MuMMI_R: Analyzing and Modeling Power and Time under Different Resilience Strategies

Xingfu Wu and Valerie Taylor
Dept. of Computer Science & Engineering
Texas A&M University, College Station, TX
Email: {wuxf, vtaylor}@ tamu.edu

Zhiling Lan
Department of Computer Science
Illinois Institute of Technology, Chicago, IL
Email: lan@iit.edu

Abstract

While reducing execution time is still a major objective for high performance computing, future systems and applications will have additional power and resilience requirements that represent a multidimensional tuning challenge. In this poster we present MuMMI_R: analyzing and modeling power and time under different resilience strategies. We use FTI (Fault Tolerance Interface) library to conduct our experiments to evaluate how using FTI with checkpoints of different levels at different frequencies impacts the power consumptions at different node components (Node, CPU, Memory, Disk and Network) and energy consumptions of the MPI memory benchmark STREAM on three different architectures IBM BG/Q, Intel Haswell and AMD Kaveri. Our experimental results provide a better understanding the tradeoffs among runtime, power and resilience.

1. Introduction

Real-world scientific applications often rely on resilience techniques to successfully finish the long executions. While fault tolerance methods and power capping techniques continue to evolve, tradeoffs across execution time, power efficiency, and resilience strategies are not well understood. Existing fault tolerance studies mainly focus on the tradeoffs between execution time/overhead and resiliency, whereas most power management studies focus on the tradeoffs between execution time and power. Understanding the tradeoffs among these three factors (i.e., execution time, power, and resilience) is crucial due to the fact that future machines will be built under both reliability and power constraints.

Currently, FTI (Fault Tolerance Interface) library [1] is available to provide the means to perform fast and efficient application-level checkpointing, and it leverages local storage plus data replication and erasure codes to provide several levels of reliability and performance. In this work we use FTI to conduct our experiments to evaluate how using FTI with checkpoints of different levels at different frequencies impacts the power consumptions at different node components (Node, CPU, Memory, Disk and Network) and energy consumptions of the MPI memory benchmark STREAM (MPI version) [3] on three different architectures IBM BG/Q, Intel Haswell and AMD Kaveri.

2. MuMMI_R Framework

Figure 1 shows the MuMMI_R framework. For a MPI application executed on a power-aware system, we use MuMMI [2] to collect runtime, power (node, CPU and memory), and performance counter data under different resilience strategies, and automatically upload the data to the database. All performance counters are normalized using the total cycles of the execution to create performance event rates for each counter. Then, using Spearman correlation and principal component analysis (PCA), we identify the important performance counters for the four metrics - runtime, system power, CPU power, and memory power, and use non-negative multivariate regression analysis to generate the four models based upon the set of important performance counters and CPU frequencies under different resilience strategies. Then we can identify which resilience strategy will result in the lowest energy consumption.

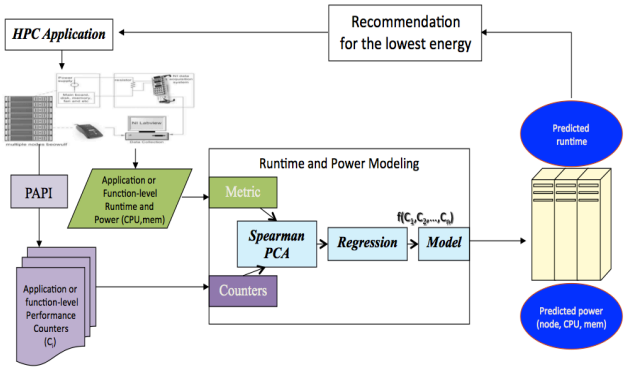

Figure 1. MuMMI_R Framework

2. Three Architectures

We conduct our experiments on three different architectures: IBM BG/Q, Intel Haswell and AMD Kaveri shown in Table 1. Each BG/Q node has 16 CPU cores, shared L2 cache of 32MB and 16GB memory; each Haswell node has 32 CPU cores, shared L3 cache of 40MB and 128GB memory; each Kaveri node has 4 CPU cores and 8 GPU cores, shared L2 cache of 2MB and 16GB memory. In this work, we use only CPU cores for our study. We port MuMMI [2] to support the power measurements (Node power, CPU power, Memory power and Network power) using MonEQ [4] on BG/Q and the power measurements (Node power, CPU power, Memory power and Disk power) using PowerInsight [5] on Intel Haswell and AMD Kaveri systems, and use MuMMI to collect power and performance data on them.

3. Using the Memory Benchmark STREAM

Because of the memory issues we found in the previous sections, we use the memory benchmark STREAM (MPI version) [3] to address the power and energy impacts of FTI. For the benchmark, we set the number of runs for each kernel 5000, and adjust the STREAM_ARRAY_SIZE to 4 times the size of the last level cache (128M(Million) for BG/Q, 160M for Haswell and 8M for Kaveri).

Table 1. On BG/Q

Frequency	Runtime	Node Power	CPU Power	Memory Power	Network Power	Energy
Original	216	53.56	32.51	7.76	1.86	11568.96
ckp(1,2,3,4)	268	53.35	32.49	7.59	1.86	14297.80
ckp(1,3,5,7)	254	53.46	32.54	7.64	1.86	13578.84
ckp(2,3,4,5)	256	56.39	34.60	8.52	1.85	14435.84
ckp(2,4,6,8)	227	54.01	32.20	8.70	1.89	12260.27
ckp(3,4,5,6)	229	57.31	35.17	8.66	1.88	13123.99
ckp(3,5,7,11)	229	56.54	34.64	8.63	1.85	12947.66
ckp(3,5,7.9)	231	57.25	35.12	8.65	1.88	13224.75
ckp(4,5,6,7)	217	57.33	35.13	8.71	1.88	12440.61

Table 1 shows the average power for 8 different checkpointing frequencies. Although we set the STREAM_ARRAY_SIZE to 128M on BG/Q, we observe that, comparing with the power consumptions for the original code, the multilevel checkpointing causes not only the increase in time but also the power consumption increase in the node, CPU, Memory and Network. Although the default FTI checkpointing is ckp(3,5,7,11), from the table, we find that the checkpointing ckp(2,4,6,8) results in the lowest energy consumption among the different checkpointings although it does not have the smallest runtime and node power consumption.

Table 2. On Haswell

Frequency	Runtime	Node Power	CPU Power	Memory Power	Disk Power	Energy
Original	800	311.26	230.34	64.83	2.32	249008.00
ckp(1,2,3,4)	1832	219.24	161.42	41.69	2.31	401647.68
ckp(1,3,5,7)	1848	230.29	170.83	43.32	2.33	425575.92
ckp(2,3,4,5)	1618	243.08	180.58	46.36	2.32	393303.44
ckp(2,4,6,8)	1311	261.27	193.95	51.17	2.33	342524.97
ckp(3,4,5,6)	1460	253.43	187.23	50.04	2.32	370007.80
ckp(3,5,7,11)	1396	261.87	194.46	51.28	2.33	365570.52
ckp(3,5,7.9)	1358	265.54	197.50	51.90	2.32	360603.32
ckp(4,5,6,7)	1450	259.06	191.50	51.39	2.33	375637.00

Table 2 shows the average power for 8 different checkpointing frequencies. We observe that, comparing with the power consumptions for the original code, the multilevel checkpointing causes not only the increase in time but also the power consumption decrease in the node, CPU and Memory. It is interesting to see that the checkpointing results in less power consumptions in Node, CPU and Memory because of the frequent checkpointing interrupts. And high frequency checkpointing causes less memory power consumption. From the table, we find that the checkpointing ckp(2,4,6,8) results in the lowest energy consumption among the checkpointings although it has very high node power consumption.

Table 3. On Kaveri

Frequency	Runtime	Node Power	CPU Power	Memory Power	Disk Power	Energy
Original	486	78.17	44.03	16.45	0.85	37990.62
ckp(1,2,3,4)	535	77.83	44.15	16.01	0.82	41639.05
ckp(1,3,5,7)	530	76.51	43.36	15.59	0.88	40550.30
ckp(2,3,4,5)	527	76.90	43.51	15.69	0.86	40526.30
ckp(2,4,6,8)	502	77.73	43.67	16.39	0.84	39020.46
ckp(3,4,5,6)	519	77.24	43.88	15.76	0.84	40087.56
ckp(3,5,7,11)	509	77.54	43.71	16.13	0.82	39467.86
ckp(3,5,7.9)	509	77.42	43.89	15.90	0.84	39406.78
ckp(4,5,6,7)	518	76.72	43.47	15.70	0.87	39740.96

Table 3 shows the average power for 8 different checkpointing frequencies. We observe that, comparing with the power consumptions for the original code, the multilevel checkpointing causes the increase in time but the power consumption stays flat for the node, CPU, Memory and Disk. From the table, we find that the checkpointing ckp(2,4,6,8) results in the lowest energy consumption among the checkpointings because of the smallest runtime.

4. Summary

In this work, we extended the MuMMI to examine the tradeoffs among resilience, execution time, power and energy of STREAM benchmark on three different architectures. Our experimental results shows that the multilevel checkpointing FTI with frequency of 2 minutes at level 1 (L1), 4 minutes at level 2 (L2), 6 minutes at level 3 (L3) and 8 minutes at level 4 (L4) results in the lowest energy consumption across the three architectures. In the future, we will extend the MuMMI to model the tradeoffs between execution time, power, energy and resilience for various application-system configurations

References

[1] L. Bautista-Gomez, D. Komatitsch, N. Maruyama, S. Tsuboi, F. Cappello, and S. Matsuoka. FTI: High performance fault tolerance interface for hybrid systems. In *SC2011, Seattle, WA*, 2011. Also see FTI: Fault tolerance interface. *http://leobago.github.io/fti/*.

[2] X. Wu, V. Taylor, C. Lively, H. Chang, B. Li, K. Cameron, D. Terpstra and S. Moore, MuMMI: Multiple Metrics Modeling Infrastructure (Book Chapter), *Tools for High Performance Computing 2013*, Springer, 2014. Also see http://www.mummi.org.

[3] STREAM memory benchmark, http://www.cs.virginia.edu/stream

[4] S. Wallace, V. Vishwanath, S. Coghlan, J. Tramm, Z. Lan, and M. E. Papka, Application Power Profiling on Blue Gene/Q, *IEEE Conference on Cluster Computing*, 2013.

[5] J. H. Laros III, P. Pokorny and D. DeBonis, PowerInsight – A Commodity Power Measurement Capability, *International Green Computing Conference*, 2013.