Power-Aware Heterogeneous Node Assembly

Preprint · February 2019

DOI: 10.13140/RG.2.2.11758.25926

4 authors, including:

Bilge Acun
University of Illinois, Urbana-Champaign
28 PUBLICATIONS **434** CITATIONS

Power-Aware Heterogeneous Node Assembly

Bilge Acun, Alper Buyuktosunoglu, Eun Kyung Lee, Yoonho Park
bilge.acun2@ibm.com, {alperb, eunkyung.lee, yoonho}@us.ibm.com
IBM T. J. Watson Research Center, NY, USA

ABSTRACT

To meet ever increasing computational requirements, supercomputers and data centers are beginning to utilize *fat* compute nodes with multiple hardware components such as manycore CPUs and accelerators. These components have intrinsic power variations even among same model components from same manufacturer. In this paper, we argue that node assembly techniques that consider these intrinsic power variations can achieve better power efficiency without any performance trade off on large scale supercomputing facilities and data centers. We propose three different node assembly techniques: (1) Sorted Assembly, (2) Balanced Power Assembly, and (3) Application-Aware Assembly. In Sorted Assembly, node components are categorized (or sorted) into groups according to their power efficiency, and components from the same group are assembled into a node. In Balanced Power Assembly, components are assembled to minimize node-to-node power variations. In Application-Aware Assembly, the most heavily used components by the application are selected based on the highest power efficiency. We evaluate the effectiveness and cost savings of the three techniques compared to the standard random assembly under different node counts and variability scenarios.

1. INTRODUCTION

High Performance Computing (HPC) systems utilize *fat* compute nodes with a variety of components. 138 systems in the Top-500 supercomputers November 2018 list contains accelerators or co-processors [1]. Oak Ridge National Laboratory's (ORNL) Summit supercomputer, which has become the top system with over 200 Petaflop/s peak performance in November 2018, and Lawrence Livermore National Laboratory's (LLNL) Sierra supercomputer, which has become second with 125 Petaflop/s, are examples of such systems with fat nodes. A single compute node in the Summit supercomputer contains two CPUs, six GPUs, two memory units, and two network adapters [2]. The architecture of the Summit compute node is shown in Figure 1. Moreover, cloud data centers show similar architectural trends as the increasing popularity of machine learning applications drives the need for computational power from accelerators. Some of Facebook's servers used for machine learning tasks contain eight GPUs per node [3]. Of course, power consumption of these large-scale systems remains a growing challenge.

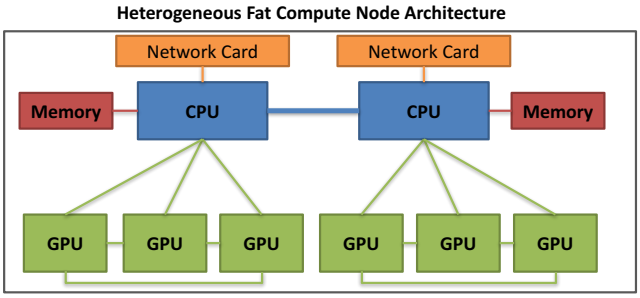

Figure 1: A *fat* compute node architecture.

Many studies have shown that there are intrinsic manufacturing differences causing power variations between processors of the same model [4, 5, 6, 7]. As integrated-circuit technology continues to scale, within-die and across-die process variation causes significant differences in performance and power. Considering a compute node in a data center, such manufacturing differences that cause power variation in CPUs, also exist in GPUs, memory components, network cards, and disks. However, nodes are randomly assembled without taking into account these variations. In future generation architectures, variability of the CMOS based circuits are expected to increase. Harnessing the variability in large-scale systems enables obtaining more power-efficient and predictable designs. In this paper, we propose simple and powerful power-aware node assembly techniques that can increase the overall power efficiency of a data center, mitigate the power variations among nodes, and help mitigate the side effects of power variations that can affect application performance.

The main contributions of this paper are as follows:

- We present an analysis of manufacturing-related variability using parametric hardware data, power and temperature variation measurements of the compute nodes using 100+ node components.
- We propose three different power-aware physical node assembly techniques and compare them with the current practice of random assembly. The three techniques are:

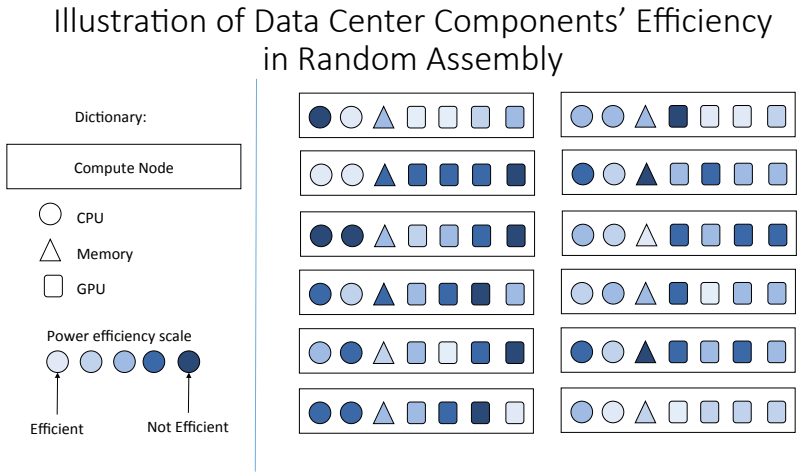

Figure 2: Illustration of a data center with nodes assembled randomly. *Performance per Watt* can be used as a power efficiency metric.

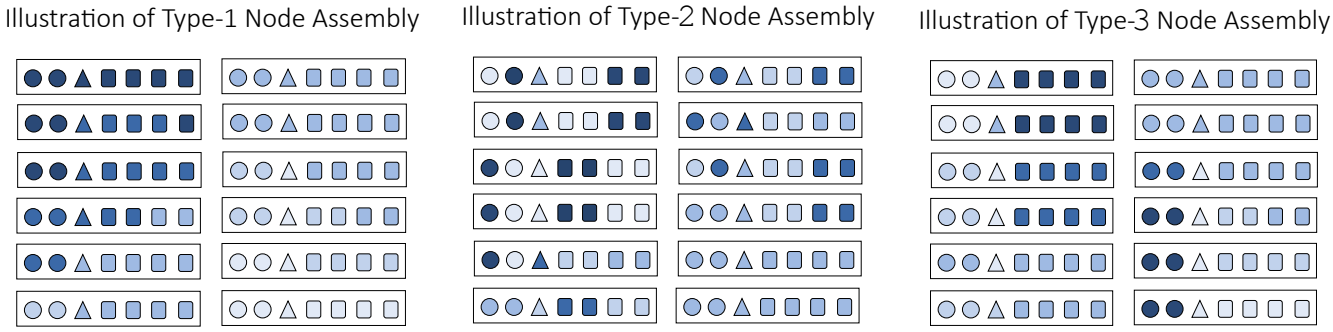

Figure 3: Illustration of the three proposed node assembly techniques. (a) Sorted assembly, (b) Power balanced assembly, (c) Application-aware assembly

1) Sorted Assembly: Node components are categorized into groups according to their power efficiency, and components from the same group are assembled into a node, nodes are sorted into racks. This method increases power efficiency if a data center is not at full utilization.

2) Balanced Power Assembly: Components are assembled in a balanced manner to minimize node-to-node power variations. This method makes the average performance and power of each node similar and more predictable by reducing the inter-node variations.

3) Application-Aware Assembly: Components which application use most heavily are selected to use the most power efficient components. Other node components which the application do not use as heavily can be less power efficient. For example, a CPU-intensive application will be assigned to use the most power efficient (low power) CPUs. This technique reduces power consumption with potential benefits on performance.

- We provide an energy cost reduction evaluation of the node assembly techniques under different node counts and variability scenarios.

To the best of our knowledge, this is the first paper that describes different power-aware compute node assembly mechanisms in the literature. None of these assembly techniques creates performance effect on applications under nominal frequency settings.

The rest of the paper is organized as follows. Section 2 provides background information. Section 3 provides measurements and analysis of power variation. Section 4 describes the proposed node assembly techniques, Section 5 evaluates them at exascale using different variability scenarios. Section 6 presents related work. Finally, Section 7 concludes the paper.

2. BACKGROUND: SOURCES OF THE POWER VARIATION

There are two main types of variations affecting microprocessors: (1) environmental factors and (2) physical factors. Environmental factors happen during the

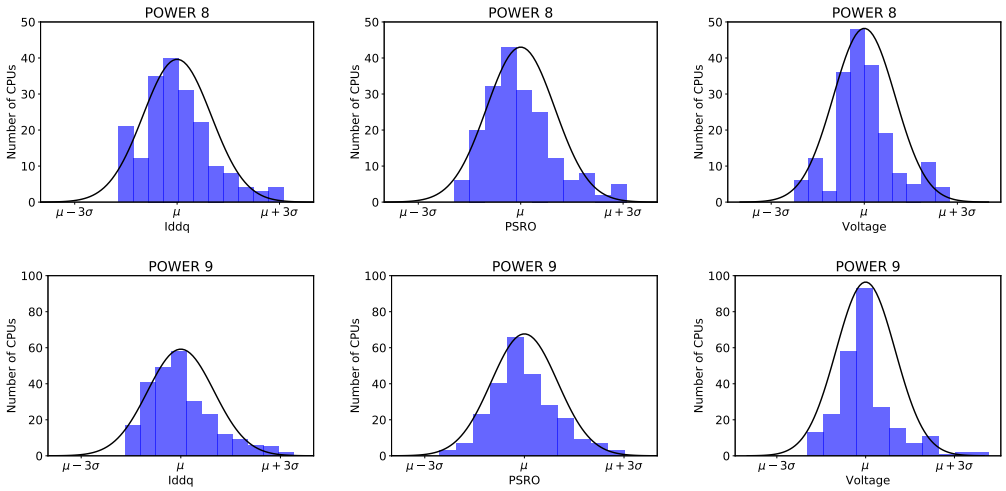

Figure 4: Parametric data of 190 IBM POWER8 and 252 POWER9 chips showing the distribution of quiescent current (Iddq), PSRO and supply voltage. (Note that μ and σ of each plot are different from each other.)

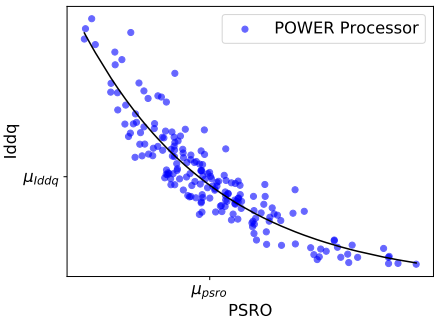

Figure 5: Parametric data of 190 IBM POWER8 chips showing the correlation between quiescent current (Iddq) and PSRO.

activity of the chip due to variations in power supply, switching activity and temperature of the cores. On the other hand, physical factors occur while manufacturing of the chips due to the limitations in processing and masking resulting in different parametric values that are permanent [8]. This permanent process variation, we argue, poses the need or the opportunity to do variation-aware node assembly.

The decreasing size of Complementary Metal-Oxide-Semiconductor (CMOS) transistors and lower voltage thresholds for energy efficient chip design are two major causes of manufacturing-related process variation. This process variation causes yield loss and processors that do not satisfy performance and power requirements must be discarded [9]. However, even processors that make it to production have variability in their power, operational frequency, and temperature.

Process variation effects both static and dynamic power of the chips due to two different reasons: (1) variations in leakage current and (2) gate delays. Decreased device and interconnect dimensions contribute to the first type of variation. For example, distribution of relatively small dopant ions creates differences in transistor threshold voltages which become significant as the device dimensions get smaller [10]. Variations in leakage current, cause variations in static (or idle) power (i.e., power dissipation due to sub-threshold leakage) [11, 12]. The second type of variation is caused by the variation in the gate length of the transistors, also commonly known as across chip line-width variation (ACLV). Uneven etching due to local shape density, lithographic distortions and etching variations during chemical mechanical polishing process are some of the underlying reasons of this variation. Gate delay variations cause variations in the dynamic power of chips (i.e., the power dissipation due to charging and discharging of load capacitance).

Both performance and power consumption in a processor have its required limits (i.e., the processor needs to sustain a frequency, and it should not exceed a power limit known as Thermal Design Power (TDP)). It has been shown that the variation can be up to 30% in frequency and 20x in leakage power [4]. Such high variation in the manufacturing process with leaky and slow chips have introduced the concept of *voltage binning*. *Voltage binning* refers to the technique of adjusting the voltage of the chip to satisfy the power or performance of the chip since changing the voltage of the chip effects both the performance and the power of the chip [13]. With this technique, some of the leaky or slow chips can be converted into "good" chips by respectively lowering or raising their supply voltage. Since leaky chips require lower voltage levels and slow chips require higher voltage levels, chips that are very leaky and very slow have to be thrown out. Different binning techniques have been proposed to increase the yield and the profit of chips [13, 14]. However, there are still variations among

already sorted "good" processors of the same type.

PSRO (performance sort/screen ring oscillator) and Iddq (quiescent current) parametric values help to understand the intrinsic variations across several processors [15, 10]. The PSRO reading provides a measure of processor performance regarding time delay in CMOS circuits. Higher PSRO values imply slower processors. Iddq reading provides a measure of the leakage current. Higher Iddq values imply possible physical defects and leakier processors. Figure 4 shows the distribution of PSRO, Iddq for 190 POWER8 and 252 POWER9 processors in our production cluster both exhibiting a Gaussian distribution. The reverse correlation between Iddq and PSRO is shown in Figure 5. So processors with higher PSRO values are slower but less leaky, and vice versa. If this distribution included the discarded processors, there would be "too leaky" processors on the left-tail and "too slow" processors on the right tail in Figure 5. The distribution of supply voltage levels (V_{dd}) of the same processors when operating at a fixed frequency level also shows a similar distribution. The processors in the cluster show behavior in a wide range of process window (+/- 3σ (standard variation)). These variations in leakage current and voltage naturally result in power variation among same model production chips. Other node components also show different power distributions. We analyze them next.

3. POWER VARIATION ANALYSIS OF NODE COMPONENTS

In this section, we describe our experimental setup and analyze power and temperature variation of different node components when they are idle and active. Using this analysis, we determine what metric to use when sorting the processors for node assembly and how it effects temperature of the chips.

3.1 Experimental Setup

We use an IBM Minsky cluster where each compute node in the cluster has two POWER8 CPUs with ten cores each and four NVIDIA Tesla P100 GPUs. The node architecture is similar to the Summit and Sierra supercomputers as shown earlier in Figure 1. Note that this is a cluster with standard binned hardware. The system contains about 100 nodes in total. Due to the availability of the cluster, we were not able to perform all experiments at the largest scale. However, the fitted distributions we use in extrapolating to larger system sizes give close fit to the collected data at the small scale. When making comparison studies, i.e., comparing idle IPS and idle temperature of chips, we used the same set of chips.

Voltage, Power, Temperature Measurements: We use the open-source AMESTER (Automated Measurement of Systems for Temperature and Energy Reporting) tool in order to make voltage, power and temperature measurements [16]. The tool can provide power, voltage measurements every 250 µs and temperature measurements every 2 ms. AMESTER also provides Instructions Per Second (IPS) measurements with 2 ms granularity. It connects to Baseboard Management Controller (BMC) of the processor in order to make measurements from a management node, therefore it does not use the CPU cycles of the measured system. AMESTER reports GPU power as a sum of the two GPUs per socket. In order to get GPU power per device individually, we use the NVIDIA System Management Interface (`nvidia-smi`).

All chip temperatures provided are the average temperatures of all cores in a chip. All power data shown throughout the paper is average stable power consumption of the benchmarks and none of the benchmarks has phases that cause large power differences throughout the run.

Applications: We use a set of benchmarks that exhibit different characteristics for performing our experiments. These consist of compute-intensive applications *MaxPow and DGEMM,* a memory-intensive application *Stencil3D*, and a communication-intensive application *kNeighbor*. *MaxPow* is a CPU-stress test benchmark running assembly instructions in a loop aiming to utilize CPU to its full capacity using the methodology described by Bertran et al [17]. *DGEMM* is a simple three-loop double precision matrix multiply code. We use a matrix size that fits in the last level cache. *kNeighbor* is a micro-benchmark with a near-neighbor communication pattern where each object exchanges fixed-sized messages with a fixed set of fourteen neighbors in every iteration. We use a message size of 16 KB. *Stencil3D* is a seven-point stencil application based on a 3D grid using Jacobi kernel. We use 30 GB grid size per chip.

3.2 Variation Analysis

In this section, we analyze idle and active power consumption of node components. Idle power measures static or leakage power of the chip. Active power measures total chip power when running a workload, sum of static and dynamic power of the chip. These measurements help us determine what metric to use when sorting the processors during assembly. We acknowledge that variation occurs not only from chip-to-chip but also among the cores of a chip (within-die) as well, however core-level power measurement capability is not provided in many processors including the POWER8 processors we use. Therefore, we consider core-to-core variation beyond the scope of this paper.

We first analyze idle power variation of the node components. Figure 6 shows the idle (static) power distribution of node components using 41 compute nodes and 82 node components in our cluster. Chips show 49%, memory units show 20%, GPUs show 18%, and nodes show 18% variation in their idle power – ranging from 570 Watts to 680 Watts. The Gaussian distribution in the supply voltage levels we show earlier in Section 2 is clearly reflected in the power consumption as well. To make sure the nodes are idle when we collect this data, we allocate the nodes exclusively and measured instruc-

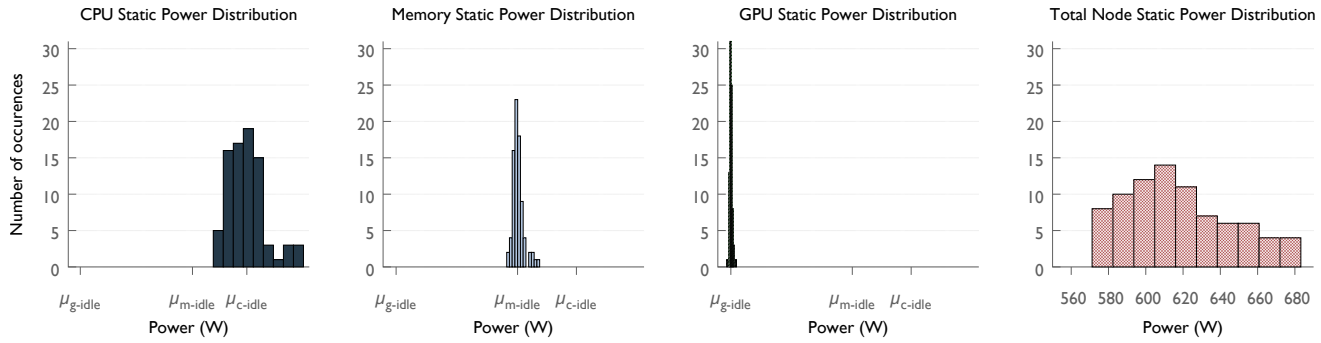

Figure 6: Static (idle) power distribution of node components. μ_{c-idle}, μ_{m-idle}, μ_{g-idle} represents the mean CPU, memory, GPU power respectively. (Note that X axes ranges are different in the total node distribution plot.)

tions per seconds (IPS). As shown in Figure 10, idle IPS levels have some variation, but it is not significant and not correlated with the power consumption of the chips.

Second, we analyze the active power consumption of the processors. Figure 7 shows the active power of the chips running three different benchmarks. The normal distribution curve seen in the idle power distribution is prominent when running all of the benchmarks too. The power variation is 28% for DGEMM, 16% for KNeighbor, 20% for Stencil3D benchmarks. Although the power distribution of the applications are shifted or slightly different, the chips that consume high power versus low power are consistent throughout different benchmarks. Note that these applications do not have imbalanced load behavior. Applications with significant load imbalance might show different distributions. Since most applications aim to achieve load balance for performance reasons, we do not consider applications with significant load imbalance in this paper.

Finally, we need to compare idle and active chip power distributions, in order to determine what power efficiency metric to use when sorting the processors for different node assembly strategies. Figure 8 shows the correlation between the idle and active power consumption of the chips using the MaxPow benchmark. There is no significant correlation between them with R-Squared value of 0.189. The chips that have high (or low) idle power do not necessarily have high (or low) active power. The average active power of the chips is equal to 1.8 times the average idle chip power. So static power is a dominant factor in total power of the chip when it is active. The trend from the previous studies show that static power has been increasing as the gate sizes become smaller [18]. However idle power alone does not provide sufficient information for sorting the processors. Active power consumption running a uniform test benchmark, in other words; *performance per watt* (i.e., IPS per Watt) can be used to sort the processors in terms of their power efficiency.

Temperature Variation: After determining the metric to sort the processors for different node assembly strategies, we need to make sure a custom node assembly technique would not have significant temperature and performance implications.

First, we analyze temperature variation. We measure the temperature and power correlation in order to determine if different power-based node assembly techniques can potentially create temperature imbalance within the data center. Figure 9 shows the average chip temperatures ranging from 43 to 70°C and their correlation with the idle power. The trend indicates that the leaky chips tend to have higher temperatures, although the R-squared correlation co-efficiency is not very high (0.57). The reason for the low correlation is likely caused by other factors related to cooling such as; physical placement and air circulation within the cluster. On the other hand, the active power and temperature correlation is much lower as shown in Figure 9, the R-squared value is only 0.34. With this low correlation between active power and temperature, we can assume that sorting the components based on their active power would not create a significant temperature imbalance within the data center.

Performance Variation: Next, we analyze performance variation. Figure 11 shows that there is only 3% variation in IPS levels among the chips running the MaxPow stress test benchmark at nominal frequency. Moreover, the 3% performance variation does not show any correlation with the chip power consumption. Since power variation shown here have no visible impact on performance, we can assume different node assembly techniques based on power would not cause performance degradation. Putting a lower power constraint on the processor or using dynamic overclocking features can create performance variability, however we do not consider such scenarios in this paper. We focus on power reduction benefit of the assembly techniques, despite the fact that there can be additional performance benefits of our techniques in systems with tight power constraints or with frequency boost features.

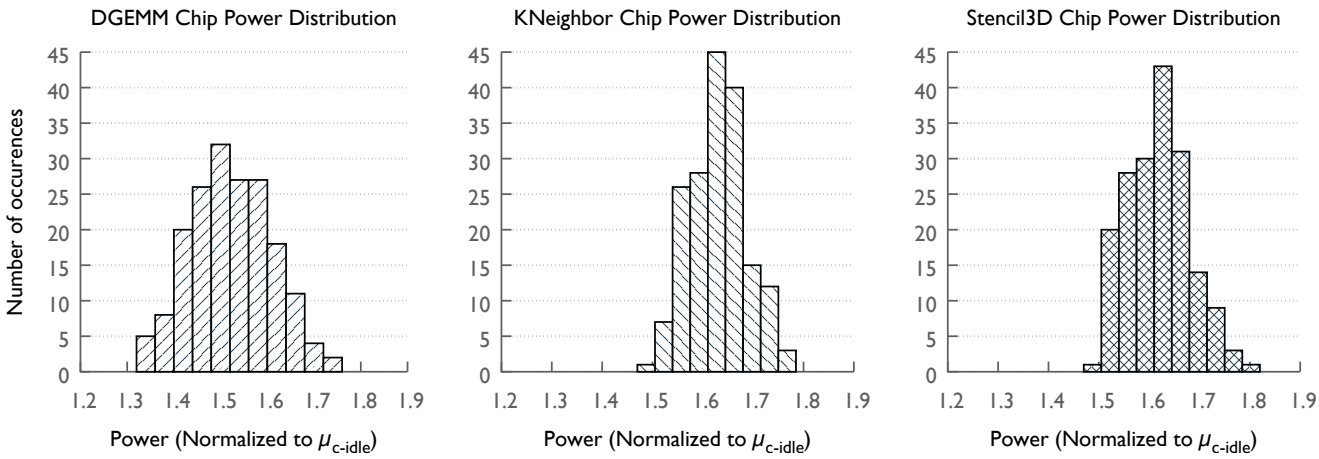

Figure 7: Average stable power distribution of chips running different benchmarks is shown. Although the benchmarks have different characteristics, all of them have balanced loads and hence showing similar distribution trends.

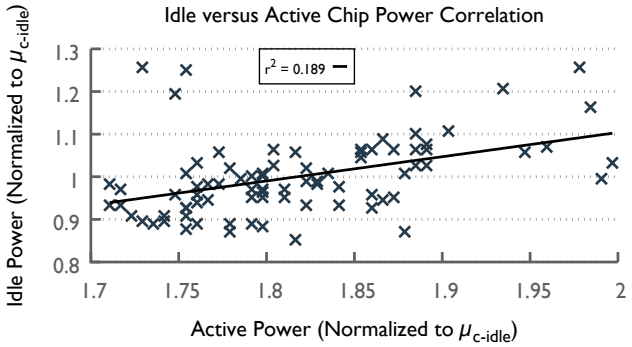

Figure 8: Idle versus active power of chips running MaxPow benchmark is shown. The chips that have high (or low) idle power do not necessarily have high (or low) active power.

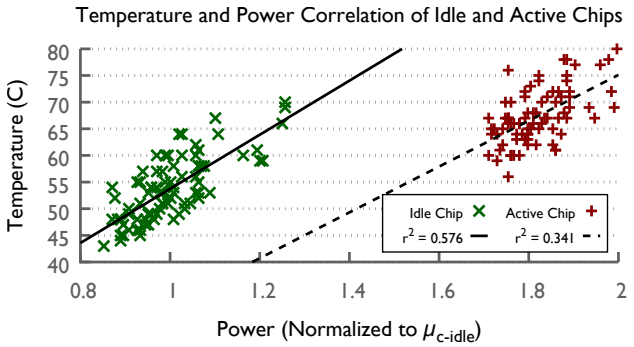

Figure 9: Temperature and power correlation is not statistically significant, especially for when the chips are active running MaxPow benchmark.

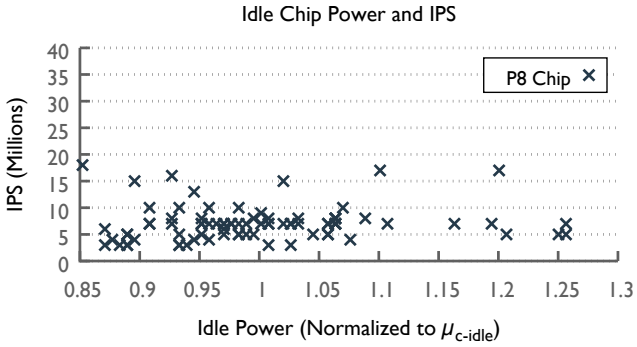

Figure 10: Idle instructions per second and power correlation is shown. The small variations in idle activity (IPS) is not correlated with the idle power of the chips.

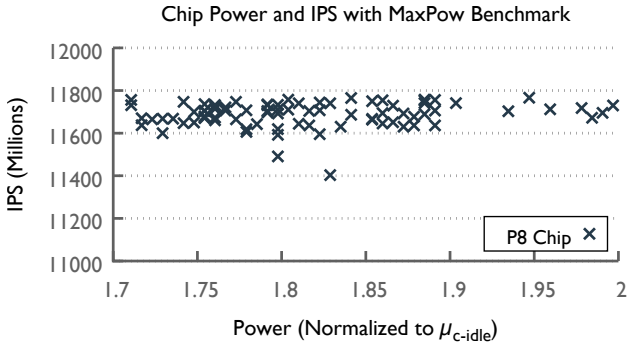

Figure 11: IPS and power correlation with MaxPow benchmark. There is only 3% variation in IPS levels, power variation is not affecting the performance.

4. VARIATION-AWARE NODE ASSEMBLY

After analyzing the power variation among components, we now describe the random node assembly and the three proposed techniques. Random assembly is currently used in production assembly lines. In using this method, the components of a physical node are assembled without considering characteristics of the individual components. Each component might have a different range of power variation, and hence the variation range (max to min power difference) can be higher in the total node power. The illustration of a datacenter assembled randomly is shown in Figure 2. The components in the figure are divided into five different categories (colors) based on their power efficiency scale. The number five is selected arbitrarily for illustration.

The three techniques we propose next are: Sorted Assembly, Balanced Power Assembly, and Application-Aware Assembly. Each of the techniques has its own advantages and use cases.

4.1 Sorted Assembly

This technique aims to create a data center where the components are sorted and assembled into the nodes based on their power efficiency. Then, nodes are placed into the racks in an ascending (or descending) sorted manner, as we show in Figure 3-(a). Each of the components within a node is sorted and assembled into bins according to power efficiency, and as a result each node contains different components with the same or similar efficiency ranking. This method enables physical nodes to be arranged based on their power efficiency (i.e., all power efficient components will be assembled into an efficient node, and all inefficient components will be assembled into an inefficient node).

Along with a job scheduler that is aware of the power efficiency of the nodes, this assembly technique can reduce the power and energy consumption compared with the random assembly if the data center is not fully utilized. For example, when the data center is at 90% load, the least efficient 10% of the nodes can be turned off first to reduce power consumption. To calculate the power reduction using Sorted Assembly at large scale, we first fit the power of the components in a Gaussian distribution as shown in Figure 12. To represent an average use case, we use DGEMM power distribution both in CPUs and GPUs, and memory consumption when using 30 GB of data per socket. Then, we generate 5,000 random nodes based on the distributions of each component. We extrapolate to 5,000 because it is about the same number of nodes in the Summit supercomputer. To simulate Sorted Assembly, we sort each of the components based on their power independently and combine them into nodes in the sorted order. Figure 13-a shows how much power reduction is achieved, when the data center is at different loads. The power reduction in terms of KW shows a dome-like curve due to the Gaussian distribution of the power of the node components. The maximum power reduction in terms of KW appears when the data center is at a 50% load with a 110 KW reduction where the entire power of active server components we simulate is around 6.5 MW. For reference, note that the total power consumption of the Summit supercomputer is about 8.8 MW [1]. The maximum reduction in terms of percentage power occurs when the data center is at the least loads. This is because as the fewer nodes are used, only the most efficient nodes will be used and inactive nodes are assumed to be turned off.

To understand the utilization levels of production supercomputers, we collected publicly available load data from three different supercomputing facilities: Argonne National Laboratory (ANL) [19], Texas Advanced Computing Center (TACC) [20], and National Energy Research Scientific Computing Center (NERSC) [21]. Figure 13-b shows the utilization levels over seven months of a period from November, 2017 to May, 2018 (with an exception of Stampede supercomputer which is decommissioned on April, 2018). The load data is collected hourly from the corresponding websites, and we show a weekly averages in the plot. Average utilization of these five systems are 75% during this period, assuming the maintenance days of the machines have 100% utilization. Compared to supercomputers, data centers too have large peaks in their loads. For example, Facebook can have approximately 40% diurnal swings in the load [22].

Sorted Assembly method can also help data center customers with limited power budget too. There might be maximum power limit for the system due to budget constraints or due the power draw infrastructure. These customers can be provided with an option to buy only the efficient nodes at higher prices. Other customers without such power budget limitations can buy the less efficient and less expensive nodes.

4.2 Balanced Power Assembly

This technique aims to equalize the total power of each node and reduce node-to-node power variation as illustrated in Figure 3-(b). Balanced power assembly helps making the system power and potentially performance more predictable. For example estimating the power when a data center needs to turn on or off some nodes depending on the utilization, predicting the required or reduced power would be much easier if node-to-node variation is lower. Similarly, expected performance per watt would also be less variable from node to node, which can help with application performance predictability and reproducibility under certain conditions. This could enable cloud providers estimate system cost and expected performance more accurately. Better power predictability can also help preventing power outages caused by sudden peaks in power at job start and end times.

Table 1 shows different component and node power variations when using balanced power assembly and random assembly. To simulate balanced power assembly, we first generate 5,000 random node components using the component distributions shown in Figure 12 in the same way as Section 4.1. This random generation creates nodes that have 27.8% CPU, 18.3% GPU, 21.5%

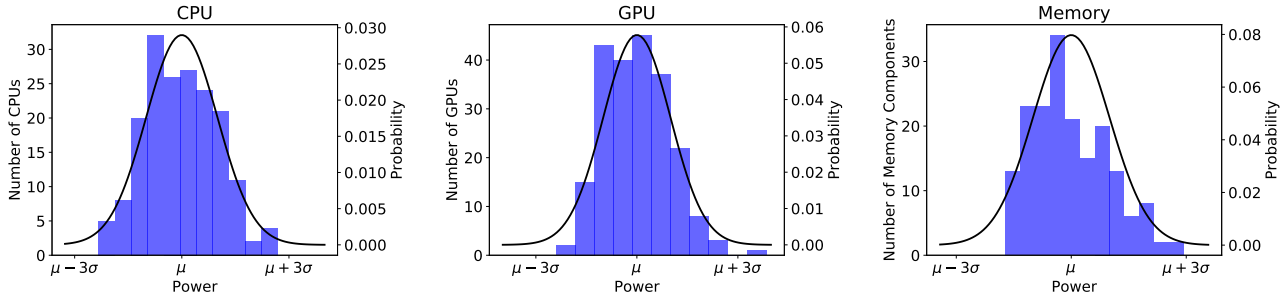

Figure 12: Distribution of the active power of different node components: CPU, GPU, Memory running DGEMM benchmark fit to the Gaussian distribution.

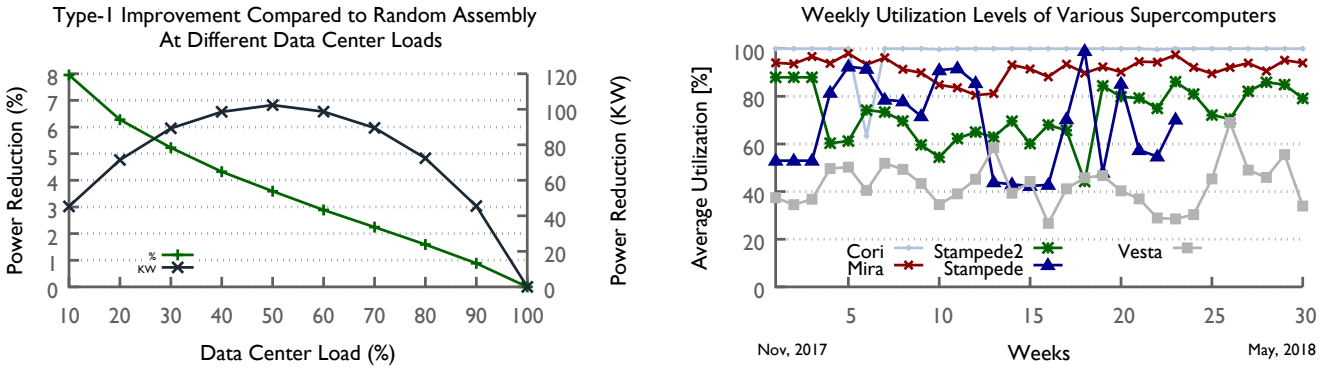

Figure 13: (a) Power reduction with sorted assembly compared to the random assembly at different data center loads with a size of 5,000 nodes. (b) Average weekly percentage utilization of different top supercomputers during a period of seven months. Data is collected hourly starting Nov 1, 2017 from ANL [19], TACC [20] and NERSC [21] public websites. Maintenance days are assumed to have 100% utilization level.

memory and 14.4% total node-to-node power variation. Next, for each component type (i.e., CPU, GPU, memory) we sort the list in ascending order according to their power consumption. From the sorted list, the component that draws the highest and the lowest power are assembled in a node to create balanced power. Assembled components are removed this way until there is no element left in the list. As a result, node-to-node variation reduces from 14.4% to only 0.4%.

While balanced power assembly balances inter-node power, it does not remove intra-node component power variation, i.e., power variation among the components within the same node. On one hand, there would not be any performance impact from this if the system is running under fixed frequency setting. On the other hand, if the system is running under dynamic overclocking or under power constraints, performance of the applications would depend on their characteristics. Inter-node imbalance would not impact overall performance and node-to-node performance would be less variable for applications that are not barrier synchronized as commonly seen in cloud data centers. However for synchronized HPC applications, inter-node variation can degrade the performance. Software based inter-node power shifting strategies can be used to mitigate the within node imbalance for HPC workloads. Overall, this assembly method would be more suitable for cloud data centers.

4.3 Application Aware Assembly

This method customizes assembly according to application needs. For example, if the node is running GPU-intensive workloads, then it should have power efficient GPUs, but it can have inefficient CPUs. By using the most power efficient components for the most intensive workloads, overall power consumption can be reduced. With this method, customers who know that they are going to run CPU-intensive applications can only purchase nodes with efficient CPUs. Application characteristics of the customers are generally well-known. For example, for Summit and Sierra, CORAL benchmarks suite provides a good representation [23]. A data center might have different types of workloads and hence can contain multiple varieties of nodes assembled using this method as illustrated in Figure 3-(c). In that case, the job scheduler of the data center can be aware of application characteristics and node characteristics to place the applications on the appropriate nodes to obtain the best power efficiency and potentially performance too (i.e., allocate CPU-intensive applications to CPU effi-

Table 1: Node to node power variation with power balanced assembly compared to random assembly. Power data is normalized to Min. Node Power in Random Assembly.

	CPU Variation	GPU Variation	Memory Variation	Min Node Power	Max Node Power	Node Variation
Random Assembly	27.8 %	18.3 %	21.5 %	1 (1097W)	1.15 (1267W)	14.4 %
Balanced Power Assembly	1.4 %	0.7 %	1.4 %	1.06 (1173W)	1.07 (1178W)	0.4 %

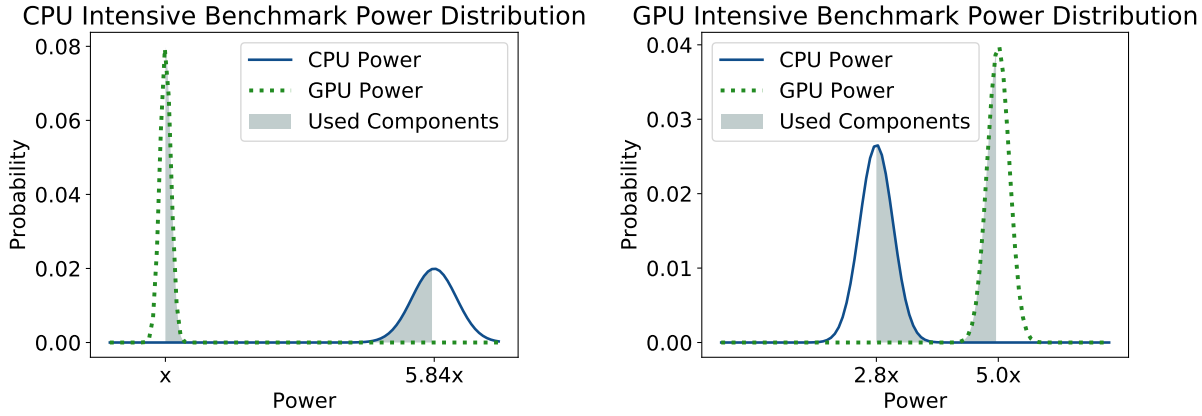

Figure 14: With application-aware assembly, CPU-intensive benchmarks run on the most power efficient *half* the CPUs, and inefficient part of GPUs. Whereas, GPU-intensive applications run on the most power efficient GPUs and inefficient *half* of the CPUs.

Table 2: Power reduction with application-aware assembly compared to random assembly. Power is normalized according to random assembly in each column.

	CPU Power	GPU Power	Node Power
Random Assembly	1	1	1
App. Aware Assembly	0.97	0.98	0.98
Power Reduction	3%	2%	2%

cient nodes).

In order to evaluate the power reduction with application-aware assembly, we consider a scenario where half of the data center applications are CPU intensive and the other half is GPU intensive. With application-aware assembly, CPU-intensive benchmarks run on the most power efficient *half* the CPUs, and inefficient part of GPUs. Whereas, GPU-intensive applications run on the most power efficient GPUs and inefficient *half* of the CPUs. This scenario is illustrated in Figure 14. The distributions provided here are estimated from our power measurements in Section 3.2 and we do not take into account the memory variation for simplicity. Resulting power reduction with application-aware assembly compared to random assembly is shown in Table 2. 130 KW of power reduction, which is equivalent to 2% of the node power, can be achieved in a data center comprised of 5,000 nodes.

This assembly method can also help improve the performance in certain cases. Past research shows how differences in chip power efficiency cause frequency variation among processors under dynamic overclocking in large-scale platforms [24]. This processor-to-processor variation is around 16% and manifests itself as performance variation in compute-intensive benchmarks. Therefore, a 16% performance improvement can be potentially achieved using this assembly method. Such variations can be higher under power constrained environments as well [25]. The frequency is fixed in our cluster, and there are no power constraints. Therefore, we did not obverse any frequency differences among the processors using any of the benchmarks. As we show earlier in Section 3.2, with the MaxPow stress test benchmark, we observe only about 3% variation in IPS levels.

5. EVALUATION

In this section, we compare and evaluate our node assembly methods in various aspects including use cases, dollar cost and power reduction under different variability scenarios.

First, we provide a simple example of how we make cost analysis of our assembly methods. We use the same architecture simulated in previous sections (i.e., with 5000 nodes). In our calculations, we take the electricity price as the average electricity price of all states and sectors in the U.S. according to data reported by the U.S. Energy Information Administration [26]. We assume system downtime is 15 days a year for maintenance, outage, or other reasons. Table 3 shows how we calculate the yearly power cost reduction. A 3.5% reduction in total system power means around 90 thousand dollars of reduction in energy bills every year. We believe the additional assembly cost will not be signifi-

cant since sorting and binning are already parts of the production line of the processors.

Table 3: Example cost reduction analysis.

AC = Additional Assembly Cost
ER = Energy Reduction
PR = Power Reduction
EP = Electricity Price = 10.48 cents per kWh [26]
T = System Up Time = 350 days = 8400 hours
Average System Utilization = 50% (or 3.5% total reduction)

$$CostReduction = EP \times ER - AC$$

$$0 < EP \times T \times PR - AC$$

$$AC < 10.48\ (cents\ per\ kWh) \times 8400\ hours \times 100\ KW$$

$$AC < \$90,083\ per\ year$$

What if node count increases? As the race to exascale continues, it is likely that the future systems are going to have larger node counts than the current ones. Here we evaluate the cost reduction with different node counts. Figure 15 shows how much cost reduction can be achieved at from 5,000 to 50,000 node counts with sorted and application-aware assembly types. Balanced power assembly does not provide cost reduction, therefore it is not shown in the plot. Application-aware assembly method shows higher cost reduction at higher data center loads, however overall sorted assembly provides higher cost reduction. It can provide up to eight million dollar cost reduction at 50,000 node scale.

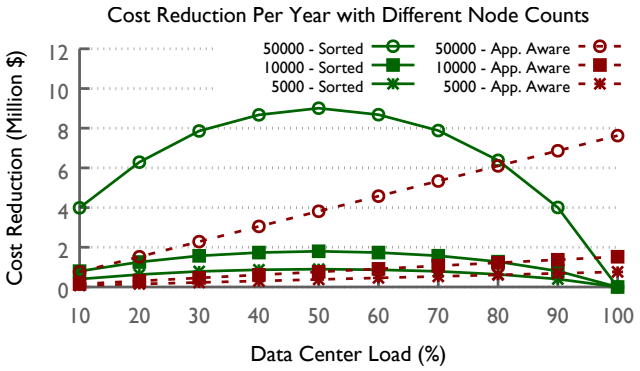

Figure 15: Dollar savings increase as the data center size increases.

What if variation increases? To evaluate increased process variation, we assume the mean power consumption of the node components remains the same while the standard deviation increases. We consider two different standard variation levels: 1.5x and 2x more than their current values. We scale the values for each node component independently. Figure 16 shows the power reduction on these scenarios for sorted assembly and application-aware assembly. Sorted assembly power reduction increases proportionally with the increase in the standard deviation. On the other hand, increased variation does not significantly increase application-aware assembly power reduction. While application-aware assembly power reduction is more when the data center load is more than 90%, sorted assembly power reduction is higher overall. Since there is no power reduction associated with balanced power assembly, it is not shown in the plot. However, regardless of the standard deviation level, balanced node assembly can reduce the node to node variation to less than 1%.

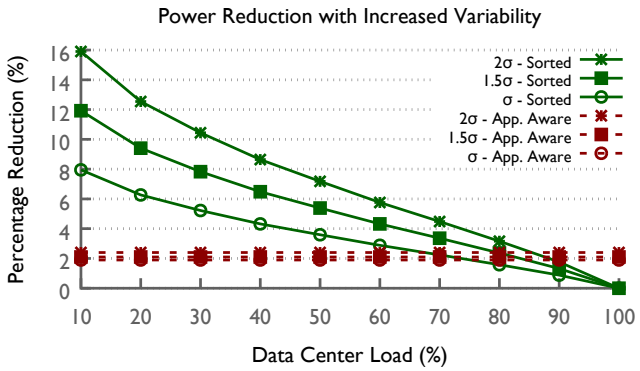

Figure 16: Power reduction increases as variability increases. σ represents measured standard deviation in the current architecture. 1.5σ, 2σ represents the scenarios when the deviation increases 1.5x and 2x times respectively.

Putting it all together: Table 4 shows a brief summary of use cases, energy savings and required job scheduler support for the assembly types. Sorted assembly is suitable for systems with variable utilization rates with minimal job scheduler support and it provides significant energy savings. Balanced power assembly is suitable for all cloud systems with no job scheduler support but it does not provide any energy savings. Application-aware assembly is suitable for systems with a mixed set of application characteristics but it requires major job scheduler support and energy savings may not be too significant. None of the methods cause performance degradation running at nominal frequency. Among the three methods, Sorted Assembly is the most favorable one because of the high energy savings and wide use case.

Discussion: Feasibility and practicality of the assembly techniques need further discussion. An important aspect regarding the cost and feasibility is to determine if re-sorting of the components is necessary and if it is, how often re-sorting needs to be performed. Resorting may be necessary because of the decay of the processors over time. Utilization levels of the processors and hence decay rates may not be the same. Determining this would require a long-term study of the decay in large scale data-centers. To prevent imbalances in the utilization, decay based OS schedulers have been proposed [27]. Similar approaches can be adapted in data center job schedulers as well if imbalanced decay is a significant concern.

Another important aspect is regarding cost and feasibility is cooling. Server power and energy consump-

Table 4: Comparison of the node assembly types.

	Use Cases	Job Scheduler Support	Energy Savings	Performance Degradation
Sorted Assembly	Systems with variable utilization rates	Minor	✓	×
Balanced Power Assembly	Cloud systems	None	×	×
App. Aware Assembly	Systems with mixed app. characteristics	Major	✓	×

tion is not the only cost of a data center. According to the 2016 United States data center energy usage report by Lawrence Berkeley National Laboratory, the average electricity spent by servers is about 30-40% of the total data center electricity costs [28]. Therefore, the cost savings with our assembly method would affect 30-40% of the overall data center electricity and cooling costs can be significant.

Cooling systems are designed for the worst case scenario, i.e. considering the highest power consuming components, therefore the cooling system does not require any changes to support our node assembly methods. With Sorted and Application-Aware Methods, there could be some cooling power reduction if the system has support for variable cooling water flow rate (which is not common in current water-cooled systems). Since the most power efficient components tend to have lower temperatures, despite the fact that power and temperature is not highly correlated (see Figure 9), water flow rate could be reduced with Sorted Assembly Method as the utilization rate goes down. Quantifying the cooling benefits (which may not be significant) requires a detailed study which uses thermodynamically accurate data center temperature models.

6. RELATED WORK

Variability in performance and power of CMOS architectures is a well-known problem [4, 5, 29, 30]. Similar issues have been identified and discussed in the HPC node context before [31], but they are not addressed in the way our solution does. Different ideas to improve the performance under variability have been proposed in the context of multicore processors [32]. Donald et al. analyzed core-to-core power variations in a CMP due to Within Die (WID) variation and proposed to turn off cores when cores consume excessive leakage power to improve the chip-wide performance or power metric [33]. Teodorescu et al. proposed heuristics to schedule applications for performance or power efficiency in the context of many-core processors with variation [34]. Rangan et al. proposed using mean frequency of the cores as a metric along with a Throughput-Driven Fairness (TDF) scheduling policy instead of dictating the chip frequency based on the slowest core in the chip to provide homogeneous performance [35].

There are various other software-level approaches proposed to mitigate power variations and their side effects. Runtime-based dynamic load balancing strategy has been proposed to mitigate the frequency variations under dynamic overclocking (i.e., Turbo Boost) [24]. In power-constrained systems, power variations manifest themselves as performance variations more strongly. Variation-aware job scheduling approaches at the node-level [25, 36, 37] and within-node [38] have been proposed to mitigate variations. However, these approaches can increase the complexity of the scheduling algorithms and require consideration of the trade-off between performance and power. Hence, these approaches could be infeasible at large scale. Moreover, power-aware job scheduling techniques could conflict with other job requirements such as topology requirements for better network performance. A hardware-level approach can remove the need for such software methods. To the best of our knowledge, node-level power variations and assembly of the "fat heterogeneous" node components have not been studied in the literature before.

The alternative approach to "fat heterogeneous" nodes with GPUs is "fat homogeneous" nodes with large amounts of small cores. This approach increases the chance of having large intra-chip, core-to-core variations that cannot be mitigated via assembly methods proposed in this work since there is only one big chip. Software-based dynamic scheduling approaches may become more important for such architectures to cope with variability.

7. CONCLUSION AND FUTURE WORK

In this paper, we first measure power variation of components within a physical compute node of a modern HPC system. Then, we propose three new node assembly techniques to mitigate the adverse effects of component power variation in nodes. Each of these techniques has its own merits and use cases. The first one (Sorted Assembly) produces nodes with components of similar power efficiency. The second one (Balanced Power Assembly) tries to minimize node-to-node power variations. Finally, the third one (Application-Aware Assembly) assembles nodes based on application characteristics. None of these techniques imposes a performance degradation to the applications while reducing the power consumption and variation. In terms of cost savings and practicality, we conclude that Sorted Assembly is the most advantageous one among the three.

The variation in power consumption and performance might grow further with the scaling of the integrated circuits, hence the systems should adapt and be designed to harness the variability. We show our assembly methods enable significant power reduction in such scenarios without impeding the performance. As the cost versus performance is an essential factor for both supercomputer customers and data center operators, we hope to continue in-depth price and profitability analysis based on the work presented in this paper.

Acknowledgements

We thank Jon Tetzloff for his help in extracting the parametric hardware data.

8. REFERENCES

[1] "TOP500 List - November 2018." https://www.top500.org/lists/2018/11/, 2018.

[2] "Summit Supercomputer at ORNL." https://www.olcf.ornl.gov/summit/, 2017.

[3] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch sgd: training imagenet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.

[4] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variations and impact on circuits and microarchitecture," in *Proceedings of the 40th annual Design Automation Conference*, pp. 338–342, ACM, 2003.

[5] K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer, "High-performance cmos variability in the 65-nm regime and beyond," *IBM journal of research and development*, vol. 50, no. 4.5, pp. 433–449, 2006.

[6] C. Vieu, F. Carcenac, A. Pepin, Y. Chen, M. Mejias, A. Lebib, L. Manin-Ferlazzo, L. Couraud, and H. Launois, "Electron beam lithography: resolution limits and applications," *Applied surface science*, vol. 164, no. 1-4, pp. 111–117, 2000.

[7] J. W. Tschanz, J. T. Kao, S. G. Narendra, R. Nair, D. A. Antoniadis, A. P. Chandrakasan, and V. De, "Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 11, pp. 1396–1402, 2002.

[8] D. Boning and S. Nassif, "Models of process variations in device and interconnect," *Design of high performance microprocessor circuits*, p. 6, 2000.

[9] R. R. Rao, D. Blaauw, D. Sylvester, and A. Devgan, "Modeling and analysis of parametric yield under power and performance constraints," *IEEE Design & Test of Computers*, vol. 22, no. 4, pp. 376–385, 2005.

[10] P. J. Osler, J. M. Cohn, and D. Chinnery, "Design closure," in *Electronic Design Automation for IC Implementation, Circuit Design, and Process Technology: Circuit Design, and Process Technology*, ch. 13, CRC Press, 2016.

[11] B. Jovanović and M. Jevtić, "Static and dynamic power consumption of arithmetic circuits in modern technologies," in *ETRAN, National Conference*, pp. EL3–6, 2011.

[12] A. P. Chandrakasan and R. W. Brodersen, "Minimizing power consumption in digital cmos circuits," *Proceedings of the IEEE*, vol. 83, no. 4, pp. 498–523, 1995.

[13] V. Zolotov, C. Visweswariah, and J. Xiong, "Voltage binning under process variation," in *Proceedings of the 2009 International Conference on Computer-Aided Design*, pp. 425–432, ACM, 2009.

[14] A. Datta, S. Bhunia, J. H. Choi, S. Mukhopadhyay, and K. Roy, "Profit aware circuit design under process variations considering speed binning," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 7, pp. 806–815, 2008.

[15] M. Bhushan and M. B. Ketchen, "Ring oscillators," in *Microelectronic test structures for CMOS technology*, ch. 6, Springer Science & Business Media, 2011.

[16] "AMESTER: Automated Measurement of Systems for Temperature and Energy Reporting." https://github.com/open-power/amester, 2017.

[17] R. Bertran, A. Buyuktosunoglu, M. S. Gupta, M. Gonzalez, and P. Bose, "Systematic energy characterization of cmp/smt processor systems via automated micro-benchmarks," in *Microarchitecture (MICRO), 2012 45th Annual IEEE/ACM International Symposium on*, pp. 199–211, IEEE, 2012.

[18] A. Vassighi and M. Sachdev, "Power, junction temperature, and reliability," *Thermal and Power Management of Integrated Circuits*, pp. 13–49, 2006.

[19] "Argonne Leadership Computing Facility Machine Status." https://www.alcf.anl.gov/machine-status, 2017.

[20] "TACC User Portal Compute/Visualization Resources." https://portal.tacc.utexas.edu/system-monitor, 2017.

[21] "NERSC Live Status - Compute Systems." http://www.nersc.gov/users/live-status, 2017.

[22] K. Hazelwood, S. Bird, D. Brooks, S. Chintala, U. Diril, D. Dzhulgakov, M. Fawzy, B. Jia, Y. Jia, A. Kalro, J. Law, K. Lee, J. Lu, P. Noordhuis, M. Smelyanskiy, L. Xiong, and X. Wang, "Applied machine learning at facebook: A datacenter infrastructure perspective," in *High Performance Computer Architecture (HPCA), 2018 IEEE 24th International Symposium on*, IEEE, 2018.

[23] "CORAL Collaboration Benchmark Codes." https://asc.llnl.gov/CORAL-benchmarks/, 2018.

[24] B. Acun, P. Miller, and L. V. Kale, "Variation among processors under turbo boost in hpc systems," in *Proceedings of the 2016 International Conference on Supercomputing*, ICS '16, (New York, NY, USA), pp. 6:1–6:12, ACM, 2016.

[25] Y. Inadomi, T. Patki, K. Inoue, M. Aoyagi, B. Rountree, M. Schulz, D. Lowenthal, Y. Wada, K. Fukazawa, M. Ueda, *et al.*, "Analyzing and mitigating the impact of manufacturing variability in power-constrained supercomputing," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, p. 78, ACM, 2015.

[26] U. E. I. Administration, "Average Price of Electricity to Ultimate Customers by End-Use Sector." https://www.eia.gov/electricity/monthly/epm_table_grapher.php?t=epmt_5_6_a, 2017.

[27] J. L. Hellerstein, "Achieving service rate objectives with decay usage scheduling," *IEEE Transactions on Software Engineering*, vol. 19, no. 8, pp. 813–825, 1993.

[28] A. Shehabi, S. Smith, N. Horner, I. Azevedo, R. Brown, J. Koomey, E. Masanet, D. Sartor, M. Herrlin, and W. Lintner, "United states data center energy usage report," *Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775*, vol. 4, 2016.

[29] S. Dighe, S. R. Vangal, P. Aseron, S. Kumar, T. Jacob, K. A. Bowman, J. Howard, J. Tschanz, V. Erraguntla, N. Borkar, *et al.*, "Within-die variation-aware dynamic-voltage-frequency-scaling with optimal core allocation and thread hopping for the 80-core teraflops processor," *Solid-State Circuits, IEEE Journal of*, vol. 46, pp. 184–193, Jan 2011.

[30] D. Marculescu and E. Talpes, "Variability and energy awareness: a microarchitecture-level perspective," in *Design Automation Conference, 2005. Proceedings. 42nd*, pp. 11–16, IEEE, 2005.

[31] A. Kannan, N. E. Jerger, and G. H. Loh, "Enabling interposer-based disintegration of multi-core processors," in *Microarchitecture (MICRO), 2015 48th Annual IEEE/ACM International Symposium on*, pp. 546–558, IEEE, 2015.

[32] T. Vijayaraghavany, Y. Eckert, G. H. Loh, M. J. Schulte, M. Ignatowski, B. M. Beckmann, W. C. Brantley, J. L. Greathouse, W. Huang, A. Karunanithi, *et al.*, "Design and analysis of an apu for exascale computing," in *High Performance Computer Architecture (HPCA), 2017 IEEE International Symposium on*, pp. 85–96, IEEE, 2017.

[33] J. Donald and M. Martonosi, "Power efficiency for variation-tolerant multicore processors," in *Proceedings of the 2006 international symposium on Low power electronics and design*, pp. 304–309, ACM, 2006.

[34] R. Teodorescu and J. Torrellas, "Variation-aware application scheduling and power management for chip

multiprocessors," in *Computer Architecture, 2008. ISCA'08. 35th International Symposium on*, pp. 363–374, IEEE, 2008.

[35] K. K. Rangan, M. D. Powell, G.-Y. Wei, and D. Brooks, "Achieving uniform performance and maximizing throughput in the presence of heterogeneity," in *High Performance Computer Architecture (HPCA), 2011 IEEE 17th International Symposium on*, pp. 3–14, IEEE, 2011.

[36] T. Patki, D. K. Lowenthal, B. Rountree, M. Schulz, and B. R. de Supinski, "Exploring Hardware Overprovisioning in Power-constrained, High Performance Computing," in *Proceedings of the 27th international ACM conference on International conference on supercomputing*, pp. 173–182, ACM, 2013.

[37] N. Gholkar, F. Mueller, and B. Rountree, "Power tuning hpc jobs on power-constrained systems," in *Proceedings of the 2016 International Conference on Parallel Architectures and Compilation*, pp. 179–191, ACM, 2016.

[38] E. Totoni, A. Langer, J. Torrellas, and L. Kale, "Scheduling for HPC Systems with Process Variation Heterogeneity," *PPL Technical Report*, 2014. http://charm.cs.uiuc.edu/media/14-35.