Accelerating Network-on-Chip Simulation via Sampling

Wenbo Dai, Natalie Enright Jerger
Department of Electrical and Computer Engineering, University of Toronto
{daiwenbo, enright}@eecg.toronto.edu

***Abstract*—Architectural complexity continues to grow as we consider the large design space of multiple cores, cache architectures, networks-on-chip and memory controllers for emerging architectures. Simulators are growing in complexity to reflect each of these system components. However, many full-system simulators fail to take advantage of the underlying hardware resources such as multiple cores; as a result, simulation times have grown significantly in recent years. Long turnaround times limit the range and depth of design space exploration that is tractable.**

Communication has emerged as a first class design consideration and has led to significant research into networks-on-chip (NoC). The NoC is yet another component of the architecture that must be faithfully modeled in simulation. Given its importance, we focus on accelerating NoC simulation through the use of sampling techniques; sampling can provide both accurate results and fast evaluation. We propose NoCLabs and NoCPoint, two sampling methodologies utilizing statistical sampling theory and traffic phase behavior, respectively. Experimental results show that our proposed NoCLabs and NoCPoint estimate NoC performance with an average error of 5% while achieving one order of magnitude speedup on average.

I. Introduction

As the number of cores in contemporary processors continues to scale, the criticality of Network-on-Chip (NoC) design to overall performance increases accordingly. NoC designers rely heavily on full-system simulation to faithfully evaluate their designs. In full-system simulation running multi-thread applications, the interaction between applications, cache coherence protocols and the network is fully exercised; the performance of new designs is accurately evaluated.

Although full-system simulation enjoys the benefit of high fidelity, it suffers from prohibitively long turnaround times. Sampled full-system simulation [2], [5], [7], [9] is an effective technique to reduce simulation turnaround times for single-, multi-threaded and multiprogrammed applications. In sampled full-system simulation, only a *small* but *representative* portion of the application is simulated in detail. Performance metrics measured with the sampled application are used to estimate the true values of those metrics; the unsampled intervals are either fast forwarded using functional simulation or skipped entirely. Existing work applies to a wide range of applications (single or multi-threaded) and architectures (homogeneous or heterogeneous); however, they mainly focus on evaluating micro-architecture designs, and report metrics such as CPI (single-thread and multiprogrammed applications) or run time (multi-thread applications). To the best of our knowledge, there is no existing work exploring sampling methodologies for NoC simulation. In this paper, we introduce two sampling methodologies for NoC simulation: *NoCLabs* and *NoCPoint*. They are based on statistical sampling theory and traffic phase behavior information, respectively.

II. NoCLabs: Latency based Statistical Sampling

Inferential statistical sampling aims to estimate a given accumulative property of a population by only measuring a sample. The minimal sample size n needed to represent the population is quadratically proportional to the target metric's variation. We apply statistical sampling theory to NoC simulation. Average packet latency is one of the most commonly reported metrics; therefore, we base our sample selection on its variation. We call our technique Latency based Statistical Sampling for NoC simulation, or NoCLabs.

Full application traffic is first divided into non-overlapping units of size U; the population size N is the total number of U-sized units in the full application traffic. Sample size n refers to the number of units included in the sample. In order to characterize the application traffic in a network-independent manner, we use the user mode instruction count (UMIC) to measure the traffic, as the UMIC is stable across different network configurations for a multi-threaded application. In NoCLabs, the minimal sample size n to represent the population depends upon three variables: 1) The coefficient of variance of the packet latencies per unit in the population, $\hat{V}$; 2) Confidence level $(1-\alpha)$; and 3) Confidence interval $\pm\varepsilon$. $\hat{V} = \frac{\sigma}{\mu}$, where σ and μ are the standard deviation and mean value, respectively. Confidence level and interval are specified by the NoCLabs user. Informally, they indicate that one can be $(1-\alpha)$ confident that the estimated value is within $\pm\varepsilon$ of the true value. The minimal sample size n is defined as: $n \geq (\frac{z}{\varepsilon} \cdot \hat{V})^2$, where z is the $100[1-\frac{\alpha}{2}]$ percentile of the standard normal distribution. After n is decided, systematic sampling is performed on the population: one traffic unit out of every k units is picked as a sample, where $k = \frac{N}{n}$.

III. NoCPoint: Exploiting Traffic Phase Behavior

Prior work [4] shows that application traffic exhibits phase behavior and there is a correlation between traffic phase and network performance. To exploit this phenomenon, we propose NoCPoint sampling methodology. It selects a representative sample for an application by exploiting its traffic phases. The two steps of NoCPoint are described as follows:

1) Characterizing and classifying traffic: To determine the phases that exist in the traffic, one must first characterize how the traffic behaves both temporally and spatially. In NoCPoint, we profile each traffic unit U by a row-column injection-ejection rate vector (IERV). In an IERV, each element represents the injection or ejection rate of a row or column in the network. After the behavior of the full application traffic is characterized, we use hierarchical clustering to cluster

TABLE I. SIMULATION CONFIGURATION

Cores	16, P4-like
L1 Cache (I & D)	private, 4-way, 32KB each, 64 Byte Blocks
L2 Cache	private, 8-way, 512KB each, 64 Byte Blocks
Cache Coherence	MOESI distributed directory
Network 1	4x4 2D-Mesh, adaptive XY/YX routing, 8 VCs, 8 Buffers/VC, 8 Byte flit
Network 2	4X4 2D-Mesh, dimension-order routing, 4 VCs, 4 Buffer/VC, 4 Byte flit

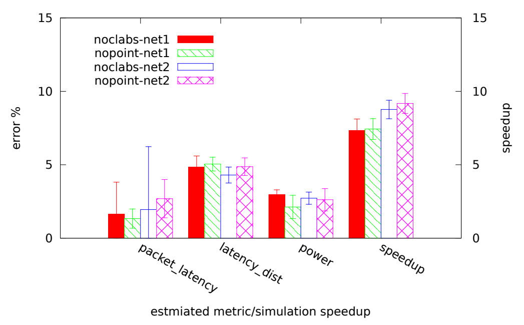

Fig. 1. Accuracy comparisons for NoCLabs and NoCPoint for packet latency, latency distribution, network power for $Network1$ and $Network2$ are shown. Speedup is also presented. Results are averaged across all applications.

traffic phases. The Manhattan distances between IERVs are calculated, then classes are formed so that intra-class IERVs are closer to each other than inter-class IERVs. That is to say, traffic within the same class manifests similar behavior.

2) Sampling the traffic and measuring the network performance: Once the full application traffic has been clustered, the traffic units within one cluster share similar temporal and spatial behavior. We sample the traffic by selecting one unit from each cluster: each chosen unit is the centroid of that cluster. The last step is to measure network performance by simulating the sampled traffic. As clusters may vary in size, the packet latencies of each measured unit are weighted to generate the overall average packet latency. Other network performance metrics, such as latency distribution and network power can also be calculated in a similar way.

IV. EXPERIMENTAL RESULTS

We perform full-system simulation using a cycle-accurate x86 simulator, FeS2 [6]; the network is simulated with Booksim [3]. The configuration is listed in Table I. The simulator switches between detailed timing simulation and fast functional simulation to simulate sampled intervals and unsampled intervals respectively. We select applications from the PARSEC [1] benchmark suite. All applications run with 16 threads, use simsmall input, and run to completion. Only the regions of interest (ROI) are measured. For NoCLabs, we set the traffic unit size U as 10,000 user-mode instructions (UMIs) and use an accuracy requirement of 0.99 confidence level and $\pm 3\%$ confidence interval. For NoCPoint, we empirically set U as 1 million UMIs. To estimate network power data, we use DSENT [8]. DSENT is configured to use a 45nm process and 1 GHz frequency. To calculate the simulation speedup, we compare the wall-clock time of the sampled simulation against that of the full simulation.

Figure 1 shows both the estimation accuracy and simulation speedup gained by NoCLabs and NoCPoint. For the estimation accuracy, the errors are with respect to the true values measured from full-system simulation. As network designers are not only interested in average latencies, but also care about the network congestion level and power consumption, we also provide results for latency distribution and power estimation.

NoCLabs and NoCPoint both have high accuracy in network performance estimation. The errors of latency and power estimation are all less than 3%. The latency distribution errors are higher (maximal average error is 5%) due to the accumulation of multiple comparison points. For all the performance estimation errors, we can also notice the coefficient of variance are less than 1 (with the exception of 1.3 for the latency estimation with NoCPoint-net2), suggesting a stable accuracy across all the applications. In terms of simulation speed, NoCPoint reports a $9.17\times$ speedup with $Network2$; NoCPoint offers an order of magnitude speedup over full-system simulation. Simulations previously running for a week now can be finished within a day.

V. CONCLUSION

In this paper, we propose two sampling methodologies to accelerate NoC simulation: NoCLabs and NoCPoint. They utilize statistical sampling theory and traffic phase behavior information respectively. They select and simulate a small portion of the full application in detail to faithfully evaluate NoC designs. We evaluate NoCLabs and NoCPoint against the full system simulation. Network performance metrics including average packet latency, latency distribution and power are estimated within 5% of the true values. Meanwhile, we speed up simulation by one order of magnitude. With a reduced simulation turnaround time, the range and depth of NoC design space exploration can be enhanced.

ACKNOWLEDGEMENTS

This work was supported by Intel, the Natural Science and Engineering Research Council of Canada and the University of Toronto.

REFERENCES

[1] C. Bienia, "Benchmarking modern multiprocessors," Ph.D. dissertation, Princeton University, January 2011.

[2] T. E. Carlson, W. Heirman, and L. Eeckhout, "Sampled simulation of multi-threaded applications," in *International Symposium on Performance Analysis of Systems and Software*, Apr. 2013.

[3] N. Jiang, D. U. Becker, G. Michelogiannakis, J. Balfour, B. Towles, J. Kim, and W. J. Dally, "A detailed and flexible cycle-accurate network-on-chip simulator," in *Proc. of the International Symposium on Performance Analysis of Systems and Software*, 2013.

[4] Y. Jin, E. J. Kim, and T. M. Pinkston, "Communication-aware globally-coordinated on-chip networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 2, pp. 242–254, Feb. 2012.

[5] E. K. Ardestani and J. Renau, "Esesc: A fast multicore simulator using time-based sampling," in *High Performance Comp Arch*, 2013.

[6] N. Neelakantam, C. Blundell, J. Devietti, M. M. K. Martin, and C. Zilles, "Fes2: A full-system execution-driven simulator for x86," in *Architectural Support for Prog Lang and Operating Systems*, 2008.

[7] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder, "Automatically characterizing large scale program behavior," in *Architectural Support for Programming Languages and Operating Systems*, 2002.

[8] C. Sun, C.-H. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, and V. Stojanovic, "DSENT - a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling," in *International Symposium on Networks on Chip*, 2012.

[9] R. E. Wunderlich, T. F. Wenisch, B. Falsafi, and J. C. Hoe, "SMARTS: accelerating microarchitecture simulation via rigorous statistical sampling," in *Intl Symp on Computer Architecture*, 2003, pp. 84–97.