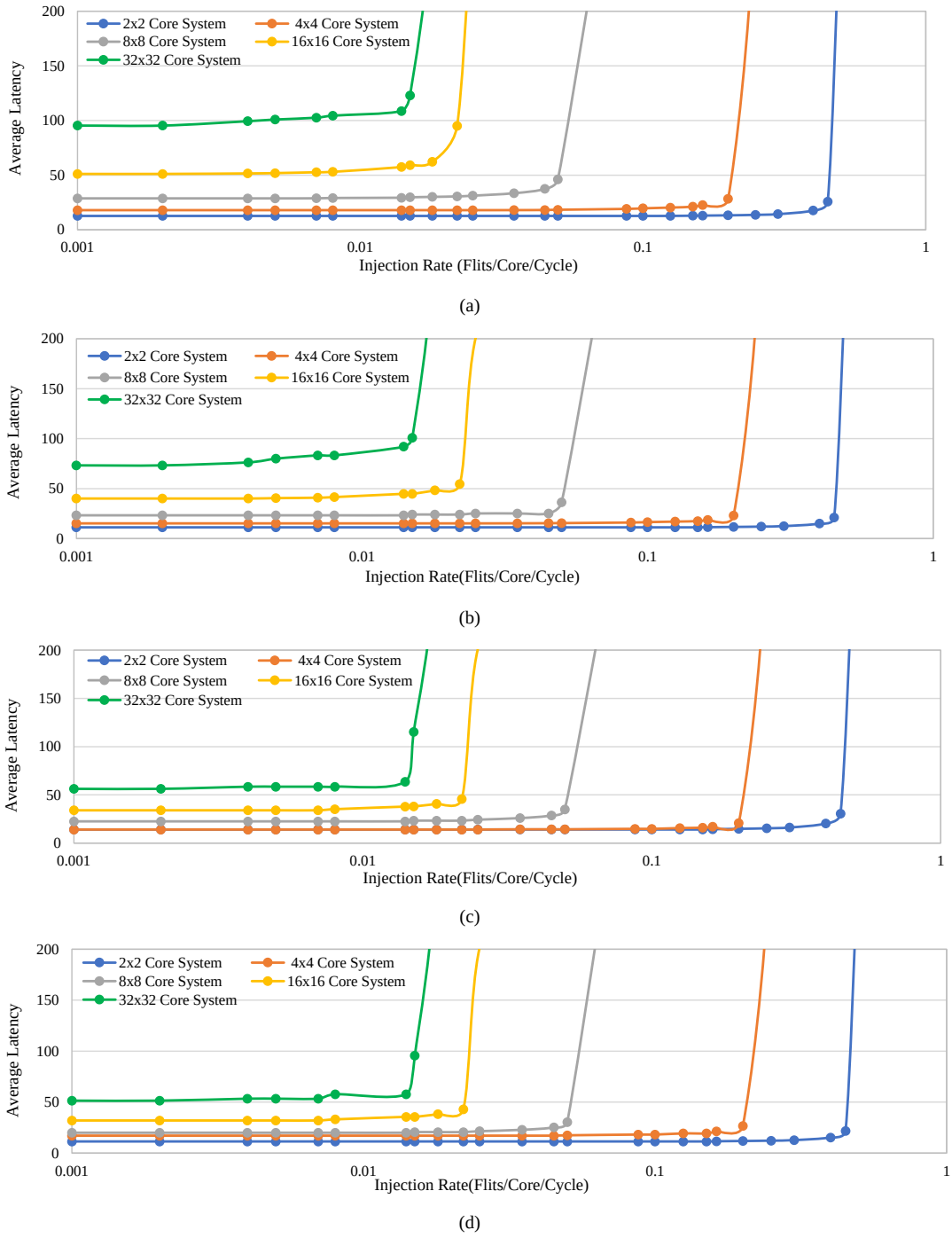
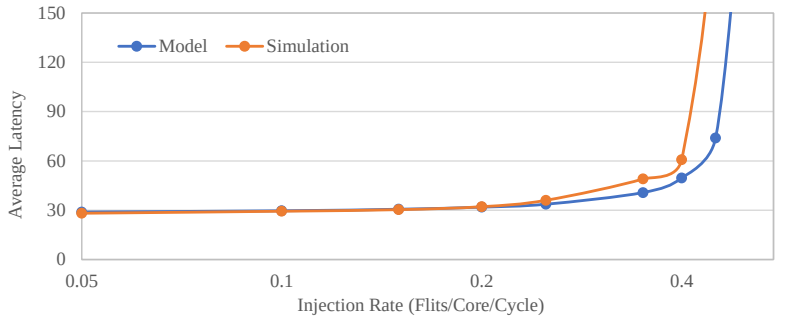
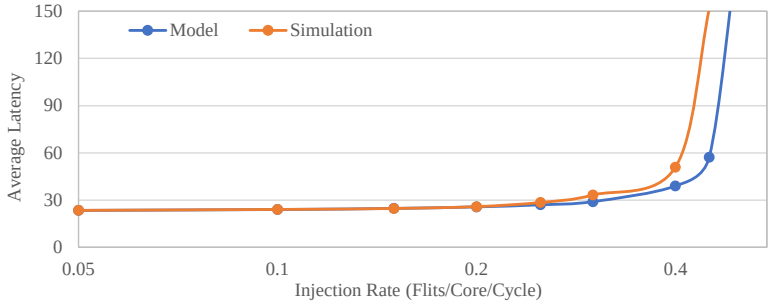


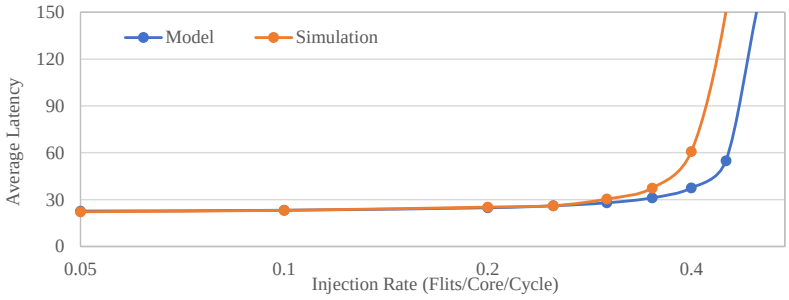
Figure 3.5: Packet latencies for (a) mesh, (b) folded torus, (c) 3D mesh, and (d) small-world topologies with matrix multiplication traffic pattern.



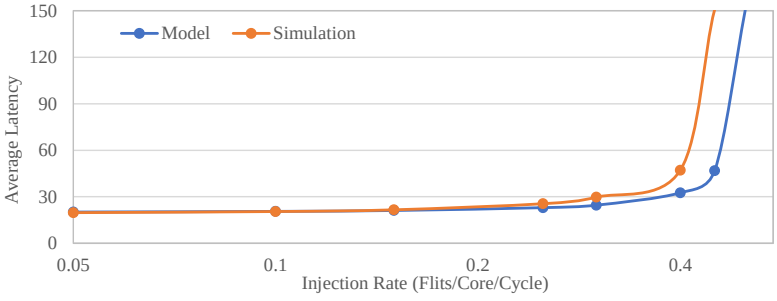
(a)



(b)



(c)



(d)

Figure 3.6: Packet latency from model and simulation for uniform traffic pattern with (a) mesh, (b) folded torus, (c) 3D mesh, and (d) small-world topologies.

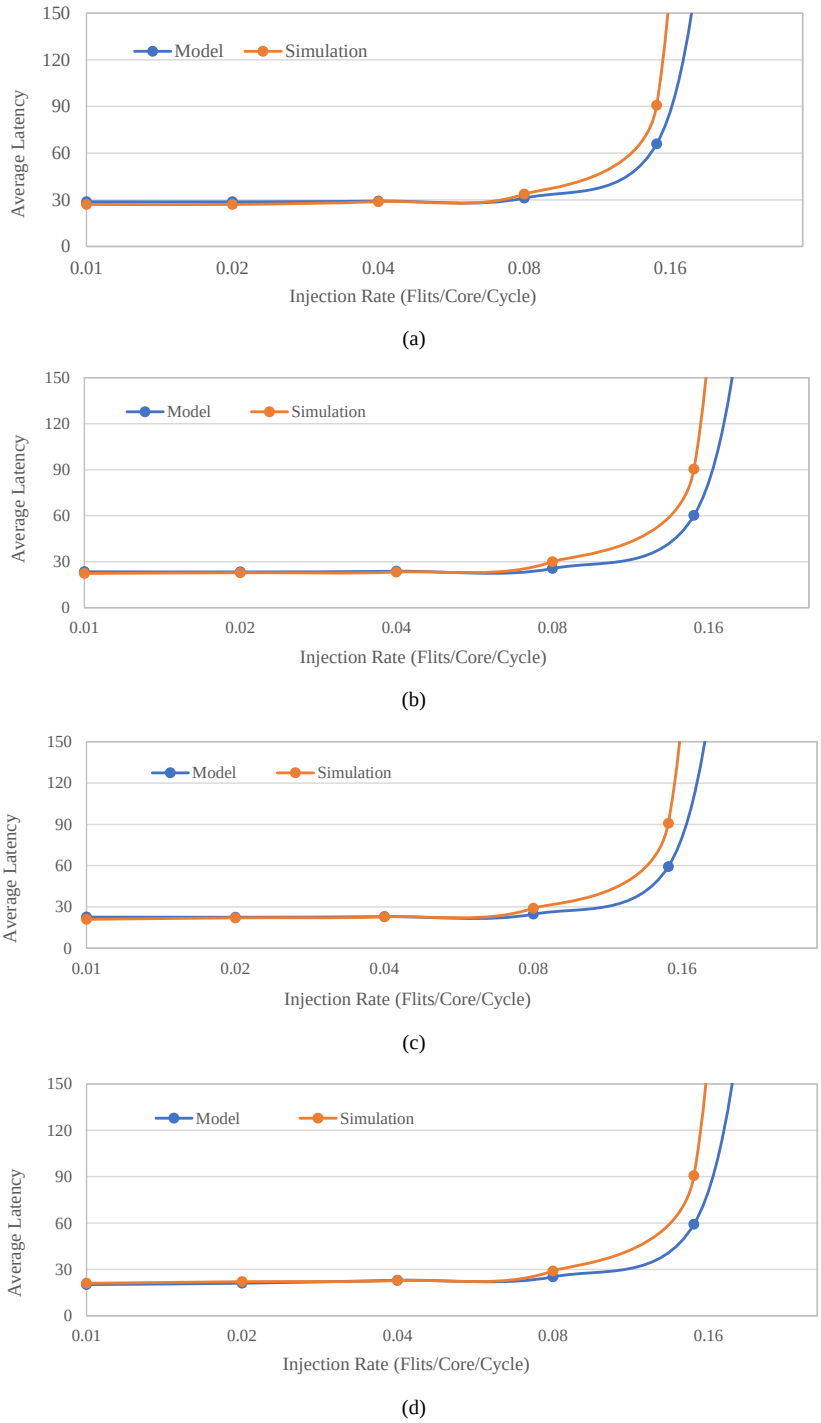


Figure 3.7: Packet latency from model and simulation for hotspot traffic pattern with (a) mesh, (b) folded torus, (c) 3D mesh, and (d) small-world topologies.

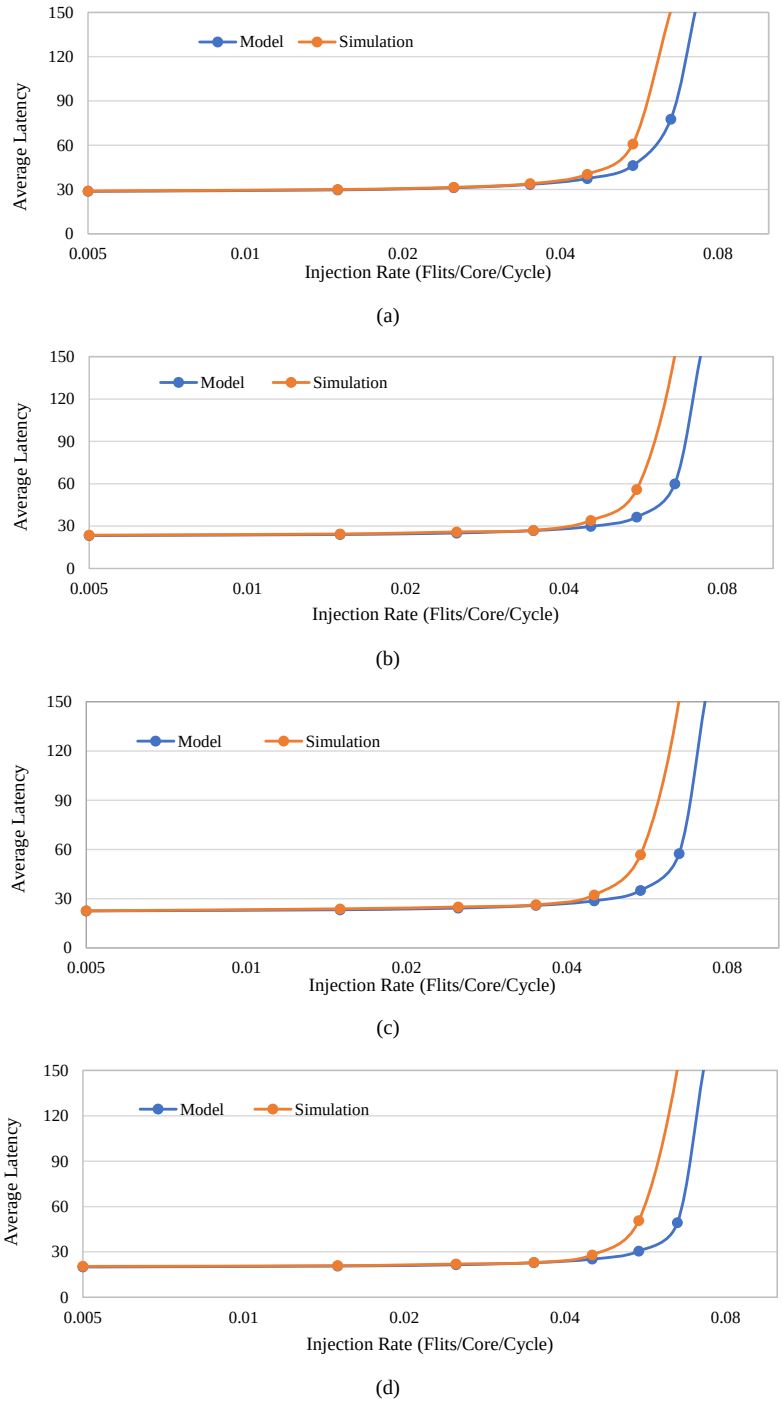


Figure 3.8: Packet latency from model and simulation for matrix multiplication traffic pattern with (a) mesh, (b) folded torus, (c) 3D mesh, and (d) small-world topologies.

Table 3.2: Accuracy of our proposed latency model with recent published latency models.

Queue Model	Source	Error Rate
M/G/1	[45]	9%
M/M/1	[44]	8%
G/G/1	[46]	10%
M/G/1/K	[47]	7.8%
G/G/1/K	[48]	13%
M/M/1/K	Our proposed Model	7%

3.5 Comparison with Other Proposed NoC Latency Models

As we have mentioned in the related work section, a few analytical latency model for NoC based on queuing theory has been published. In this section, we compare the result from our proposed latency model with those that have been published in recent studies. The comparison has been made based on computing the accuracy (i.g. calculating the error rate between the results from the proposed method and the simulation results). the computation error is based on the pre-saturation conditions before the network entering the congestion region or the injection rate of flits per core per cycle that makes the latency starts increasing drastically. Summary for this comparison can be seen in Table 3.2.

Assuming finite buffer capacity, Authors in [48] and [47] proposed G/G/1/K and M/G/1/K queuing models for NoC performance analysis. Their proposed model achieved less than 13% and 7.8% error in the pre-saturating network with various traffic pattern. On the other hand, authors in [45], [44], and [46] assumed M/G/1 , M/M/1, and G/G/1 queuing model to analytically estimate NoC latency. The error rate of their models with simulation results was found to be 9%, 8%, and 10%.

It is clear that our proposed model has better estimation for the NoC latency as it achieves less that 7% error rate with low injection networks.

Chapter 4

The PaSE Framework

This chapter describes our proposed Parallel Speedup Estimation (PaSE) framework that can be used to analytically calculate the achievable speedup taking into consideration the effect of the network latency which also estimated in this thesis. Then, in this chapter, we proposed a case study of multi-core speedup with data parallel application i.g. matrix multiplication. We show how the system size, NoC architecture and computation-to-communication ration effect the speedup both in pre-saturation and saturation network. We also show a comparison between our PaSE framework and a previous study in term of accuracy of estimation the speedup.

4.1 Speedup Model

The speedup model PaSE is used to determine the speedup of a NoC based multi-core system. The speedup is defined as the ratio of the serial execution time on a single core and the parallel execution time in a multi-core environment.

The speedup model requires application profile and the architecture of the cores. The application profile contains the fraction of the application that requires serial execution S and the fraction of the application that requires parallel execution P. It also contains the type and number of operations (i.e. O_I) required by the application. On the other hand, the architecture of the cores defines the number of cycles required to complete different instructions (i.e. C_I). Using these parameters and the latency of

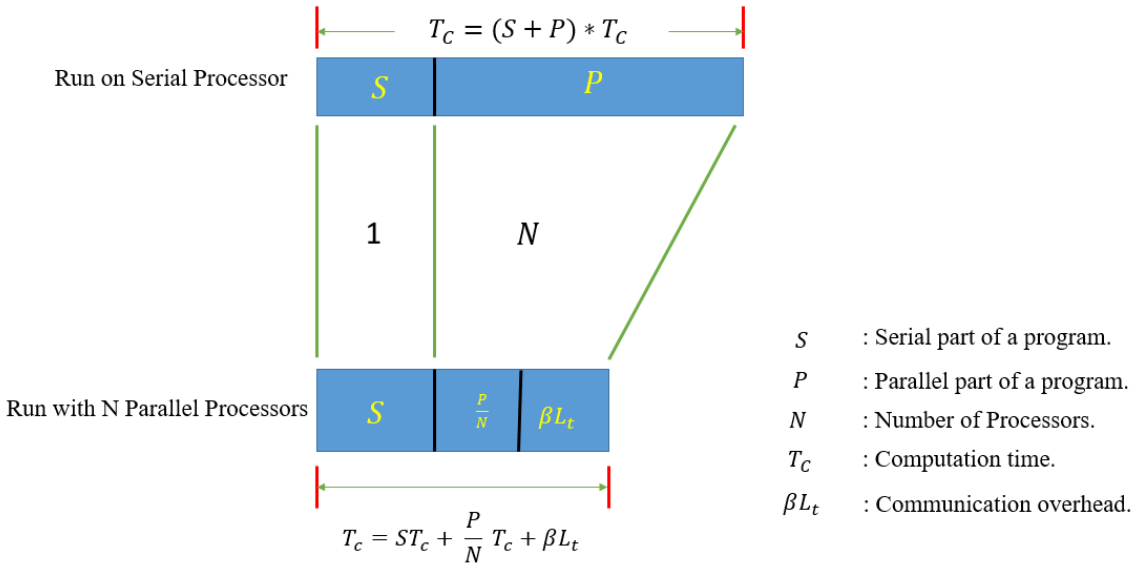


Figure 4.1: Proposed PaSE Framework takes into consideration the effect of the NoC latency which is proposed based on using M/M/1/K queuing model.

the NoC from the proposed latency model, the speedup model computes the parallel execution time, T_p with a N core system using the following equation,

$$T_p = ST_C + \frac{P}{N}T_c + \beta L_t \tag{4.1}$$

Where, L_t is the communication overhead of the application and calculated using (3.1), T_c is the time required to finish the total task on a single core and β is a coefficient denoting the dependency of the application on communication overhead. β can be a positive whole number that signifies the number of messages required to arrive at the core before execution of a task can proceed. It can also have real fractional values if portion of the packet latency can be masked by computation at the core due to its architecture design. This equation models the parallel execution time The time required to finish a subtask depends on the type and the number of instructions in the subtasks and the time required by the core to finish each of the instructions. Using the parallel execution time from equation (4.1), the speedup for the multi-core system with N cores using PaSE is given by the following equation,

$$Speedup = \frac{(S+P)T_c}{ST_c + \frac{P}{N}T_c + \beta L_t} \tag{4.2}$$

A representation for the PaSE framework speedup equation which includes latency model is shown in Figure 4.1.

The last term in the denominator which is βL_t describes the communication latency to perform the program. Ignoring this term, we can get the simplified version of Amdahl's law in [8] as the equation below.

$$Speedup = \frac{(S+P)T_c}{ST_c + \frac{P}{N}T_c} \tag{4.3}$$

4.2 Case Study: Speedup Analysis with Matrix Multiplication

In this section, we present a case study of multi-core speedup with data parallel application like matrix multiplication. For the matrix multiplication application, we consider the multiplication of two [32 x 32] matrix, A and B. The resultant matrix is also a [32 x 32] matrix. The interaction between the cores depends on the mapping of the application between cores in the multi-core system. For this evaluation, we consider the following mapping, computation, and communication among the cores.

- The data of the two matrices A and B are equally distributed among the cores and each core computes the same number of elements of the resultant C matrix (i.e. uniform workload distribution). We consider the cores to run only the matrix multiplication with no other overhead (e.g. no scheduling overhead), thereby making $S = 0$ and $P = 1$.

- To calculate one element of the resultant C matrix, elements in a row of matrix A is multiplied with corresponding elements in a column of matrix B. These partial products are then added to determine one element of the resultant ma-

trix. Therefore, calculating one element of the resultant matrix, C requires 32 multiplications and 31 addition instructions. Hence, for all the elements of the matrix the total computation time, T_C will be $32*32*(32M+31A)$ cycles. Here, M and A refers to the time required by the core to finish a multiplication and addition instruction respectively. Here, we assume a core micro-architecture such that each multiplication or addition operation does not begin until the previous one is finished.

- To perform this matrix multiplication each core shares its values of matrix A and B with other cores in the same row and column. Furthermore, we assume a core can start its computation as it receives the first packet containing the elements of A and B. The rest of the packet latency is masked by the computation. Hence, we assume β to be 1 in our evaluation.

We consider three different cases: ideal-core, integer (i.e. matrix elements are integer numbers), and denormal (i.e. matrix elements are denormal numbers, NaNs or infinity) for this evaluation. In the ideal-core case, we assume a core architecture such that it can complete any instruction (whether it is addition or multiplication) within one clock cycle making $M = A = 1$. For the integer and denormal type matrix elements the number of cycles corresponding to these operations in the Intel knights landing processor is used [55]. Each of these cases will have different values of T_c with ideal-core case being the lowest and denormal case being the highest. We also evaluate the speedup for different NoC architectures and system sizes. With varying system size the problem size assigned to each core will also vary. For example, for a system size of 8x8(64) cores, the problem size is $1024/64 = 16$ whereas for a system size of 32x32(1024) cores, the problem size is $1024/1024 = 1$ at each core. Furthermore, to capture the variation in packet injection due to the different core architecture (e.g. prefetching mechanism, cache replacement policy) we consider both pre-saturation and post-saturation cases for the speedup evaluation using the framework. However,

to validate the latency model with matrix multiplication traffic, we compare the latency results from the latency model with the latencies from simulation for different injection loads. These packet latencies are shown in Figure 3.8. As can be seen from the figure, the latency results from the model closely resembles the latencies from the simulation. Figure 4.2 and Figure 4.3 show the speedup for the different NoC architectures for the matrix multiplication traffic pattern under pre-saturation and post-saturation cases respectively. We analyze these results based on three aspects effect of system size, effect of the NoC architecture, and effect of the computation-to-communication ration. This analysis is presented below:

4.2.1 Effect of System Size

In pre-saturation, the speedup increases with increasing system size. This is shown in Figure 4.2 (a-c). From the figures, we can observe that for all NoC architectures the speedup is lowest for the ideal-core case (Figure 4.2 (a)) and highest for denormal numbers (Figure 4.2 (c)). This is because, in ideal-core case, the computation time decreases with increasing system size (more parallel resources). However, the packet latency increases, as the effective number of hops increases. Due to this, the communication latency plays a vital role in the speedup. This is evident, from (Figure 4.2 (a)), as NoC architecture with higher latencies shows lower speedup for a system size of 32 x 32(1024) cores. On the other hand, for denormal numbers, the computation time is higher than the packet latency even with increasing system size. As the computation time dominates communication, the effect of the latency becomes insignificant for speedup evaluation.

Figure 4.3 (a-c) shows the speedup achieved in the various cases with increase in system-size while the NoC is in saturation. For all the 3 cases considered here, the speedup of the system in saturation is lower than when the system is operating in pre-saturation range. This is because the NoC latency in post-saturation cases are

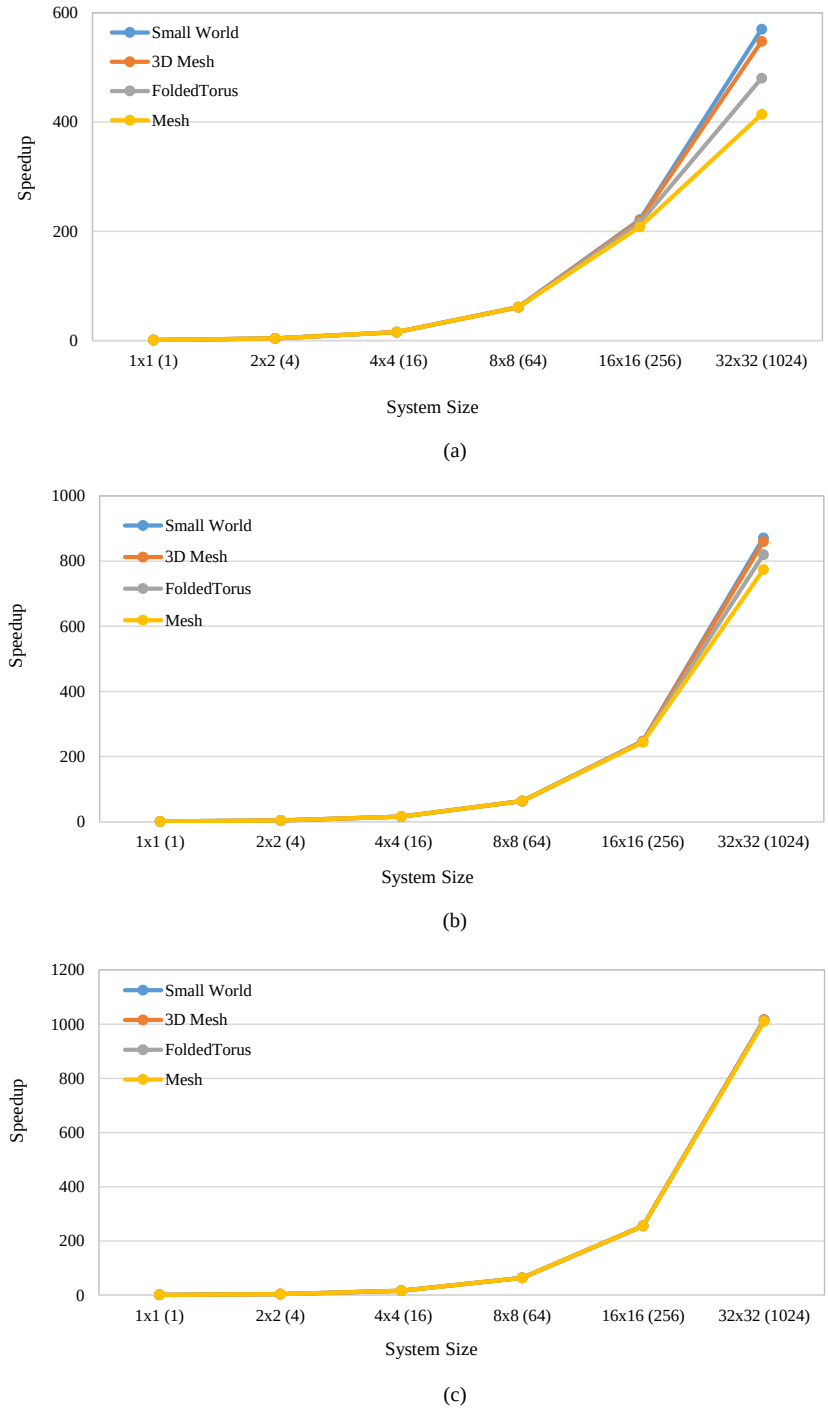


Figure 4.2: Speedup of NoC based multi-core with network at pre-saturation. (a) Ideal-case case (b) Integer case (c) Denormal case.

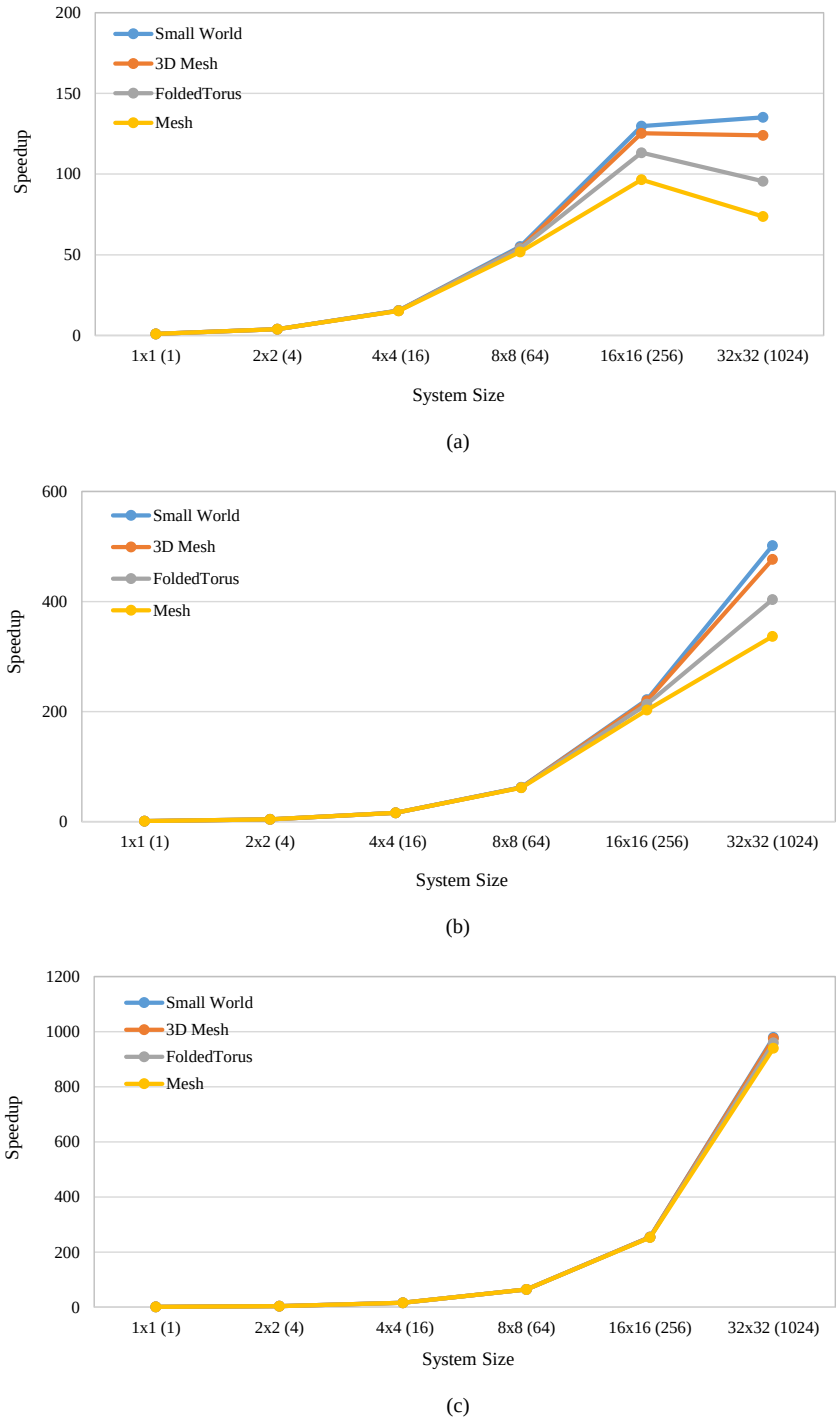


Figure 4.3: Speedup of NoC based multi-core with network at post-saturation. (a) Ideal-case case (b) Integer case (c) Denormal case.

higher resulting in a higher parallel execution time. Moreover, in the post-saturation case, the speedup in the ideal-core case is lower than the speedup in the denormal case as in pre-saturation. However, in this case it is interesting to note that the speedup does not monotonically continue to increase with size. This is because with size the latency numbers increase significantly and overshadows the effect of increased parallelization among the cores.

In case of integer numbers, the number of cycles required for computation increases in proportion to that of the communication latency and hence the computation parts starts to dominate the speedup. However, for very large systems with number of cores higher than 256 cores, the communication latency starts dominating the speedup. Therefore, different NoC architectures have different speedups for large systems. Lastly, for the denormal case the number of cycles required in computation is very large which eliminates the artefacts of the NoC topologies and the speedup is dominated solely by the effect of increase parallelization.

4.2.2 Effect of NoC Architecture

The packet latency is dependent on the NoC architecture. NoC architectures with lower effective number of hops have lower latencies as they provide shorter distance to the destination. Hence, more efficient NoCs will result in higher speedups. However, in our case studies the effect of the NoCs on the speedup is visible only when the execution time is dominated by the NoC latency. This happens for the ideal-core and integer cases for both pre-saturation and post saturation operation. As the NoC latencies decrease from mesh, folded torus, 3D mesh to small world topology, the speedups increase in the reverse sequence in all the cases.

4.2.3 Effect of the C-to-C Ratio

The C-to-C ratio signifies the volume of computation of a parallel application with respect to its volume of communication. For computation intensive application the C-to-C ratio is high and for communication intensive application the C-to-C ratio is low. When an application is computation intensive, its speedup is dominated by the computation time. This can be observed from Figure 4.2 (c) and Figure 4.3 (c) that shows the speedup for the denormal case in both pre and post saturation scenarios. As the computation time is high, the effect of packet latency is masked and all the NoC architectures have similar speedup behavior. On the other hand, when an application is communication intensive, the latency governs the speedup of the application. This can be observed for Figure 4.2 (a),(b) and Figure 4.3 (a),(b) where are post saturation load speedup of different NoC architectures with ideal-core and integer cases are shown. It can be seen from the figures that when the system size increases, due to increase in packet latency and decrease in computation time the C-to-C ratio decreases. In such case, NoC architecture with higher latency (e.g. Mesh), yielding a lower C-to-C ratio has the lowest speedup among other NoC architectures for which the C-to-C ratio is comparatively higher. Thus in cases with low C-to-C ratios the effect of the NoC becomes dominant particularly, at large system sizes.

4.3 Comparison with other Speedup model

In this section, we compare the achievable speedup from our PaSE framework with the model that has been derived in [43]. Authors in [43] take the product of two matrices, A [64,1] and B [1,64] as an application example to get the speedup value using their proposed model as well as they perform system simulation to get the actual speedup for the comparison purpose. This has been done with respect to mesh and torus topologies, uniform and hotspot traffic patterns, and different system

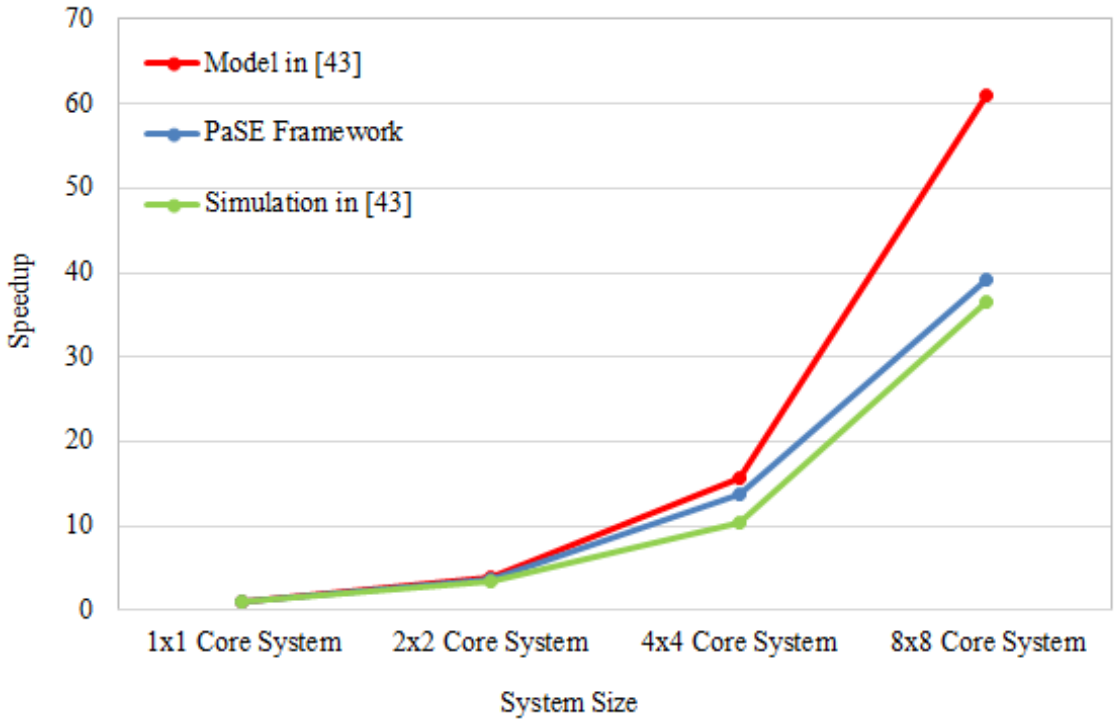


Figure 4.4: Speedup comparison between PaSE framework and the proposed model in [43].

sizes varying from 1x1 (1), 2x2 (4), 4x4 (16), 8x8 (64) core system. However, they did not mention to the size of the packet and the router design, and the width of the interconnect (link width). As it demonstrated earlier that our PaSE framework takes into consideration the network latency which includes both the latency of header and body flits as well as the router design. To estimate these latencies, both of packet size, routers design, interconnects width should be known. Therefore, to compare our PaSE speedup result with the model and simulation results in [43], we have made some assumptions with respect to same network sizes, mesh topology, and uniform traffic pattern that have been used in [43].

Authors in [43] assumed a maximal communication overhead in their estimation equation of speedup. As the communication overhead is a term in the denominator of the proposed speedup model, this gives them the minimal speedup or the lower bound speedup considering all the other parameters (computation time Tc, serial and parallel subtask) is fixed. For this reason, we used our proposed PaSE model to analysis the

speedup assuming that the network is in saturation i.g. when the network enters the congestion regions which resulted in increasing the latency drastically. In addition, we assumed that the packets size is 8 flits which is a practical estimation for the packet size and the width of interconnects is same as flit size so that flit will spend just one cycle to traverse between any two neighboring routers. We also assumed three stage pipelines router design with 2 VCs one for input and one for output with same as packet size. Then, we have consider all these assumption to estimate the network latency using our proposed latency model and estimate the speedup using our PaSE framework. Thereafter, we compared the speedup results using our PaSE framework and the proposed model in [43] with respect to the system simulation results that the author did in [43]. The comparison for these speedup results can be seen in Figure 4.4 and it shows that as we increased the network size, the PaSE speedup achieve better results and get closer to simulation than the theoretical model results in [43]. It shows that PaSE framework results closely resemble to the simulation in [43] with an error rate less than 8% with 8x8 (64) core system whereas their model shows an error rate with 40%.

To sum up, speedup analysis using our PaSE framework shows better results than the best estimation of the speedup model derived in [43]. On the other hand, given the exact simulation parameters including the packet size, interconnects width, and routers design, PaSE framework might shows better estimation for the speedup with respect to the simulation results.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, we propose a framework to model the speedup of a NoC based multi-core processor while performing parallel tasks. By using a queuing theory based model to compute the latency of packet transfer in various NoC architectures and then use the latency to calculate the communication overhead of parallel tasks running on a multi-core chips interconnected with the NoC. Using this overhead we calculate the achievable speedup in such a system. We find that the speedup depends upon a number of factors such as system-size, the nature of the task and the computation to communication ratio. Interestingly, we find that under certain circumstances, when the system is dominated by communication latency increasing the number of cores in a system may not necessarily result in higher speedup. Therefore, using our model it is possible to understand an optimal system-size for certain applications and then use that as a design guideline for more precise simulation based performance estimates. This model can therefore reduce design time and effort of such NoC based multi-core processors.

5.2 Future Work

Both of the proposed Parallel Speedup Estimation PaSE and the latency model can be extended to include the following:

- In this thesis, we use one of the emerging interconnect in our evaluation which 3D mesh topology. Therefore, it is interesting to extend this work to estimate the latency with other two emerging interconnect photonic and wireless As these two emerging interconnects has proved to be the promising interconnect that can achieve low power dissipation and increase performance.

- One assumption has been made in this work is that the width of the interconnects have the same size of the flit which is result in transferring the flit between any two adjacent routers in one cycle. However, sometimes the flits size is greater than the interconnects width. This would make the flit spend more than one cycle to traverse to the neighboring routers. Therefore, it is significant to extend the proposed latency work for such a cases.

- Running a full system simulation to see how the proposed PaSE framework results closely resemble the speedup results from simulation and estimate the error rate for that.

Bibliography

[1] R. R. Schaller, "Moore's law: past, present and future," *IEEE Spectrum*, vol. 34, no. 6, pp. 52–59, Jun 1997.

[2] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, L. Bao, J. Brown *et al.*, "Tile64-processor: A 64-core soc with mesh interconnect," in *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International.* IEEE, 2008, pp. 88–598.

[3] S. Borkar, "Thousand core chips: a technology perspective," in *Proceedings of the 44th annual Design Automation Conference.* ACM, 2007, pp. 746–749.

[4] R. Ubal, J. Sahuquillo, S. Petit, and P. Lopez, "Multi2sim: A simulation framework to evaluate multicore-multithreaded processors," in *Computer Architecture and High Performance Computing, 2007. SBAC-PAD 2007. 19th International Symposium on*, Oct 2007, pp. 62–68.

[5] Dally, T. William J, and Brian, "Route packets, not wires: On-chip interconnection networks," in *Design Automation Conference, 2001. Proceedings.* IEEE, 2001, pp. 684–689.

[6] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Design Automation Conference, 2001. Proceedings.* IEEE, 2001, pp. 684–689.

[7] Y. Ben-Itzhak, I. Cidon, and A. Kolodny, "Delay analysis of wormhole based heterogeneous noc," in *Networks on Chip (NoCS), 2011 Fifth IEEE/ACM International Symposium on.* IEEE, 2011, pp. 161–168.

[8] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, spring joint computer conference.* ACM, 1967, pp. 483–485.

[9] M. D. Hill and M. R. Marty, "Amdahl's law in the multicore era," *Computer*, vol. 41, no. 7, 2008.

[10] S. Borkar, "Obeying moore's law beyond 0.18 micron [microprocessor design]," in *ASIC/SOC Conference, 2000. Proceedings. 13th Annual IEEE International.* IEEE, 2000, pp. 26–31.

[11] S. Winegarden, "Bus architecture of a system on a chip with user-configurable system logic," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 3, pp. 425–433, 2000.

[12] D. Flynn, "Amba: enabling reusable on-chip designs," *IEEE micro*, vol. 17, no. 4, pp. 20–27, 1997.

[13] M. Sharma and D. Kumar, "Design and synthesis of wishbone bus dataflow interface architecture for soc integration," in *India Conference (INDICON), 2012 Annual IEEE*. IEEE, 2012, pp. 813–818.

[14] A. Goel and W. R. Lee, "Formal verification of an ibm coreconnect processor local bus arbiter core," in *Proceedings of the 37th Annual Design Automation Conference*. ACM, 2000, pp. 196–200.

[15] C. Grecu, P. P. Pande, A. Ivanov, and R. Saleh, "Structured interconnect architecture: a solution for the non-scalability of bus-based socs," in *Proceedings of the 14th ACM Great Lakes symposium on VLSI*. ACM, 2004, pp. 192–195.

[16] R. Ho, K. W. Mai, and M. A. Horowitz, "The future of wires," *Proceedings of the IEEE*, vol. 89, no. 4, pp. 490–504, 2001.

[17] P. Kapur, G. Chandra, J. P. McVittie, and K. C. Saraswat, "Technology and reliability constrained future copper interconnects. ii. performance implications," *IEEE Transactions on Electron Devices*, vol. 49, no. 4, pp. 598–604, 2002.

[18] D. Sylvester and K. Keutzer, "Impact of small process geometries on microarchitectures in systems on a chip," *Proceedings of the IEEE*, vol. 89, no. 4, pp. 467–489, 2001.

[19] J. Duato, S. Yalamanchili, and L. M. Ni, *Interconnection networks: an engineering approach*. Morgan Kaufmann, 2003.

[20] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani, "A network on chip architecture and design methodology," in *VLSI, 2002. Proceedings. IEEE Computer Society Annual Symposium on*. IEEE, 2002, pp. 117–124.

[21] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Design Automation Conference, 2001. Proceedings*. IEEE, 2001, pp. 684–689.

[22] A. Ganguly, K. Chang, S. Deb, P. P. Pande, B. Belzer, and C. Teuscher, "Scalable hybrid wireless network-on-chip architectures for multicore systems," *IEEE Transactions on Computers*, vol. 60, no. 10, pp. 1485–1502, 2011.

[23] L. Shang, L. Peh, A. Kumar, and N. K. Jha, "Temperature-aware on-chip networks," *IEEE Micro*, vol. 26, no. 1, pp. 130–139, 2006.

[24] A. Ganguly, K. Chang, S. Deb, P. P. Pande, B. Belzer, and C. Teuscher, "Scalable hybrid wireless network-on-chip architectures for multicore systems," *IEEE Transactions on Computers*, vol. 60, no. 10, pp. 1485–1502, 2011.

[25] S. Deb, A. Ganguly, K. Chang, P. Pande, B. Beizer, and D. Heo, "Enhancing performance of network-on-chip architectures with millimeter-wave wireless interconnects," in *Application-specific Systems Architectures and Processors (ASAP), 2010 21st IEEE International Conference on*. IEEE, 2010, pp. 73–80.

[26] V. F. Pavlidis and E. G. Friedman, "3-d topologies for networks-on-chip," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 10, pp. 1081–1090, 2007.

[27] M. S. Shamim, A. Ganguly, C. Munuswamy, J. Venkatarman, J. Hernandez, and S. Kandlikar, "Co-design of 3d wireless network-on-chip architectures with microchannel-based cooling," in *Green Computing Conference and Sustainable Computing Conference (IGSC), 2015 Sixth International*. IEEE, 2015, pp. 1–6.

[28] A. Shacham, K. Bergman, and L. P. Carloni, "Photonic networks-on-chip for future generations of chip multiprocessors," *IEEE Transactions on Computers*, vol. 57, no. 9, pp. 1246–1260, 2008.

[29] J.-J. Lin, H.-T. Wu, Y. Su, L. Gao, A. Sugavanam, J. E. Brewer *et al.*, "Communication using antennas fabricated in silicon integrated circuits," *IEEE Journal of solid-state circuits*, vol. 42, no. 8, pp. 1678–1687, 2007.

[30] N. Mansoor and A. Ganguly, "Reconfigurable wireless network-on-chip with a dynamic medium access mechanism," in *Proceedings of the 9th International Symposium on Networks-on-Chip*. ACM, 2015, p. 13.

[31] V. Vijayakumaran, M. P. Yuvaraj, N. Mansoor, N. Nerurkar, A. Ganguly, and A. Kwasinski, "Cdma enabled wireless network-on-chip," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 10, no. 4, p. 28, 2014.

[32] N. Mansoor, P. J. S. Iruthayaraj, and A. Ganguly, "Design methodology for a robust and energy-efficient millimeter-wave wireless network-on-chip," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 1, no. 1, pp. 33–45, 2015.

[33] S. Deb, K. Chang, X. Yu, S. P. Sah, M. Cosic, A. Ganguly, P. P. Pande, B. Belzer, and D. Heo, "Design of an energy-efficient cmos-compatible noc architecture with millimeter-wave wireless interconnects," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2382–2396, 2013.

[34] B. S. Feero and P. P. Pande, "Networks-on-chip in a three-dimensional environment: A performance evaluation," *IEEE Transactions on computers*, vol. 58, no. 1, pp. 32–45, 2009.

[35] J. Duato, S. Yalamanchili, and L. M. Ni, *Interconnection networks: an engineering approach*. Morgan Kaufmann, 2003.

[36] E. Kakoulli, V. Soteriou, and T. Theocharides, "Intelligent hotspot prediction for network-on-chip-based multicore systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 3, pp. 418–431, 2012.

[37] R. Marculescu, U. Y. Ogras, L.-S. Peh, N. E. Jerger, and Y. Hoskote, "Outstanding research problems in noc design: system, microarchitecture, and circuit

perspectives," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 1, pp. 3–21, 2009.

[38] X.-H. Sun and Y. Chen, "Reevaluating amdahlâĂŹs law in the multicore era," *Journal of Parallel and Distributed Computing*, vol. 70, no. 2, pp. 183–188, 2010.

[39] J. L. Gustafson, "Reevaluating amdahl's law," *Communications of the ACM*, vol. 31, no. 5, pp. 532–533, 1988.

[40] B. H. Juurlink and C. Meenderinck, "Amdahl's law for predicting the future of multicores considered harmful," *ACM SIGARCH Computer Architecture News*, vol. 40, no. 2, pp. 1–9, 2012.

[41] E. Yao, Y. Bao, G. Tan, and M. Chen, "Extending amdahl's law in the multicore era," *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, no. 2, pp. 24–26, 2009.

[42] S. M. Londoño and J. P. De Gyvez, "Extending amdahl's law for energy-efficiency," in *Energy Aware Computing (ICEAC), 2010 International Conference on*. IEEE, 2010, pp. 1–4.

[43] X. Chen, Z. Lu, A. Jantsch, and S. Chen, "Speedup analysis of data-parallel applications on multi-core nocs," in *ASIC, 2009. ASICon'09. IEEE 8th International Conference on*. IEEE, 2009, pp. 105–108.

[44] Z. Guz, I. Walter, E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "Network delays and link capacities in application-specific wormhole nocs," *VLSI Design*, vol. 2007, 2007.

[45] U. Y. Ogras and R. Marculescu, "Analytical router modeling for networks-on-chip performance analysis," in *Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE'07*. IEEE, 2007, pp. 1–6.

[46] A. E. Kiasari, Z. Lu, and A. Jantsch, "An analytical latency model for networks-on-chip," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 1, pp. 113–123, 2013.

[47] M. Lai, L. Gao, N. Xiao, and Z. Wang, "An accurate and efficient performance analysis approach based on queuing model for network on chip," in *Computer-Aided Design-Digest of Technical Papers, 2009. ICCAD 2009. IEEE/ACM International Conference on*. IEEE, 2009, pp. 563–570.

[48] Z. Qian, D.-C. Juan, P. Bogdan, C.-Y. Tsui, D. Marculescu, and R. Marculescu, "A comprehensive and accurate latency model for network-on-chip performance analysis," in *Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific*. IEEE, 2014, pp. 323–328.

[49] Z.-L. Qian, D.-C. Juan, P. Bogdan, C.-Y. Tsui, D. Marculescu, and R. Marculescu, "A support vector regression (svr)-based latency model for network-on-chip (noc) architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 3, pp. 471–484, 2016.

[50] M. B. Marvasti and T. H. Szymanski, "An analysis of hypermesh nocs in fpgas," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 10, pp. 2643–2656, 2015.

[51] M. S. Shamim, N. Mansoor, R. S. Narde, V. Kothandapani, A. Ganguly, and J. Venkataraman, "A wireless interconnection framework for seamless inter and intra-chip communication in multichip systems," *IEEE Transactions on Computers*, vol. 66, no. 3, pp. 389–402, 2017.

[52] D. G. Kendall, "Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded markov chain," *The Annals of Mathematical Statistics*, pp. 338–354, 1953.

[53] J. Sztrik, "Basic queueing theory," *University of Debrecen, Faculty of Informatics*, vol. 193, 2012.

[54] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures," *IEEE transactions on Computers*, vol. 54, no. 8, pp. 1025–1040, 2005.

[55] A. Fog, "Lists of instruction latencies, throughputs and micro-operation breakdowns for intel, amd and via cpus," *Technical University of Denmark*, 2017.