# PERFORMANCE MODELLING OF MULTICORE AND MANYCORE NETWORKED SYSTEMS

Abu Asaduzzaman

Dept of Elec Eng and Computer Sci, Wichita State University, Wichita, Kansas, USA
Abu.Asaduzzaman@wichita.edu

## *ABSTRACT*

*Multicore computer systems are adopting more cores to meet the increasing performance requirement. Network topology to make communication in such a multicore/manycore system (like Intel 80-Core Research Chip and IBM Roadrunner) is extremely important to achieve high performance by supplying low energy. Designing and analysing such a gigantic system is very complicated and it costs up to millions of dollars. Performance modelling technique is attracting researches from many organizations and different disciplines to conduct research on multicore/manycorenetworked systems in an inexpensive way. However, there is no standard in performance modelling of such complex systems. This paper proposes a methodology of developing conceptual models of multicore/manycorenetworked systems for performance analysis. Important features of this methodology include data acquisition to improve accuracy by capturing the important details about the system under consideration, system-to-model abstraction to make the modelling manageable without eliminating any characteristics of the target system, and model validation to make sure that the model meets the requirements with an optimal choice of the hardware and software. Networked systems with up to 64 cores are modelled; according to the experimental results, the proposed performance modelling methodology is very promising.*

## *KEYWORDS*

*Conceptual modelling, multicore/manycore architecture, network topology, networked systems, performance analysis*

## 1. INTRODUCTION

According to power-aware multicore/manycore computer architecture design trends, it is expected that in the near future, a high performance computing system should consist thousands to millions of processing cores, where each core should be capable of executing multiple threads. As a result, challenges should appear from the facts that multi-level caches, local and remote main memory, and intra-/inter-nodal communication networks should offer an increased and/or unpredictable memory access time within computing systems. Discovering and exploiting parallelism within codes and overlapping different types of operations should be another great challenge [1]-[3]. Network topology used to support core-to-core data communication in multicore architecture is very important to meet the required performance goal [4]-[6]. To conduct research on such a complex and expensive system, conceptual modelling may be the best feasible option for many researchers [7]-[12]. However, conceptual modelling is relatively a new field, it has its own issues, and there is no standard. In this work, some of the key issues regarding modelling multicore/manycore networked systems are addressed and then a new methodology is presented explaining how to develop conceptual models of such complex systems.

Multicore computer/network architecture supporting real-time multimedia applications deals with timing constraints and usually interacts with the environment rather than the human operator. Because timeliness and reliability are so important in their behaviour, real-time multimedia systems are often distributed among multiple program units (a.k.a., tasks) running

DOI : 10.5121/ijcnc.2012.4204 

simultaneously to perform required functions. Concurrent execution of tasks on a single processor, in many respects including energy and thermal constraints, is inadequate for achieving the required level of performance or required level of reliability. Therefore, the tasks are moved to different interconnected processors, making a real-time system parallel and/or distributed. If the communication time between processing units is negligible with respect to the processing time, then the system is referred as parallel; otherwise it is referred as distributed. Because of their high performance and reliability, the popularity and demand of multicore systems (supporting parallel and/or distributed processing) are increasing in both the desktop and the embedded markets [2]-[5].

Chip vendors are developing high-performance teraflop and petaflop computing systems with hundreds and thousands of processing cores. For example, Intel 80-Core Research Chip is a teraflop processor. IBM Blue Gene/L has 128K+ cores and Roadrunner has 130K+ cores. Intel 80-Core Research Chip is not publicly available IBM Gene/L and Roadrunner cost millions of dollars [13]. Modelling technique provides an inexpensive way to conduct research on such complex and expensive systems. The multicore/manycore systemswe model in this work haveup to 64 cores and two levels of caches. Using representative synthetic workload, we explore the impact of the number of cores and various cache parameters on performance and total power consumption.

This paper is organized as follows. In Section 2, some related articles are reviewed. Section 3 discusses various modelling issues related to multicore/manycore networked systems. A methodology for modelling multicore/manycore systems is presented in Section 4 and the methodology is evaluated in Section 5. Some important simulation results are discussed in Section 6. Finally, this work is concluded in Section 7.

## 2. SURVEY

In this section, some published articles we find relevant to performance modelling of multicore/manycore systems are discussed.

Various issues and theories behind modelling and simulation simple computer systems are discussed in [14][15]. Today's computers are much more complicated as they are multicore/manycore systems and they require fast, safe, and reliable network support.

In [16], suggestionsare made to develop a procedure for problem formulation and indicators to verify the formulated problem. In [17], an attempt is made to automate simplification. It should be noted that formulation and automation may degrade the accuracy of the results.Six important principles of modelling are discussed in [18] that cover simplicity versus complexity and serve as the rough guide to modelling. Simplicity in modelling may be dangerous, especially in medical fields as suggested in [19]. Therefore, advanced techniques in modelling multicore/manycore networked systems for analysis is a great challenge.

In [20], a scheme to model and validate high-performance embedded systems is proposed.This approach depends on the repetitive model of computation used to express the parallelism of such systems. In [21], relational coordination is suggested to mediate the association between the high-performance work practices and outcomes. In [22],an interpreter based modelling fora high-performance system is proposed. Modelling techniques presented in [20]-[22] are not suitable for multicore/manycore performance analysis.

Multicore/manycore systems help achieve aggressive performance/power targets; however, designing the interconnect framework and cache memory sub-system are challenging.Some of those issues are addressed in [1]-[6]. However, they do not suggest any modelling schemes.

Various conceptual modelling issues are addressed in [7]-[10]. As mentioned in [7], conceptual modelling is an important part of a simulation study; however, it is not understood very well. According to [9], there are no standard for decomposing the representation of the simulation subject into the entities. In [10], an approach of semiautomatic generation of conceptual models from text is presented.These articles discuss various conceptual modelling issues and try to formulate methodologies for concurrent computing systems. However, none of these techniques is suitable for performance modelling of multicore/manycore networked systems, simply because of the complexity of such a networked system.

In this paper, we propose a methodology to model complex multicore/manycore networked systems for performance analysis. Important features of this methodology include data acquisition to improve accuracy, system-to-model abstraction, and model validation to make sure that the model meets the requirements. We model multicore systems with up to 64 cores and present some important experimental results.

## 3. PERFORMANCE MODELLING

Conceptual modellingis considered as one of the vague areas of simulation study [7]. In the following subsections, some important modelling issues are addressed.

### 3.1. Model Requirements

A simple model has both merits and demerits. A simple model is always expected mainly because it is easy and fast to develop, takes less input and runs fast, and is better understood [11]. However, a simple model may be dangerous as it requires more extensive assumptions to simplify the target system [19]. Especially for high performance multicore computer systems where both the hardware and the software are very complex; the models are expected to be complex as well. Different researchers identify various performance criteria for a good model [7]. It is suggested that a conceptual model should meet the following requirements: simplicity, usability, accuracy, reliability, affordability, and sustainability.

*Simplicity:* Even though simple models may be dangerous, simplicity is emphasized for various reasons. One reason is the fact that conceptual modelling is the first step for most cases and it is far way from the actual implementation (i.e., real danger). If the conceptual model indicates any problem, then it can be easily fixed. Spending a lot of efforts to come up with a complex model may not be always profitable.

*Usability:* Usability is very important, because a model should be used by people at various levels inside and/or outside of the organization (from testers to developers to managers to outside clients). Conceptual models should be easy to use/reuse so that the actual system becomes easy to understand by all users.

*Accuracy:* No wonder, accuracy is an important requirement. Various levels of accuracy during the design cycle are suggested. At the early stage, a reasonable amount of accuracy can be traded-off with the faster development and execution times. However, at a later stage, accuracy becomes more crucial and should not be sacrificed.

*Reliability:* In modelling (and simulation), reliability is an important factor. Reliability is considered to be very critical, as accuracy may be sacrificed at the early stage. At an early stage, if the conceptual model indicates that the target solution is profitable; at a later stage, the refined model should indicate more accurately how profitable it may be.

*Affordability:* The solution of the conceptual modelling should be affordable so that more organizations can take advantage of it.

*Sustainability:* The solution of the conceptual modelling should be sustainable so that it remains diverse and productive over time.

## 3.2. Model Development

Conceptual modelling is more like an art than a science [15]. However, model development is guided by modelling principles and modelling simplification (as explained below).

*Modelling principles:* Modellers have proposed various principles. Some important modelling principles suggested by [14] are,

- Model simple – think complicated
- Be parsimonious – start small and add
- Apply similarities – do not redo what (or if similar thing) has already has been done.

*Modelling simplification:* Simplification means removing details from a model while maintaining a sufficient level of accuracy. Various simplification methods are proposed. Some simplification methods including dropping unimportant components, using random variables to depict parts of the model, and grouping components of the model are discussed in [18]. Some improved methods are discussed in [17] which are based on the initial methods developed in [18].

To some extent, conceptual modelling is involved with traditional modelling and simulation discipline. Usually, simulation models are often developed for a single purpose. Once a model is accepted, it is quite common and desirable to reuse the application of the model to several other areas. But, this may not be a straight-forward evolution because a model designed and developed to evaluate one performance measure of a system may not be well-suited for others. However, (unlike simulation model) conceptual model defined for a system can be easily reused to other systems. Conceptual modelling has gained a lot of interest in recent years and simulation modellers are particularly interested in understanding the processes involved in arriving at a conceptual model. Needless to mention at this point that conceptual modelling is essential in the process of developing and implementing simulation models for next-generation computing systems [7].

## 3.3. Model Validation

Validation is important to improve the credibility of the model and to increase the correctness of the simulation. Validation is also important to check if the conceptual model is sufficiently accurate for its intended purpose. However, defining methods for performing the validation is difficult. A detailed questionnaire for validating the conceptual model is provided in [16]. Using the conceptual model description (project specification) as a means of validation of a model is suggested by [7]. It is recommended that validation process include inputs, outputs, and the requirements.

## 3.4. Other Challenges

As mentioned earlier, modern high performance computing systems are expected to have lots of cores. Multicore architecture brings some important blessings like improved performance / power ratio. However, multicore systems introduce several design and implementation challenges like complexity. To understand the complexity of a high performance computing system, let's briefly discuss IBM's Roadrunner supercomputer. Roadrunner is the first hybrid supercomputer, where general purpose processors are coupled with specialized co-processors. Roadrunner has more than 130,000 cores (13,824 Opteron cores + 116,640 Cell cores = 130,464 cores). Roadrunner delivers a peak performance of 1.7 petaflops per second (average

performance of 1.026 petaflops per second) and costs US$133 million. Roadrunner has a total of 6,912 Opteron processors (6,480 for computation and 432 for operation) and a total of 12,960 PowerXCell processors (12,960 PPE cores and 103,680 SPE cores) [13]. For conceptual modelling of multicore systems following crucial challenges should be addressed.

*Inter-core communication:* Most researchers use 2D mesh network for many-core architecture mainly because of its simplicity. However, there are other alternatives that should be tested.

*Defining the architecture:* The target architecture should be (re-)defined for modelling. The actual (existing or future) system may have millions of cores and other components. However, only some selected portions and/or some simplified portions should be sufficient to represent the system.

*Cache-memory organization:* Usually each core of a multicore system has private level-1 cache (CL1) and the processor may have shared and/or distributed level-2 and higher level caches (like CL2 and CL3). Cache memory is power-hungry and it increases unpredictability and data inconsistency. So, cache memory should be modelled carefully.

*Power and heat management:* Power consumption and heat dissipation are very crucial for multicore high performance systems where cores are built very close to each other.

Other important issues include multicore programmability and system cost.

## 4. PROPOSED PERFORMANCE MODELLING METHODOLOGY

As modern multicore/manycore networked computing systems are becoming more complex and expensive, developing conceptual models for such complex systems is in great need to analyse them. In this section, a methodology for conceptual modelling is described which is fast, easy, flexible, and suitable to develop the first stage (in the design cycle) conceptual model of multicore high performance systems. This methodology thinks complicated, but models simple; starts small, but adds more stuff as needed. Three major steps of this conceptual modelling methodology, data/knowledge acquisition, (system-to-) model abstraction, and model validation, are shown in Figure 1.

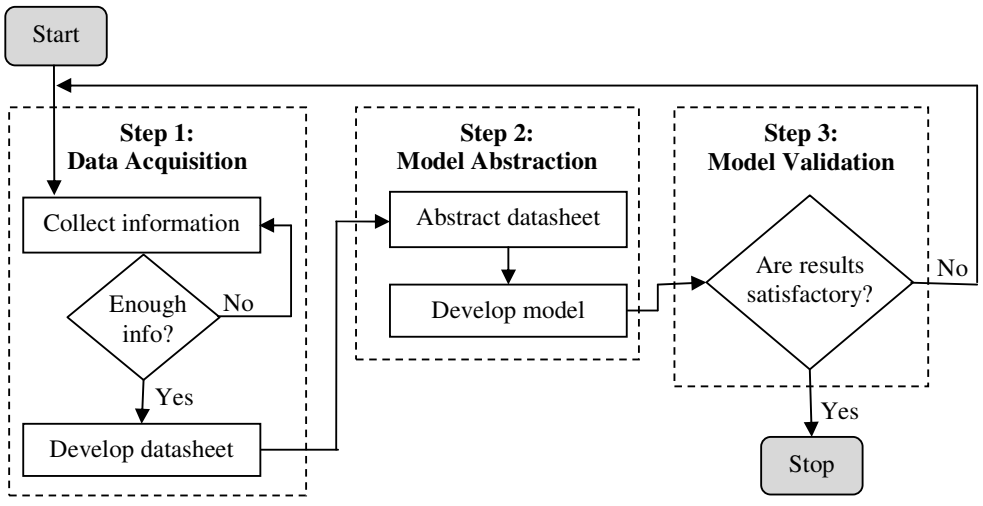


Figure 1.Major steps of the proposed performance modelling methodology

### 4.1. Data Acquisition

Data acquisition is the first step of this methodology. Two important tasks of this step are collecting information and developing data-sheet based on the requirements.

*Collecting information:* Collecting useful information about a high performance complex system is not easy. For future computing systems, sometime the end user's requirements are not even clear. For existing products, the owner company usually does not release important information.

*Developing data-sheet:* Once necessary (may not be sufficient) information is available, data-sheet is created applying the rules and requirements. This data-sheet should be updated as more information and/or requirements are available.

### 4.2. Model Abstraction

Model abstraction is the key step of this methodology. Two important tasks of this step are abstracting data-sheet and developing model.

*Abstracting data-sheet:* The most recent data-sheet is used for high level abstraction. While abstracting for the conceptual model, irrelevant and/or less important details are excluded. However, a sufficient level of accuracy is maintained.

*Developing model:* The abstracted data-sheet is mapped into a simulation model. Selecting and/or developing simulation tools become very important at this point. Also, the quality of the workload used in the simulation is important for the accuracy and completeness of the simulation results. The workload defines all possible scenarios and environmental conditions that the system-under-study will be operating under.

### 4.3. Model Validation

Validation of the model is the last and another important step of this methodology. Validation results primarily indicate if the model meets the requirements with an optimal choice (of the hardware and software used and/or proposed). If validation result is not satisfactory, above steps may be repeated as needed. Validation results can be used to determine if more functions/components need to be included in the conceptual model.

## 5. EVALUATION

In this work, we model and simulate multicore/manycore systems with up to 64 processing cores using the proposed modelling methodology and investigate the impact of cache parameters on performance and total power consumption. Discussed below are the assumptions, workloads, and input/out parameters related to the simulation program.

### 5.1. Assumptions

The following assumptions are made to model the target architecture and to run the simulation program.

a) Simulated multicore systems are considered as homogeneous. Two-level cache memory hierarchy is simulated, where level-1 caches are private to the cores and level-2 cache and main memory are shared.

b) Bus based network topology is used in this work.

c) Cores are grouped into clusters. Clusters are connected via shared level-2 cache. Inside each cluster, snoopy protocol is used for communication.

d) Tasks are evenly distributed among the cores.

e) Delay associate with the bus that connects CL1s and CL2 is 2/100th of the delay associated with the main memory.

f) Write-back policy (for write misses) and random cache replacement policy (for cache block replacements) are used.

## 5.2. Simulated Multicore Networked Architecture

In this work, we presentamodellingtechnique to simulate multicore/manycorenetworked systems to assess the performance and total power consumption. We simulate multicore architecture that includes up to 64 processing cores, level-1, and level-2 caches. Figure 2 shows such a system with 16 cores. As illustrated here, level-1caches (CL1s) are private to the cores and level-2 cache (CL2) is shared by the cores. We abstract the target architecture by considering all important components and ignoring non-relevant minor details.Cores are grouped together in a cluster and each cluster is connected to CL2 via a switch. CL2 has 4 communication ports (interfaces).Inside the cluster, snooping protocol is used. Shared CL2 is used for cluster-to-cluster communications.

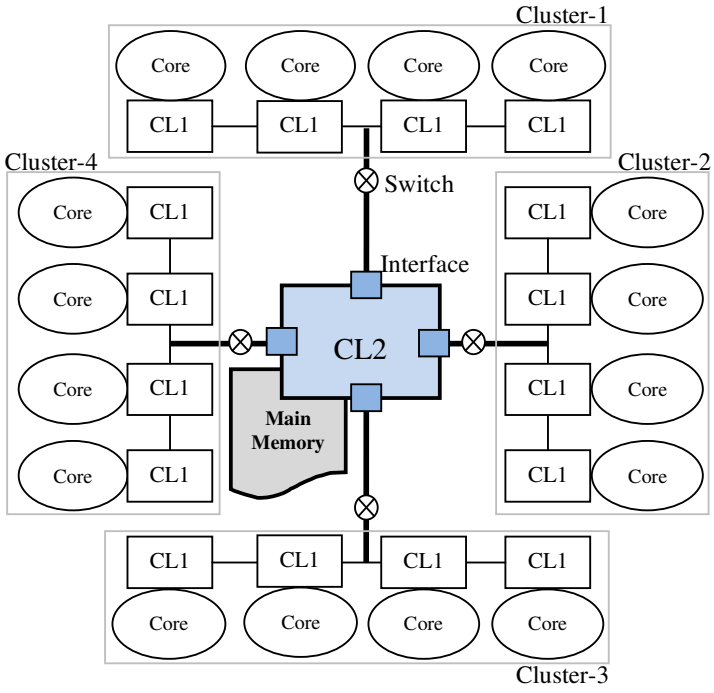


Figure 2.Simulated multicore networked architecture with sixteen cores

This networked architecture is good for small number of cores (suitable for most contemporary PCs and embedded systems).As the number of cores increases, the switches can be replaced by routers to accommodate more clusters. If the number of cores increases even more, various popular network topologies (like ring, mesh, and hypercube) can be used. We plan to explore some of those network topologies in our next endeavour.

### 5.3. Workloads

In this work, we use synthetic workload to run our simulation program. For the distribution of data requirements for various instructions, we assume that each processing core processes 60.0% of its instructions without requesting any external data. The remaining 40% requests for data are satisfied from the caches. Based on thesuggestion made in[23], we assume 95% level-1 cache hit. So, 38% (95% of 40.0) has a match in the level-1 cache and responds to the core with the data. Similarly, assuming 95% level-2 cache hit, 1.9% (95% of 2.0) has a match in the level-2 cache and responds to the core with the data while 0.1% has to take the additional steps to the main memory (MM) to get the data. We define task time be the time that one task needs to be processed. The processing time used for each architectural component is shown in Table 1.

Table 1. Processing time distribution.

| **Parameters** | **Values** |
|---|---|
| Task (completion) time | 10.0 time units |
| Core time | Task time * 0.6 |
| Main memory time | Task time * 0.4 |
| Bus time | Memory time * 0.02 |
| Level-1 cache time | Memory time * 0.04 |
| Level-2 cache time | Memory time * 0.08 |
| Others | Negligible |

### 5.4. Input and Output Parameters

Important input parameters used in the simulation include the number of cores, cache size, line size, and associativity level [see Table 2]. We keep CL2 cache sizes fixed at 4MB. As suggested in[12], we lock 25% of the cache size to see the impact of cache locking. Every time, we run our simulation program for 10000 units of simulation time.

Table 2.System input parameters.

| **Input Parameters** | **Values** |
|---|---|
| Number of cores | 4, 8, 16, 32, or 64 |
| CL1 size (KB) | 16, 32, 64, 128, 256 |
| CL2 size (MB) | 4 (fixed) |
| CL1/CL2 Line size (Byte) | 4, 8, 16, 32, or 32 |
| CL1/CL2 Associativity level (n-way) | 1, 2, 4, 8, or 16 |
| Cache locking at level-1 | 25% of the cache size |
| Cache locking at level-2 | 25% of the cache size |

We obtain mean delay per task and total power consumption by the system as the output parameters. Delay is the time between the start of execution of a task and the end [3][23].

### 5.5. Simulation Tool

VisualSim simulation package is used to model the selected architecture, run the simulation program, and collect results [23].

## 6. RESULTS AND DISCUSSION

In this work, we introduce a conceptual modelling technique to simulate multicore/manycore networked systems for performance analysis.We select a system that may have up to 64 cores;

each core has its own private level-1 cache; and level-2 cache is shared by all cores. Cores are grouped together as a cluster. Using the proposed conceptual modelling technique, we explore the impact of cache parameters on performance and total power consumption. We obtain results by varying the number of cores, line size, and associativity level. We present some important simulation results in the following subsections.

## 6.1. Number of Cores

We first simulate a system with 4 cores. Then we add more cores and simulate systems with 8, 16, 32, and 64 cores, respectively. We apply cache locking only at CL1 and only at CL2 (not at CL1 and CL2 at the same time). The average delay per task for no locking, level-1 locking, and level-2 locking are shown in Figure 3. Experimental results show that the mean delay per task decreases with the addition of cores for all three cases. From 4 to 16 cores, the decrement in mean delay per task is very sharp. Beyond 16 cores, the decrement is not significant.It is also observed thatperformance can be improved by applying level-1 and level-2 cache locking.Experimental results show that level-2 cache locking reduces the mean delay per task better than level-1 cache locking does.

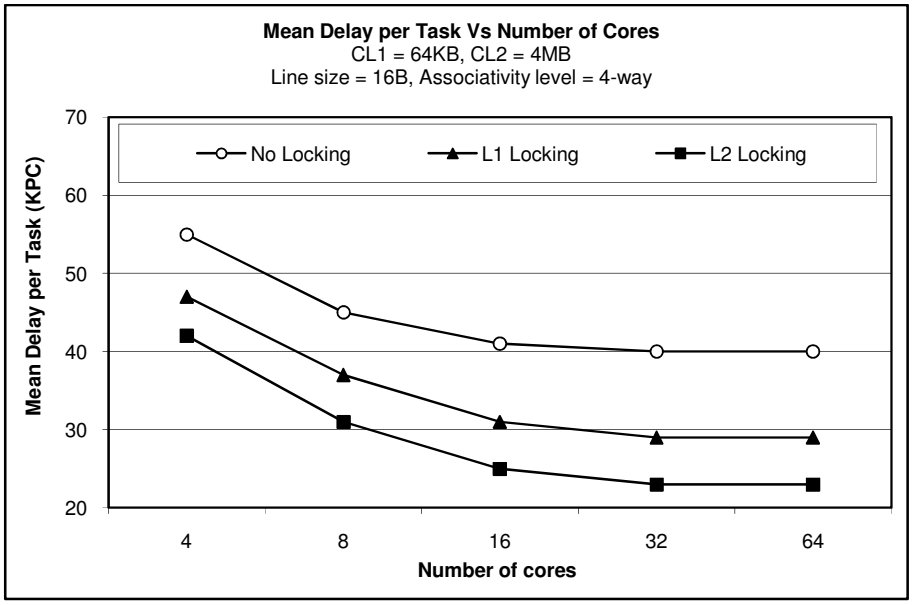


Figure 3.Mean delay per core versus number of cores

The impact of the number of cores and cache locking on total power consumption is shown in Figure 4. Experimental results show that total power consumption increases as more cores are added. However, the increment is not significant as the number of cores is higher. It is noted that the total power consumption can be reduced by applying level-1 and level-2 cache locking. Experimental results show that level-2 cache locking reduces the total power consumption better than level-1 cache locking does.

According to the experimental results, it is observed that cache locking at level-2 outperforms cache locking at level-1 (see Figures 3 and 4). This is primarily because of the synthetic workload used to run the simulation program. If application size is smaller than CL1 cache size (which is the case in this experiment), level-1 cache locking is not very efficient as the entire program fits inside CL1. In the case of program or data size smaller that CL1, cache locking at level-2 offers better performance/power ratio.

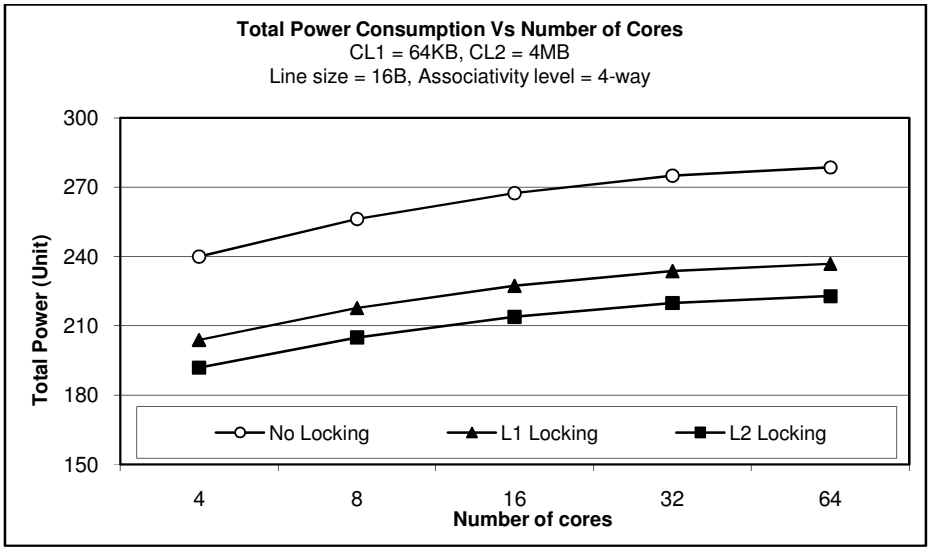


Figure 4. Total power consumptionversus number of cores

## 6.2. CL1 Size

Larger cache size may improve performance by reducing capacity and collision cache misses. However, too large a cache size may increase total power consumption but may not decrease cache misses. The mean delay per taskversus CL1 size for no cache locking, level-1 cache locking, and level-2 cache locking are shown in Figure 5. Experimental results show that performance can be improved by increasingCL1 sizefrom 16 KB.The mean delay per task starts decreasing sharply with the increase in CL1 size between 16KB and 64KB.Alsoobserved that performance can be improved (i.e., mean delay per task can be reduced) using cache locking for any CL1 cache size.

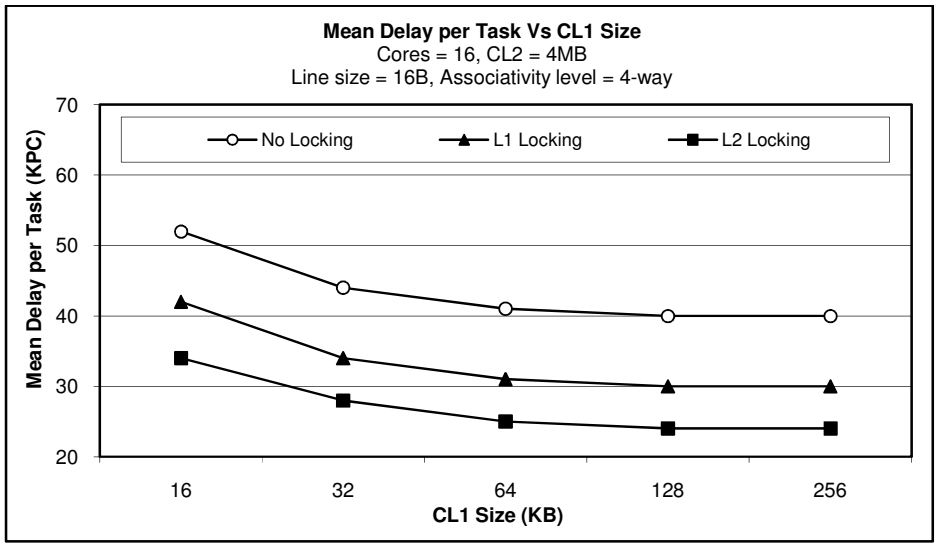


Figure 5.Mean delay per core versusCL1 size

Figure 6 illustrates the total power consumed for various CL1cache sizes. The total power starts decreasing sharply with the increase in CL1 size (between 16KB and 64KB) regardless of cache

locking. However for CL1 size beyond 64 KB,the total power consumption starts increasing with the increase in CL1 cache size.

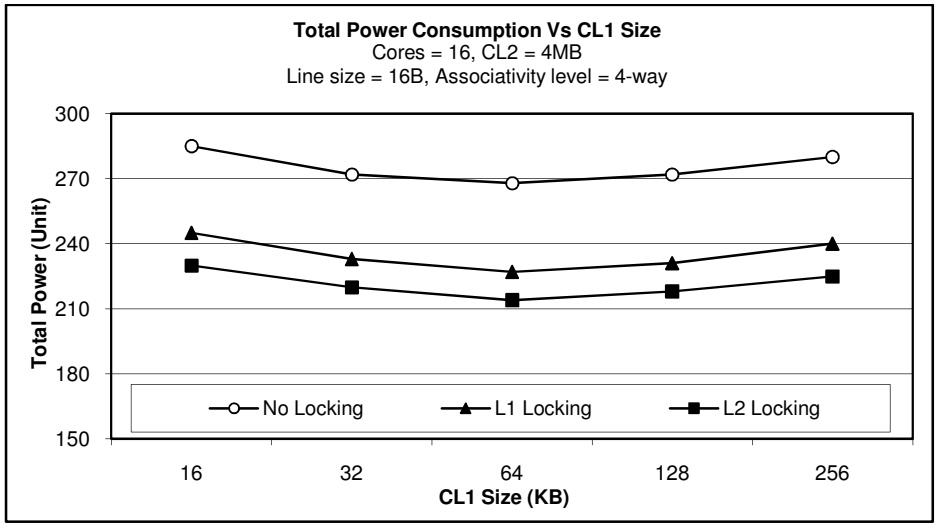


Figure 6. Total power consumptionversusCL1 size

## 6.3. Line Size

Larger line size may improve performance by reducing compulsory cache misses. However, too large a line size may increase capacity cache misses and that may reduce performance and increase power requirement. The mean delay per taskversus line size for no cache locking, level-1 cache locking, and level-2 cache locking are shown in Figure 7. Experimental results show that performance can be improved by increasingline sizefrom 4 Bytes.The mean delay per task starts decreasing sharply with the increase in line size between 4 Bytes and 16 Bytes. For line size greater than 32 Bytes, the mean delay per task starts increasing with the increase in line size due to cache pollution. It is alsoobserved that performance can be improved (i.e., mean delay per task can be reduced) using cache locking for any line size.

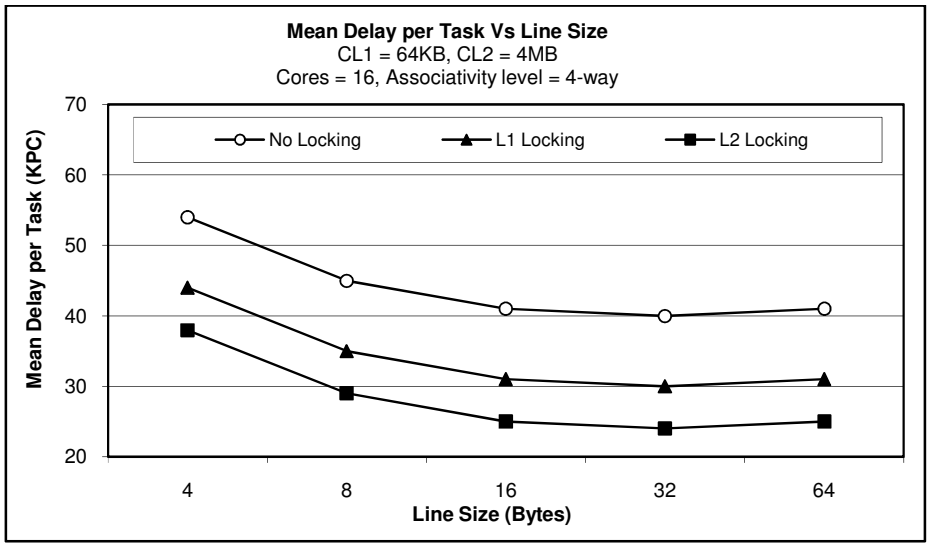


Figure 7.Mean delay per core versusline size

Figure 8 illustrates the total power consumed for various line sizes. The total power starts decreasing sharply with the increase in line size (between 4 Bytes and 16 Bytes) regardless of cache locking. Like the mean delay per task, the total power consumption starts increasing with the increase in line size due to cache pollution for line size greater than 32 Bytes.

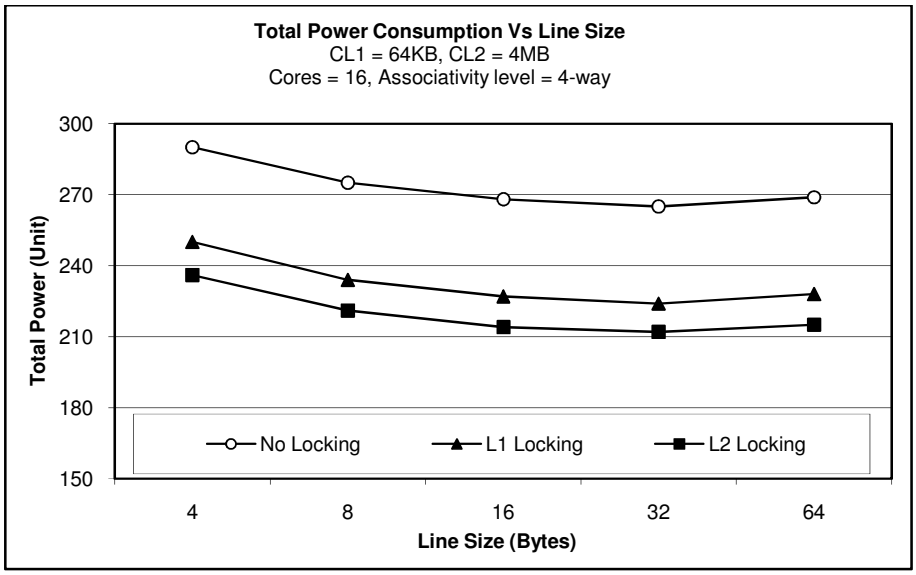


Figure 8. Total power consumptionversusline size

## 6.4. Associativity Level

Finally, we discuss the impact of various associativity levels on performance and total power consumption. Experimental results show that the mean delay per task can be decreased by increasingthe level of associativity (see Figure 9). From 1-way (direct mapping) to 4-way, the mean delay per task decreases sharply. Beyond 4-way, the decrement is not significant. For different associativity levels, the mean delay per task can be decreased by using cache locking and level-2 cache locking outperforms level-1 cache locking.

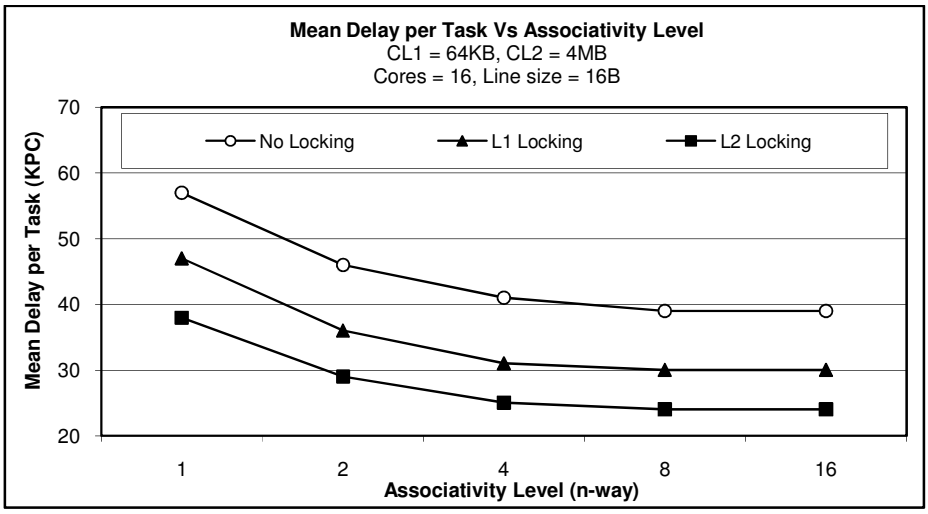


Figure 9.Mean delay per core versusassociativity level

Figure 10 shows the impact of various associativity levels and cache locking on total power consumption. The total power consumption decreasesas the level of associativity increases. However, only from 1-way (direct mapping) to 4-way, the decrement is significant. Again,level-2 cache locking offers better performance/power ratio than level-1 cache locking does.

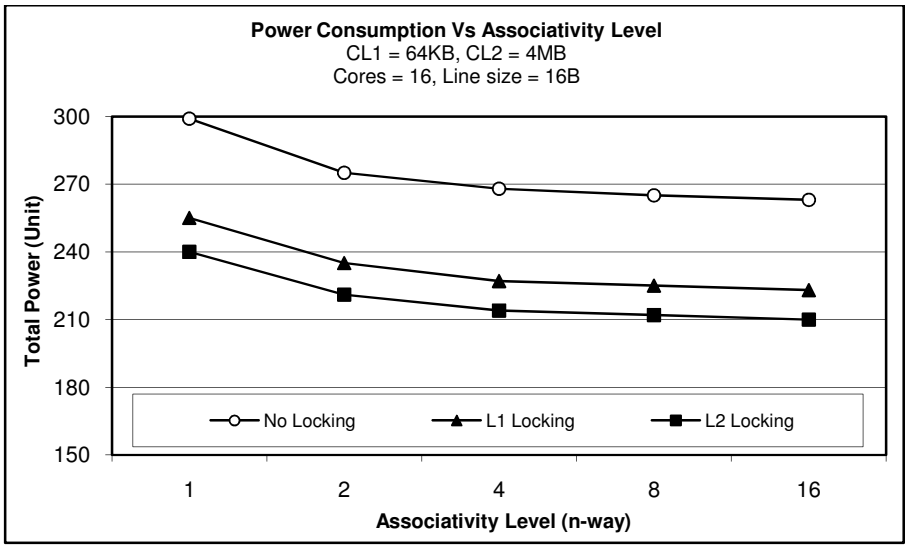


Figure 10.Total power consumptionversusassociativity level

## 7. CONCLUSIONS

Network topology and cache memory organization to make better communication in multicore/manycore systems is extremely important to achieve high performance/power ratio. Manycore systems like IBM Roadrunner have thousands of processing cores and are very expensive (cost up to millions of dollars). Conducting research on such a complexand expensive system is not possible for most researchers and organizations. Therefore, conceptual modelling technique is attracting researches from many institutions to conduct research on multicore/manycore networked systems in an inexpensive way.So far different modelling issues have been addressed; however, there is no standard in performance modelling of complex systems. In this work, we present a methodology for performance modelling for multicore/manycore networked systems. Various challenges and rewards of conceptual modelling are briefly discussed in this article. The conceptual modelling is very helpful to assess the validity of a situation not explicitly tested. It is very effective for future systems (simply because they do not exist) and for the multicore high performance systems (because they are complicated and expensive). We simulate networked systems up to 64 cores using the proposed performance modelling methodology. Each core has its own level-1 private cache. Level-2 cache is shared by all cores. Cores are grouped into clusters; clusters are connected via shared level-2 cache. Snoopy protocol is used for communication inside each cluster.

The proposed performance modelling methodologyis fast, simple, flexible, and suitable to develop the first stage (in the design cycle)conceptual model of multicore/manycorenetworked systems. This methodology models simple, but thinks complicated; it starts with a small model, but adds more details as needed. Three important features of the proposed methodology are: data acquisition in order to improve accuracy of the system, system-to-model abstraction to make the modellingmanageable, and model validation to make sure that the model meets the requirements.

Using the proposed modelling technique, we model a system with up to 64 cores. Experimental results indicate that for the synthetic workload used, mean delay per task decreases and total power consumption increases with the addition of cores. However, the decrement in the mean delay per task is very significant when compared with the increment in total power consumption. From Figures 3 and 4, mean delay per task decreases up to 23% but total power consumption increases only up to 6%. Simulation results also show that both level-1 and level-2 cache locking help reduce the mean delay per task and total power consumption. In this experiment, level-2 cache locking outperforms level-1 cache locking. This is because cache locking at level-1 is not efficient for this workload as itentirely fits in CL1 cache.We find the proposed performance modelling methodology easy and user-friendly.

We model systems up to 64 cores and simulate using synthetic workload in this work. We plan to model systems with thousands cores connected via ring, mesh, and hypercube network topologies and simulate them using real-time multimedia workloads in our next endeavour.

## REFERENCES


[1] Narayanaswamy, G., Balaji, P., &Feng, W., (2009) "Impact of Network Sharing in Multi-core Architectures", URL: www.mcs.anl.gov/~balaji/pubs/2008/icccn/icccn08.multicore.pdf

[2] Duranton, M., (2006)“The Challenges for High Performance Embedded Systems”, *9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools (DSD 2006)*, pp3-7.

[3] Asaduzzaman,A.& Mahgoub, I., (2006)“Cache Modelling and Optimization for Portable Devices Running MPEG-4 Video Decoder”, *Multimedia Tools and Applications (MTAP’06)*, pp239-256.

[4] Haritan,E.,Yagi,H., Et al, (2008)“Multicore design is the challenge! What is the solution?”,*45th ACM/IEEE Design Automation Conference Proceedings*, pp128-130.

[5] Mittal, M., (2009) "Optimizing Multicore for Networking Applications", URL: http://www.ecnmag.com/Articles/2009/12/optimizing-multicore/

[6] Grover, S., Dhanotia, A., & Byrd, G.T., (2011) "A Canonical Multicore Architecture for Network Routers" 2011 Seventh ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), pp134-144.

[7] Robinson, S., (2006)“Conceptual Modelling for Simulation: Issues and Research Requirements”, *Proceedings of the 2006 Winter Simulation Conference*.

[8] Borah, J., (2009)“Simulation Conceptual Modelling Study Group Gets Rolling”, URL: www.sisostds.org

[9] Pace, D.K., (2000)“Ideas about Simulation Conceptual Model Development”, *Johns Hopkins APL Technical Digest (21-3)*, pp327-336.

[10] Cyre, W.R., (1999)“Conceptual modelling and simulation”, *International Conference on Computer Design (ICCD’99)*, pp293-296.

[11] Chwif,L.,Barretto,M.R.P., & Paul, R.J., (2000)“On Simulation Model Complexity”, *Proceedings of the 2000 Winter Simulation Conference*, IEEE, Piscataway, NJ, pp449-455.

[12] Asaduzzaman,A.& Sibai, F.N., (2010)“Improving Cache Locking Performance of Modern Embedded Systems via the Addition of a Miss Table at the L2 Cache Level", *Journal of Systems Architecture (JSA)*.

[13] Wikipedia, (2012) IBM's Roadrunner is the first Petascale computer. URL: http://en.wikipedia.org/wiki/IBM_Roadrunner

[14] Shannon, R.E., (1975)"Systems Simulation: The Art and Science",*Prentice-Hall, Englewood Cliffs, NJ, ISBN-13: 978-0138818395*.

[15] Zeigler, B.P., (1976)"Theory of Modelling and Simulation", *Wiley, New York, NY, ISBN-13: 978-0127784557*.

[16] Balci,O.& Nance, R.E., (1985)"Formulated Problem Verification as an Explicit Requirement of Model Credibility", *Simulation, Vol. 45, No. 2*, pp76-86.

[17] Sevinc, S., (1990)"Automation of Simplification in Discrete Event Modelling and Simulation", *International Journal of General Systems*, Vol. 18, pp125-142.

[18] Pidd, M., (1999), "Just Modeling Through: A Rough Guide to Modeling", INTERFACES Vol. 29, No. 2, pp118-132.

[19] Davies, R., Roderick, P., &Raftery, J., (2003) "The Evaluation of Disease Prevention and Treatment using Simulation Models", European Journal of Operational Research, pp53-66.

[20] Gamatie,A.,Rutten,E.,Huafeng,Y.,Boulet,P., &Dekeyser, J.L., (2008),"Modelling and Formal Validation of High-Performance Embedded Systems",*International Symposium on Parallel and Distributed Computing (ISPDC '08)*, pp215-222.

[21] Gittell,J.H.,Seidner,R.,&Wimbush, J., (2010)"A Relational Model of How High-Performance Work-Systems Work",*Organization Science*,pp21:490-506.

[22] Zhou,N.F., Sato,T., &Hasida, K., (2011)"Toward a High-performance System for Symbolic and Statistical Modelling"URL: www.sci.brooklyn.cuny.edu/~zhou/papers/prism.pdf

[23] VisualSim, (2012)"VisualSim – a simulation tool from Mirabilis Design, Inc."URL: www.mirabilisdesign.com

**Abu Asaduzzaman** received the Ph.D. and M.S. degrees, both in computer engineering, from Florida Atlantic University (FAU), Boca Raton, Florida in 2009 and 1997, respectively, and the B.S. degree in electrical engineering from Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh in 1993. Currently, he is working as an Assistant Professor in the department of electrical engineering and computer science at Wichita State University in Wichita, Kansas. He has published several journal and conference papers and book chapter out of his research work. His research interests include computer architecture, embedded systems, parallel computing, and performance evaluation. Mr. Asaduzzaman is a member of the IEEE and other prestigious the honor societies. He served as a TPC member of IEEE IPCCC 2011 conference and as reviewer of NSF TUES (CS) and GRFP programs. He is currently serving as an IPC member of IEEE ICCIT 2012 conference.