Energy Efficiency of Hierarchical Server Load Distribution Strategies

Jóakim v. Kistowski*, John Beckett†, Klaus-Dieter Lange‡, Hansfried Block§, Jeremy A. Arnold¶ and Samuel Kounev*
*University of Würzburg, †Dell Inc., ‡Hewlett Packard Enterprise, §Fujitsu Technology Solutions GmbH, ¶IBM Corporation
{joakim.kistowski, samuel.kounev}@uni-wuerzburg.de, john_beckett@dell.com,
klaus.lange@hpe.com, hansfried.block@ts.fujitsu.com, arnoldje@us.ibm.com

***Abstract*—Energy efficiency of servers has become a significant issue over the last years. Load distribution plays a crucial role in the improvement of energy efficiency as (un-)balancing strategies can be leveraged to distribute load over one or multiple systems in a way in which resources are utilized at high performance, yet low overall power consumption. This can be achieved on multiple levels, from load distribution on single CPU cores to machine level load balancing on distributed systems. With modern day server architectures providing load balancing opportunities at several layers, answering the question of optimal load distribution has become non-trivial. Work has to be distributed hierarchically in a fashion that enables maximum energy efficiency at each level. Current approaches balance load based on generalized assumptions about the energy efficiency of servers. These assumptions are based either on very machine-specific or highly generalized observations that may or may not hold true over a variety of systems and configurations.**

In this paper, we use a modified version of the SPEC SERT suite to measure the energy efficiency of a variety of hierarchical load distribution strategies on single and multi-node systems. We introduce a new strategy and evaluate energy efficiency for homogeneous and heterogeneous workloads over different hardware configurations. Our results show that the selection of a load distribution strategy depends heavily on workload, system utilization, as well as hardware. Used in conjunction with existing strategies, our new load distribution strategy can reduce a single system's power consumption by up to 10.7%.

I. Introduction

The energy efficiency of servers has become a significant issue as data center energy consumption has risen dramatically over the past decade. In 2010, the U.S. Environmental Protection Agency (U.S. EPA) estimated 3% of all electricity consumed in the U.S. to be used in running data centers [12]. According to a New York Times study from 2012, data centers worldwide consume about 30 billion watts per hour. This is equivalent to the approximate energy output of 30 nuclear power plants [1]. This leads to an increasing pressure on hardware vendors to design systems with a high energy efficiency. Equally, software developers are tasked with the design and development of energy efficient applications.

Servers are rarely fully idle; but instead, they often serve requests that arrive at low frequencies leading to a typical load at a low-resource utilization level [2]. As a result, software can be designed to distribute its load on target systems in a manner that reduces power consumption. Typically, work is consolidated on as few systems as possible, allowing unused systems to enter power saving states. Designing such workload consolidation mechanisms and policies is challenging, as load intensity varies dynamically [26]. Additionally, power consumption, performance, and energy efficiency characteristics change depending on workload, utilization, and machine hardware and software configuration [25]. Finally, newer server processors, such as Intel's Haswell generation processors, feature advanced core-level power management mechanisms, which may warrant the application of workload consolidation on a processor core level.

Current load distribution approaches balance load based on generalized assumptions about the energy efficiency of servers. Most existing power consolidation approaches, such as [16] and [4], consolidate as much work as possible on each machine until a preconfigured threshold of performance degradation is met. These approaches assume optimal energy efficiency at full machine level utilization. Similar assumptions are made for many existing power management solutions. These approaches also do not attempt to fully maximize energy efficiency as they only minimize power consumption within specified performance constraints. They do not maximize the tradeoff between performance and power.

Hierarchical power management solutions explore the possibilities of triggering power saving states and mechanisms of hardware components. [20] and [24], e.g., take the effects of CPU dynamic voltage and frequency scaling (DVFS) into account. They do not, however, make use of lower level load distribution, such as core-wise load distribution.

In this paper, we use a modified version of the SPEC Server Efficiency Rating Tool (SERT) to measure the energy efficiency of a variety of load distribution strategies on single and multi-node systems. To do so, we modify SERT's target load intensity (transaction arrival rate) and target workload on a client by client basis for each load level, with the clients being pinned to specific CPU cores. Using this approach we can emulate any load distribution policy for SERT's transactional workloads, allowing the evaluation of energy efficiency for homogeneous and heterogeneous workloads over different hardware and software configurations.

The goal of this paper is to gain insight in how different load balancing and unbalancing strategies affect overall power consumption and energy efficiency. The core contributions of the paper are:

1) We investigate power consumption and energy efficiency of load balancing and consolidation policies at multiple

utilization levels on a range of hardware configurations, including different architectures.
2) We explore the power consumption and energy efficiency of hierarchical load distribution on four levels: logical CPUs, physical CPUs, separate CPU sockets, and machine level load distribution for distributed multi-node systems.
3) We demonstrate that the effectiveness of load distribution policies depends on multiple factors, including workload type and load level for both homogeneous and heterogeneous workloads.
4) We combine existing strategies to create a new load distribution strategy using partial consolidation to target maximum energy efficiency across as many execution units as possible.

We evaluate the load distribution strategies on a range of systems with increasing complexity: a one-socket system, two dual-socket systems of different architectures and two dual-socket systems running the workload in a multi-node configuration. For each of these systems, we test a variety of combinations applying different load distribution strategies on the different levels of the execution hierarchy (nodes, sockets, physical CPU cores, logical CPU cores).

We show that the selection of a single most energy-efficient strategy is only possible on smaller or older systems. For other systems the most energy efficient load distribution strategy depends on workload type and load level. For some loads, the most efficient strategy is not always the commonly assumed one, e.g., full load consolidation on a multi-node level can have a smaller energy efficiency than other load distributions. We also show that our new strategy can save up to 10.7% of power consumption on a single server node.

The remainder of this paper is structured as follows: Section II describes the related work to this study. Section III introduces the SERT rating tool, its architecture, workloads, and measurement methodology. We then describe our modifications to SERT in Section IV, explaining how client-wise distribution of transactional workloads is achieved. Measurement results are then presented in Sections V and VI. Finally, we conclude the paper in Section VII.

II. Related Work

We group related work for this paper into two major non-exclusive categories: Experimental studies of server power management and (hierarchical) load distribution algorithms:

Experimental studies of server power management: Several studies of server power management, including the effects of workload consolidation have been conducted in the past: In [15], the authors present a study that explores the integration of power management policies, including workload consolidation in virtualized environments using micro-benchmarks. In [22], energy efficiency for workload consolidation is explored on the server level for several workload mixes. In [3], the power consumption of three different task types on a single physical server is analyzed. Examined impact factors include the number of running VMs, task configuration parameters, and four different target load levels. [14] examines multiple balancing and unbalancing strategies for virtual machines in cloud environments. The authors of [17] analyze the effect of CPU pinning configurations for two virtualized processes on performance degradation through resource contention, they also measure power consumption during their experiments.

In contrast to this existing work, our study focuses on hierarchical compositions of multiple load distribution strategies at a high number of load levels. We analyze the interdependencies of processor architecture, workload type, load level, and distribution strategy with the goal of achieving optimum energy efficiency for both homogeneous and heterogeneous workloads.

Load distribution algorithms: There are numerous different approaches to workload consolidation, which differ with respect to various aspects. Some account for temperature, some for device wear-and-tear, some account for resource provisioning and deprovisioning times, and so on. [16] features one of the first approaches to node-wise load distribution. It showed the promise of workload consolidation, although many of the mechanisms were still executed manually. In [4], a workload distribution mechanism that accounts for power consumption, temperature, and device wear-and-tear is introduced. The authors of [19] compare multiple resource allocation strategies, including strategies that use a UML-based meta-modeling approach for power prediction. The approach in [9] allocates work using lookahead control. It has the goal of minimizing power consumption, while keeping quality-of-service (QoS) above a set threshold. [5], [20], and [24] feature hierarchical power management approaches. Load consolidation is, however, only managed on a per-machine or core-cluster level. CPU-level task consolidation is evaluated separately in [6] and [8]. Both works focus on power density with the purpose of minimizing heat generation within microprocessors.

The above mentioned load distribution mechanisms feature advanced decision making engines that decide when to change or reconfigure the current load distribution. The target load distribution is based on basic distribution strategies, primarily load consolidation. We evaluate these basic strategies and introduce a new one with the goal of offering additional options to power management decision engines.

III. SERT

SERT has been developed by the SPEC OSG Power Committee as a tool for the analysis and evaluation of the energy efficiency of server systems. Its design and development goals and process have been introduced in [12]. In contrast to energy-efficiency benchmarks, such as JouleSort [21], SPECpower_ssj2008 [10], and the TPC-Energy benchmarks [18], SERT is not intended to be used as a benchmark for a single system energy-efficiency score. It does not aim to simulate real world end user workloads, but instead provides a set of focused synthetic micro-workloads called worklets that exercise selected aspects of the Server (or System) Under Test (SUT). Specifically, the worklets have been developed to exercise the processor, memory, and storage I/O subsystems, and may be combined into various configurations running serially or in parallel to provide ”system” tests as part of a larger workload. We make use of this feature by combining multiple worklets into heterogeneous workloads to be distributed across the SUT.

According to [2], servers nowadays spend most of their time in a CPU utilization range between 10% and 50%. As a result, SERT and its worklets are designed for the measurement of system energy efficiency at multiple load levels. This sets it further apart from conventional performance benchmarks, such as SPEC CPU [7], which targets maximum load and performance. To achieve workload execution at different load levels, SERT calibrates the load by determining the maximum transaction rate for the given worklet on the SUT. The maximum transaction rate is measured by running as many of the worklet's transactions as possible concurrently on each client. This calibrated rate is then set as the 100% load level for all consecutive runs. For each target load level (e.g., 100%, 75%, 50%, 25%), SERT calculates the target transaction rate and derives the corresponding mean time from the start of one transaction to the start of the next transaction. During the measurement interval, these delays are randomized using an exponential distribution that statistically converges to the desired transaction rate. As a result, lower target loads consist of short bursts of activity separated by periods of inactivity.

A detailed description of the SERT memory worklets and their applicability can be found in [11]. A detailed description of the storage I/O worklets and their properties can be found in [13].

A. Tool Architecture

SERT's measurements are controlled by a *controller* system. This system runs the Chauffeur harness, the reporter, the optional graphical user interface, and instances of the SPEC PTDaemon.

Chauffeur is the framework on which SERT is built. It handles both the coordination with the SUT director triggering the execution of worklets, as well as the communication with other controller-internal components, such as the PTDaemon and the reporter. The *Chauffeur director* communicates with the *SERT host* running on the SUT. The host in turn spawns separate *clients* for each logical CPU (logical processor, also called hyperthreading or SMT unit). These clients are bound to their logical and physical CPU using an *affinity provider*. The transactional workload is executed sequentially on the clients. Parallelism is achieved by running multiple clients concurrently.

The *SPEC PTDaemon* is a tool that allows network-based communication with a host connected to power and temperature measurement devices. PTDaemon supports a range of SPEC-accepted devices, all featuring a maximum measurement uncertainty of 1% or better. Additional components in SERT include a graphical user interface (GUI) for easy test-run execution and a reporter, which generates the final report including the measurement results and energy-efficiency scores.

SERT requires at least one power analyzer and one temperature sensor. The power analyzer measures the power consumption of the entire SUT, while the temperature sensor verifies the validity of measurements by assuring that all experiments are conducted under similar environmental conditions.

B. Worklets

As this paper focuses on distribution strategies for server loads, we employ those worklets within SERT that offer opportunities to be distributed in multiple ways. As a result, we use worklets with at least some CPU-bound work, as this enables core-wise placement of the worklet's load. SERT offers seven CPU-worklets, which are designed to use the CPU as their bottleneck resource. However, some of the CPU worklets also use other resources besides the CPU. In addition to the CPU worklets, we employ the SSJ worklet, which is a hybrid worklet, using a resource mix of CPU, memory, and I/O resources. The performance metric employed for each of these worklets is throughput measured in transactions per second. Using SERT default settings, each CPU worklet is executed at a target load 25%, 50%, 75%, and 100%. SSJ is run at nine different target load levels (11%, 22% and so on). For a more detailed analysis, we have reconfigured our SERT runs to use 10% increments in load levels.

The following worklets are the core worklets employed in our measurements:

1) **CryptoAES**: Implements a transaction that encrypts and decrypts data using the AES or DES block cipher algorithms using the Java Cryptographic Extension (JCE) framework. It is a mostly CPU-bound workload. In [25] we discovered that it also depends on memory and can be bottlenecked by insufficient memory channels.
2) **LU**: Implements a transaction that computes the LU factorization of a dense matrix using partial pivoting. It exercises linear algebra kernels (BLAS) and dense matrix operations. LU is almost exclusively CPU bound and scales mostly with CPU frequency.
3) **SOR** (Jacobi Successive Over-Relaxation): Implements a transaction that exercises typical access patterns in finite difference applications, for example, solving Laplace's equation in 2D with Drichlet boundary conditions. The algorithm exercises basic "grid averaging" memory patterns. Like LU, this worklet is also mostly CPU bound. As such, we use it as our second worklet for CPU-heavy heterogeneous workloads.
4) **XMLValidate**: Implements a transaction that exercises Java's XML validation package *javax.xml.validation*. Using both SAX and DOM APIs, an XML file (.xml) is validated against an XML schemata file (.xsd). To randomize input data, an algorithm is applied that swaps the position of commented regions within the XML input data. XMLValidate uses both CPU and memory resources.
5) **SSJ**: The SSJ worklet is a hybrid worklet also used in the SPECpower_ssj benchmark. It executes a workload that represents a typical transaction based business application. As such, it makes use of multiple resource types, including processors and memory.

C. Measurement Methodology and SUT

We measure all results according to the SPEC Power and Performance Benchmark Methodology [23]. The devices are setup and configured as required by SERT (see Section III-A). The controller with the Chauffeur harness runs on an external machine, while the worklets are executed on the JVM within the SUT. An external power analyzer measures SUT AC wall power and a temperature sensor monitors the environmental temperature to ensure that it ranges between 22 and 23 °C for the duration of all test runs. This reduces power measurement

inaccuracies caused by varying leakage power, which can be a result of varying environmental temperatures.

SERT measures performance (throughput in s^{-1}) and power consumption (watt) during measurement intervals. Each interval contains a pre-measurement and a post-measurement period of 15 seconds each. In these periods the worklets are already being executed at the target load level, yet no measurements are recorded. Both periods serve the purpose of achieving a stable state for the power and performance measurements. The actually measured scenario is executed for a duration of 120 seconds. In this time all transactions are logged and power measurements are reported by PTDaemon at one second intervals. Total power consumption in the final SERT report is the average over the 120 reported values.

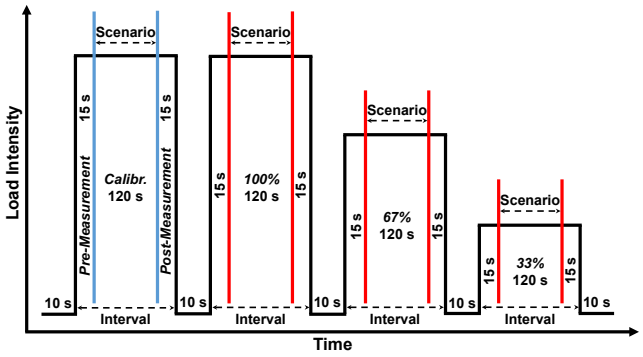

Fig. 1. SERT measurement intervals

To prevent intervals at different load levels or using different workloads from influencing one another, the system is put to sleep for 10 seconds in between each interval. Intervals are organized in sequences, the most important of which is the graduated measurement sequence. It executes the intervals at gradually diminishing target transaction rates. To determine these transaction rates, SERT uses the results of the calibration, which was run and measured using the same measurement interval structure. The measurement intervals and sequences for a worklet are illustrated in Fig. 1.

We evaluate the different load distribution strategies on four systems: A Fujitsu PRIMERGY TX1310 M1 machine is used as our SUT for the evaluation of physical core and SMT unit distribution strategies on a one socket system. The machine is equipped with an Intel Xeon E3-1230 v3 CPU (Haswell) featuring a base frequency of 3.3 GHz (up to 3.9 GHz with Turbo) and four DIMMs of DDR3 RAM. Power consumption of this machine is measured using a ZES Zimmer LMG95 power analyzer. Multi-socket load distribution strategies are tested on a Dell PowerEdge R720 and Dell PowerEdge R730 system. Both systems are equipped with 2 CPU sockets and eight DIMMs of RAM each. The Ivy Bridge based PowerEdge R720 system features two Intel Xeon E5-2667 v2 processors with a base frequency of 3.3 GHz (up to 4.0 GHz with Turbo), whereas the Haswell based PowerEdge R730 system is equipped with two Intel Xeon E4-2667 v3 CPUs with a base frequency of 3.2 GHz (up to 3.6 GHz with Turbo). The AC wall power for both of the dual-socket systems is measured using a Yokogawa WT210 power analyzer. Finally, we evaluate multi-node load distribution strategies by simulating multi-node results using the Dell systems and through separate measurements on a cluster featuring two HP ProLiant BL260a Gen9 Blades with two 18-core Intel Xeon E5-2699 v3 CPUs each. The blades differ from the other systems as they make use of shared PSUs, provisioned to support up to 16 servers. CPU frequency scaling (including Turbo) and all other BIOS power saving mechanisms have been turned on. For our experiments, all machines use a common operating system (OS) in Windows Server 2012 R2.

IV. Load Distribution of SERT Worklets

We create target load distributions by modifying SERT to override the target load percentage on a client-by-client basis. Clients then look up their local target load level for the current global load level and their specific client ID at the beginning of each interval. As clients are bound to a specific SMT unit and physical CPU, specifying the correct client allows the stressing of each specific core. The operating system ensures that the client to CPU mapping remains identical over the separate experiments. Separate configurations can be deployed on each host (in our case, server), allowing different load distribution behaviors for each host.

We define the following policies for load distribution. Each policy can be applied individually on every level of the load distribution hierarchy (e.g., servers, sockets). We create a strategy compositon by selecting one of these policies for each hierarchy level:

Balanced: Transaction counts are set to be equal for all clients, resulting in a balanced load across all systems.

Consolidated: Transactions are consolidated on as few clients as possible. As a result, all clients, with the exception of one, are either idle or at full utilization.

Energy Efficient Consolidation: This new strategy, introduced in this paper, keeps as many clients as possible at the point of maximum energy efficiency. Specifically, we consolidate load on clients, with the upper load boundary for each client being the predetermined (calibrated) point of maximum energy efficiency for the given workload. Only once all clients have reached this point, do we start to increase client load with rising global load. At this point, the extra load is still consolidated on as few clients as possible. This strategy is identical to the consolidation strategy if maximum energy efficiency for the given workload is achieved at full utilization.

Any of these strategies can be applied at any level of the load distribution hierarchy. The following hierarchical execution units are targeted: Full servers, CPU sockets, CPU cores, and logical CPU cores (also called Hyperthreading or SMT units).

To enable the execution of heterogeneous workloads, we also allow overriding of the client's executed worklet. At the beginning of each scenario, the client will check if it has been assigned a worklet other than the global one. In this case, it will replace the global worklet with its own local worklet. When doing so, calibration results become invalid, as calibration is always conducted with homogeneous workloads to eliminate the influence of inter-worklet interference. To cope with this, we provide each client with a pre-set calibration result. This calibration result has been predetermined with separate calibration runs with an unmodified SERT for each of the worklets in the final workload. Each client's 100%

target load level is then set to the calibration result of the corresponding client and workload during the unmodified run. The host configuration also includes the possibility of turning SERT's affinity provider off. In this case, clients are no longer bound to specific cores and may be redeployed by the operating system at runtime.

V. Energy Efficiency for Homogeneous Workloads

We evaluate the power consumption and energy efficiency of hierarchical load distribution for homogeneous workloads on multiple systems. First we evaluate different combinations of load distribution on the core and logical processor level using a single-socket system. The balancing and consolidation strategies are evaluated for all worklets, and the efficiency strategy is applied for the LU worklet. We then test the impact of socket-level load distribution on Ivy Bridge and Haswell systems, introducing additional load distribution combinations with the added level in the distribution hierarchy. We also evaluate load distribution with operating system level load migration, before finally investigating the effects of node level load management.

A. Load Distribution for all Worklets on Single-Socket System

Load distribution is achieved by varying the target throughput per logical processor with the goal of achieving the pre-specified global load level. This target load level is achieved with consistency for all distribution strategies. The coefficient of variation (CV) for any given worklet throughput at a target load level never exceeds 3.3%, with an average CV of 0.7%. As a result, energy efficiency of the distributions is primarily influenced by power consumption.

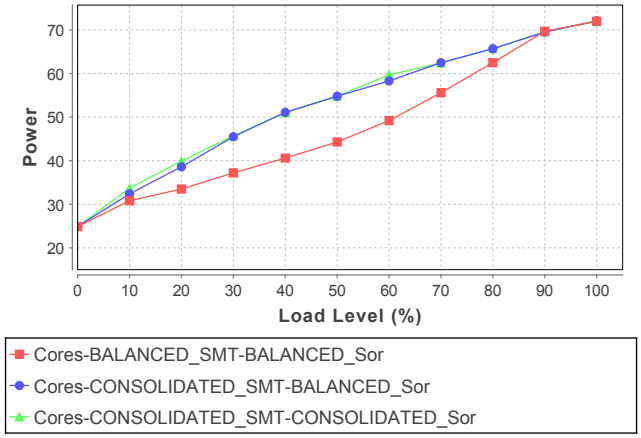

Fig. 2. Power consumption of distributions for SOR on single-socket system.

The load distribution strategies' power consumption is shown in Figure 2. It can differ significantly depending on load level. Specifically, the balanced distribution outperforms the consolidation strategies between 30% and 70% load. This observation can be explained by the CPUs frequency scaling. State of the art processors feature dynamic frequency scaling mechanisms designed to increase performance on a subset of available cores in case of uneven load distribution. This feature is usually referred to as "Turbo" and causes single CPU cores to dynamically overclock as long as a number of thermal and power constraints are met. Core-wise load consolidation causes single cores to quickly reach a load level that triggers the core's turbo. In the case of our quad-core system, a global load level of 20% already causes a local load level of 80% on core 0. At this point, power consumption diverges and energy efficiency drops in comparison to a balanced load distribution (as can be seen in Figure 3).

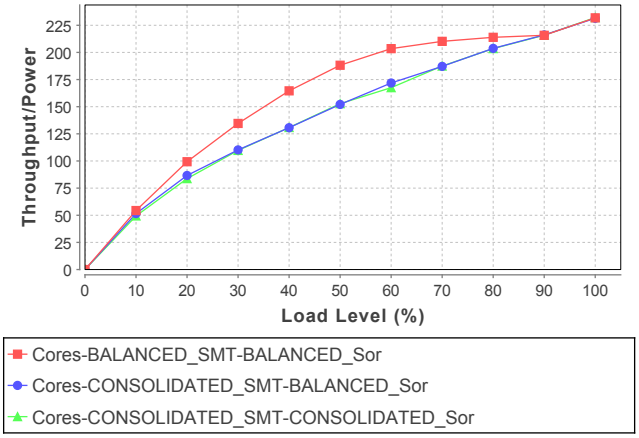

Fig. 3. Energy Efficiency of distributions for SOR on single-socket system.

However, power consumption and energy efficiency is not affected significantly by load consolidation on the level of SMT units. While SMT threads are provided and executed in hardware, they are not pinned to an exclusive set of execution units in the same sense as a physical CPU core. Instead, logical processors on the same core share their execution units. Subsequently, redistributing load between those logical processors on the same core has little impact for homogeneous loads, as the same execution units remain in use.

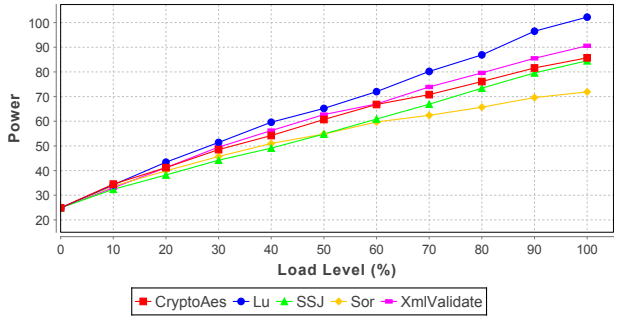

Fig. 4. Power consumption of fully consolidated load on single-socket system.

We were able to repeat the observations made for the SOR worklet for all other worklets. All worklets exhibited maximum power consumption when using consolidation strategies and maximum energy efficiency when using a balancing strategy. Power consumption for all worklets using the fully consolidated strategy is displayed in Figure 4. The worklet's order in power consumption remains identical over utilization levels and distribution strategies, with one exception. As the SSJ worklet is not a purely CPU bound workload, but a hybrid workload emulating typical business transaction software, it contains a number of non-CPU bottlenecks. As a result, it scales differently with increasing load compared to the CPU bound worklets. It is also affected differently by the selected load distribution strategy. Specifically, it always consumes more power than SOR using the fully balanced strategy, yet consumes less power between the 10% and 50% load levels using the fully consolidated strategy.

B. Efficiency Strategy on Single-Socket System

We evaluate the efficiency strategy using the LU worklet, as LU reaches the point of optimal energy consumption on a load level less than 100% on the majority of systems. On the single-socket system the load level for maximum energy efficiency with LU is 60% with 264.0 transactions/watt. However, energy efficiency at full load is only slightly worse at 260.5 transactions/Watt.

In contrast to the multi-socket system in section V-D, the new strategy is not as efficient as the balanced strategy on the single-socket system. It is, however, more efficient than full load consolidation, confirming the energy inefficiency of forced turbo overclocking through load consolidation. Once the point of maximum energy efficiency is passed, the strategy allows consolidation of load up to 100%, thus displaying the power consumption of the consolidation strategies.

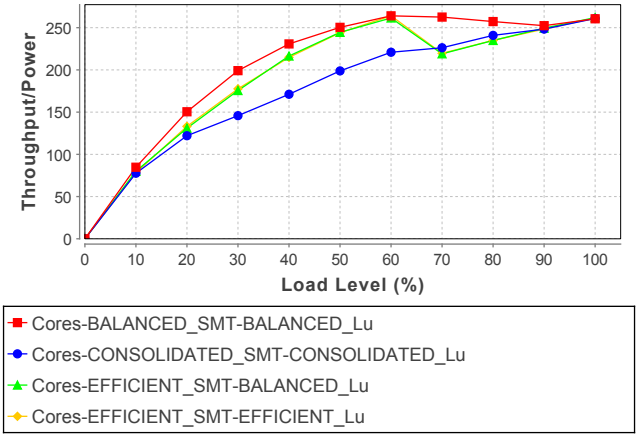

Fig. 5. Power consumption of efficiency strategies on single socket system.

C. Load Distribution for all Worklets on Dual-Socket System

The dual-socket systems offer an additional layer on the load distribution hierarchy, as load can be distributed on a socket-by-socket basis. Performance variation depending on the selected strategy remains minimal with an average CV of 0.4% on the Haswell and 0.5% on the Ivy Bridge System.

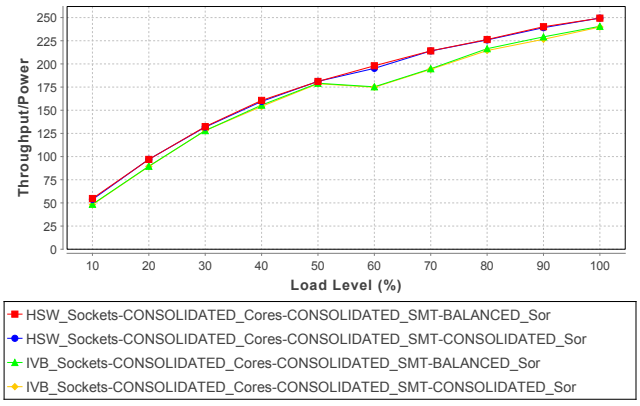

Fig. 6. Energy efficiency of consolidated strategy with and without SMT unit consolidation on both dual-socket systems for SOR.

Measurements on the dual-socket systems, also confirm that load distribution on SMT units within the same core has no effect on power consumption and energy efficiency. This observation remains true for both Haswell and Ivy Bridge architectures across all worklets and strategies. A selected comparison is shown in Figure 6. As a result, we will only display the results for strategy compositions using a balancing strategy on the SMT level for all following measurements.

The dual-socket Ivy Bridge measurement results in Figure 7 are representative for the results with all workloads. Keeping the workloads fully balanced on all levels is still the most efficient strategy, as it was for the single-socket system. However, balancing is not always the most efficient sub-strategy in all cases. When core-level load consolidation is employed, load consolidation on a socket level improves energy efficiency at lower load levels, meaning that full consolidation on both the socket and CPU levels beats a CPU level consolidation on balanced sockets. This can be explained by the observation that the amount of cores onto which work is consolidated for both of these strategies remains the same. The difference is only in the location of the given cores. When consolidating sockets, only cores on one socket are activated, allowing cores on the other socket to remain in more energy efficient modes and, more importantly, keeping them from going into turbo. Balancing cores on consolidated sockets improves energy efficiency further, as cores are kept in their most energy efficient load range.

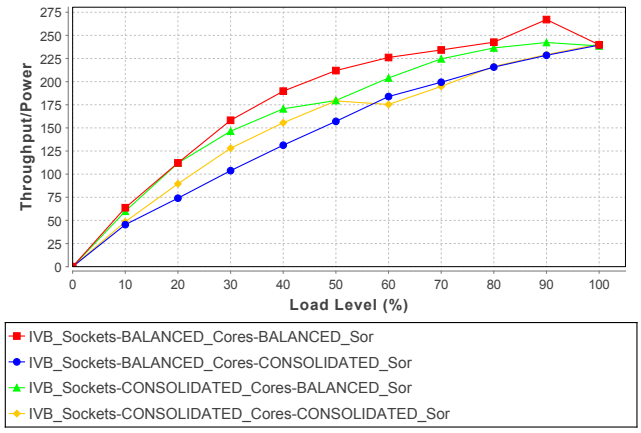

Fig. 7. Energy efficiency of load distribution strategies on Ivy Bridge dual-socket systems for SOR.

The dual-socket Haswell measurement results in Figure 8 are the first results to significantly deviate from one of our principal previous observations, as the balanced strategy is not the reliably best strategy at all load levels. Socket level load balancing improves energy efficiency for all workloads making it more efficient than pure load balancing at high load. This effect is visible to a different extent depending on the specific workload. It can differ both in the range of load levels for which consolidation is superior to balancing, as well in the amount of energy saved. A comparison of these factors is shown in Table I.

Worklet	Load level range	Avg. power difference	Max. power difference
CryptoAES	80% – 90%	12.3 W	16.3 W
LU	80% – 90%	30.0 W	34.4 W
SOR	80% – 90%	11.5 W	16.6 W
SSJ	90%	1.6 W	1.6 W
XMLValidate	90%	10.3 W	10.3 W

TABLE I. LOAD LEVEL RANGE WHERE SOCKET CONSOLIDATION HAS A LOWER POWER CONSUMPTION THAN FULLY BALANCED LOAD (INCLUSIVE). THE SHOWN DIFFERENCES ARE POWER CONSUMPTION DIFFERENCES WITHIN THIS RANGE.

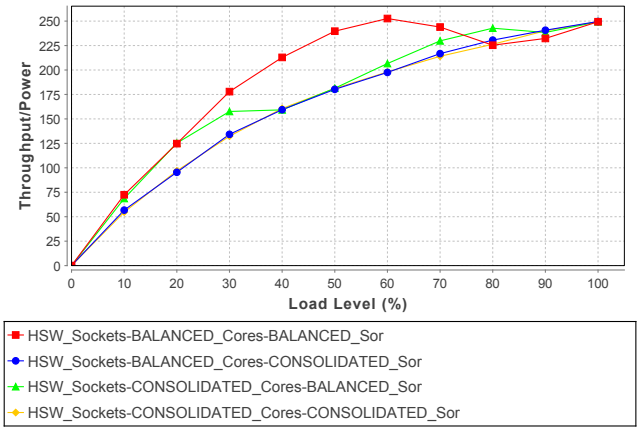

Fig. 8. Energy efficiency of load distribution strategies on Haswell dual-socket systems for SOR.

Core-level load consolidation is always less efficient than the same strategy composition without core level consolidation. However, some workloads (SOR, LU, and CryptoAES) gain such a significant increase in energy efficiency from using socket level consolidation that consolidation on all levels still outperforms the fully balanced strategy at high load. Similarly, these worklets also profit from core-wise load balancing at high load: It is, however, not as efficient as socket level consolidation only.

D. Efficiency Strategy on Dual-Socket System

Effectiveness of the efficiency strategies varies depending on processor architecture. On Ivy Bridge, maximum energy efficiency for LU is reached at the 80% load level (see Figure 9). Socket-level load distribution using the efficiency strategy features better energy efficiency than full socket-level consolidation, yet still doesn't reach the energy efficiency of a fully balanced system.

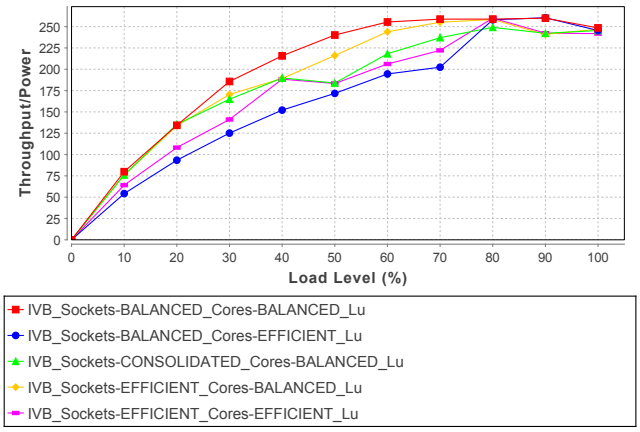

Fig. 9. Energy efficiency of load distribution strategies on Ivy Bridge dual-socket system for LU.

The Haswell dual-socket system reacts in a completely different way to the efficiency strategies, as shown in Figure 10. Particularly, the socket level efficiency strategy features better energy efficiency than the fully balanced distribution and the socket level consolidation strategy at a number of load levels. LU's load level of maximum energy efficiency is 50%. Per definition, the efficiency strategy and the fully balanced strategy result in an identical load distribution at this load level. At lower load than 50% the fully balanced approach still remains the most energy efficient. At those load levels, the efficiency strategy attempts to keep as many sockets as possible at 50% load, which leads to partial load consolidation and consumes more power than a balanced load at less than 50%. Beyond 50% load, the efficiency strategies attempt to consolidate work on as few units as possible, while keeping the rest at 50%. On the socket level this means that one socket is kept at 50% load, while the other increases in load. Beyond the 75% load level, the first socket is fully utilized and the efficiency strategy behaves identical to the consolidation strategy, as it can now only increase load on the remaining socket. At 60% load the strategy of keeping a socket at maximum energy efficiency pays off, resulting in greater efficiency than using pure balancing. Beginning at the 70% global load level the socket level efficiency strategy starts distributing load similarly (and then identically) to the socket level consolidation strategy. Consequently, the energy efficiency of these two strategies behaves the same.

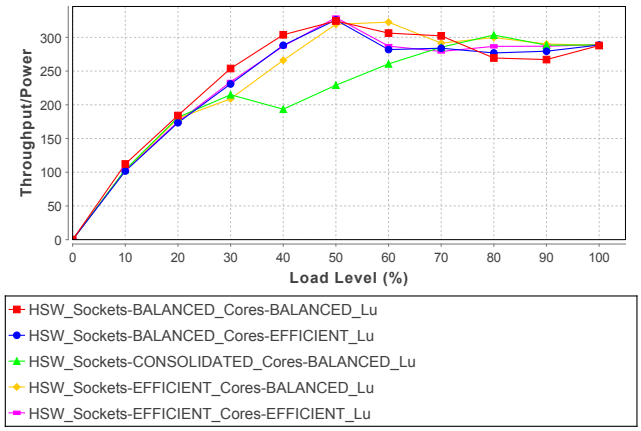

Fig. 10. Energy efficiency of load distribution strategies on Haswell dual-socket system for LU.

On our dual-socket Haswell system, the efficiency load distribution strategy saves up to 33 W (10.7% relative savings) in comparison to the fully balanced strategy. Within the interval between maximum energy efficiency (50%) and full load (both exclusive) it allows for an average energy saving of 15.4 W (5.8% relative savings). In conclusion, we recommend a mixed load balancing strategy for a multi-socket Haswell system. We recommend a fully balanced strategy up to the load level of maximum energy efficiency. At greater loads our efficiency strategy performs better and should be used instead.

E. No CPU Pinning

The OS features additional mechanisms, which can be used once CPU pinning is disabled. Specifically, it is able to migrate running threads from one core to another. This way the local CPU utilization can be changed to a level, which is seemingly independent of the actual thread loads, as threads can be continuously relocated at run-time. We disable CPU pinning to analyze the effect of this behavior on power consumption.

For the previous measurements using CPU affinity, most observations were observable over all worklets and only varied in their respective impact. Non-pinned distribution shows far greater variability depending on the workload. Figure 11 shows the energy efficiency of a few selected strategies using the

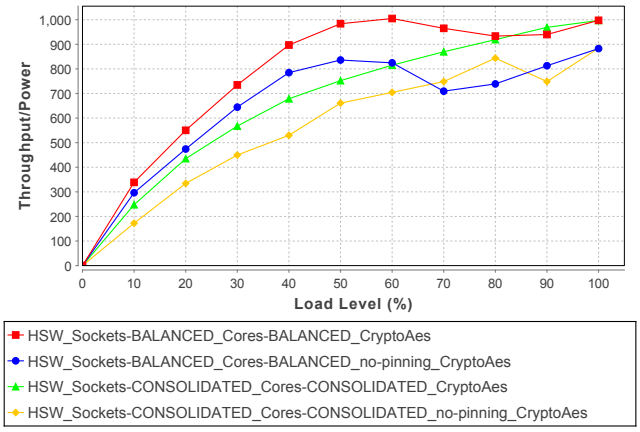

Fig. 11. Energy efficiency of load distribution strategies, including non-pinning strategies, on Haswell dual-socket system for CryptoAES.

CryptoAES worklet. For this specific worklet, non-pinned strategies cannot match the energy efficiency of the fully balanced strategy using CPU pinning. There are still significant differences in the energy efficiency of the different non-pinning strategies, as thread migration does not cancel unevenly distributed load.

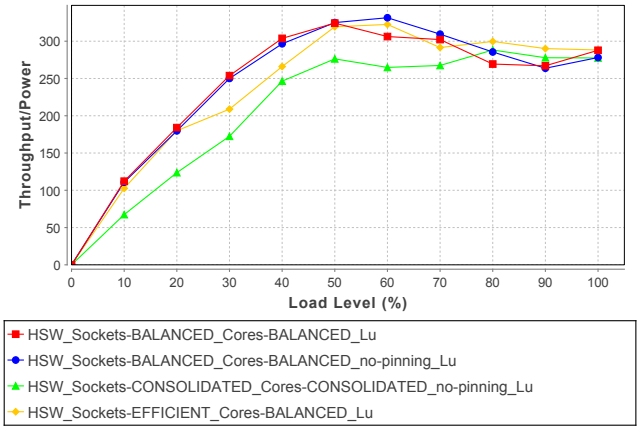

Fig. 12. Energy efficiency of load distribution strategies, including non-pinning strategies, on Haswell dual-socket system for LU.

LU (see Figure 12) behaves completely different, as the non-pinned balanced strategy performs with greater energy efficiency than the pinned strategy at high loads. It even outperforms the efficiency strategy at 60% load. It does, however, not perform as efficiently as the efficiency strategy at loads of 80% and greater. For the other worklets, which were not evaluated for the efficiency strategy, the non-pinned balanced strategy performs equal to (SOR) or better (SSJ, XMLValidate) than its pinned counterpart. The energy efficiency of pinned socket-level consolidation at 90% load is always greatest, though.

F. Multi-Node Load Distribution

In contrast to hardware components within a system, separate server nodes executing independent units of work without any common external resources do not feature any contention. As a result, we can evaluate the energy efficiency of multi-node distribution strategies by simulating multi-node performance and power from single system measurements. For each global load level, we calculate the load that must be reached by any single node within the cluster. We then read the measured power and performance data for the given local load level and compute cluster wide power and performance. Performance variation is again negligible with an average CV of 0.2% over all strategies.

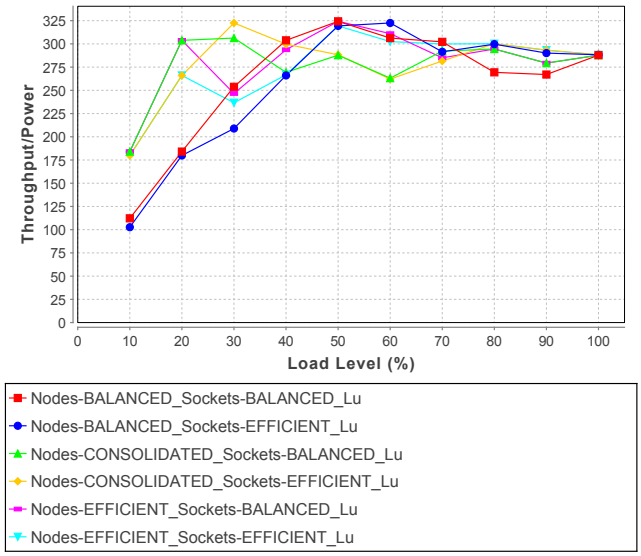

Fig. 13. Energy efficiency of load distribution strategies on two nodes using the Haswell dual-socket system for LU. Unused nodes are turned off.

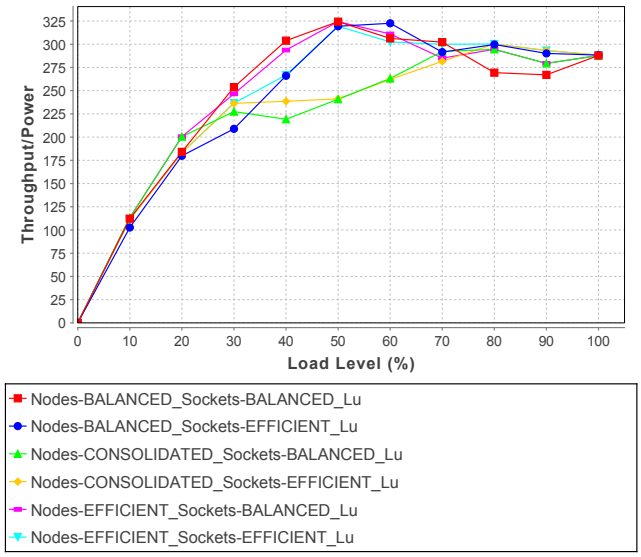

Fig. 14. Energy efficiency of load distribution strategies on two nodes using the Haswell dual-socket system for LU. Unused nodes remain in an idle state.

Figure 13 shows the energy efficiency of multi-node distribution strategies when unused systems are turned off. Figure 14, on the other hand, assumes that unused nodes remain in an idle state. Most notably, node level load consolidation only leads to maximum energy efficiency when unused nodes are switched off. Even then it is only the best strategy at low load up to 30 – 40%. In the 40% to 70% load range it is less efficient than a number of strategies, including balancing and efficiency strategies. At the highest load levels it shares maximum efficiency with node level balancing in combination with the socket level efficiency distribution. Load consolidation loses its advantage at low loads once unused systems are left idling. In that case it is only slightly more efficient than load balancing and equally as efficient as the node-level efficiency strategy at 20% load.

Figure 15 shows the power consumption of the multi-node results, as measured on our two HP servers. These results differ significantly from the previously simulated results. On these systems load consolidation consumes more power than balanced load at low load levels, whereas it saves power at higher loads. We attribute most of this difference to the Xeon E5-2699 v3 CPUs, which operate within the HP servers. These CPUs have a lower frequency (2.3 GHz) in comparison to the Xeon E5-2667 v3 CPU (3.2 GHz) yet feature 18 physical cores per socket in comparison to the 8 cores of the Xeon E5-2667 v3. Another influencing factor for the blade's power consumption is the shared power infrastructure, including shared PSUs.

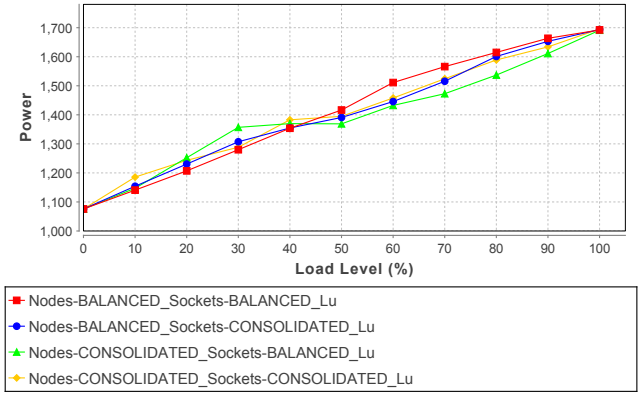

Fig. 15. Power consumption of load distribution strategies on two HP dual-socket nodes for LU. Unused nodes remain in an idle state.

G. Conclusions

The measurement results using homogeneous workloads show that our new efficiency load distribution strategy, applied on the socket level, can reduce power consumption of servers at high utilization. To a lesser degree, load consolidation on the socket level can also improve energy efficiency at high utilization. In contrast, a balanced load is the best choice for low load levels and the CPU core level. Multi-node systems benefit from load consolidation at low loads as unused systems can enter power saving states or shut down. We also demonstrate that CPU-pinned strategies perform equally or better than their non-pinned counterparts.

VI. Energy Efficiency for Heterogeneous Workloads

We evaluate the energy efficiency of concurrently executing independent workloads using the previously introduced distribution strategies. In addition, we allow the specific worklets to be pinned to logical processors on given cores. As each core on our SUTs features two logical processors, worklets can be either deployed on a core with a worklet of a different type already running on it (**mixed**) or it can be deployed on a core with another instance of the same worklet running concurrently (**clean**). For our experiments, we use the SOR and CryptoAES worklets, as they are sufficiently heterogeneous featuring different performance and energy bottlenecks [25].

Figure 16 shows the energy efficiency for the different distribution strategies on the single-socket Haswell based system. Performance impact of the distribution strategies is very small again, with an average CV of 0.1% for each load level over

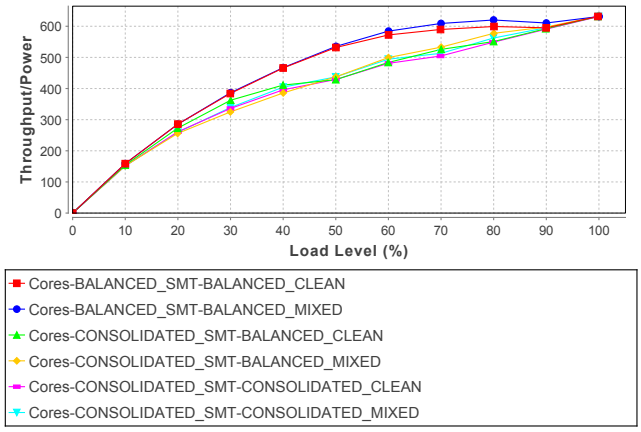

Fig. 16. Energy efficiency of heterogeneous load distribution strategies on Haswell single-socket system.

the different strategies, including mixed and clean strategies. Power consumption is influenced, however, resulting in different efficiency for the strategies. For the balanced strategy, mixing worklets onto the same core has a positive effect on efficiency, as hyperthreading distributes used and unused execution units in a more efficient manner. Heterogeneous workloads offer more options for efficient hyperthreading as heterogeneous workloads tend to use less of the same execution units, enabling concurrent hardware multi-threading on the same core. Heterogeneous load distribution on the dual-socket system (see Figure 17) confirms observations from the single-socket result. Mixed execution of worklets remains more energy efficient for the balanced strategy. Additionally, balanced load distribution is still less energy efficient than consolidated loads at high utilization, as it was for homogeneous workloads.

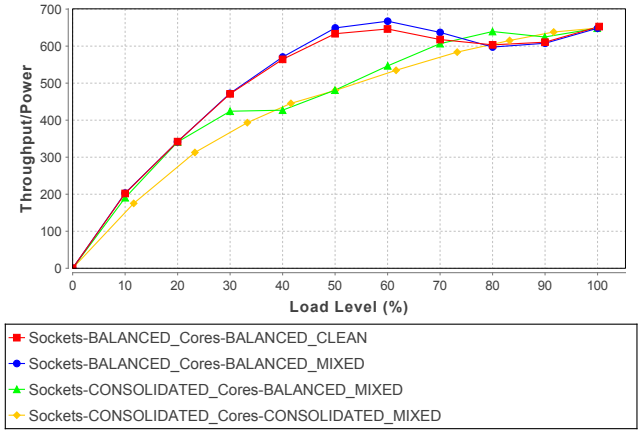

Fig. 17. Energy efficiency of heterogeneous load distribution strategies on Haswell dual-socket system.

Concluding, the observations made for homogeneous workloads remain valid for heterogeneous workloads. In addition, energy efficiency can be improved by deploying different workload types onto the same core.

VII. Conclusions

This paper demonstrates the impact of hierarchical load distribution on the energy efficiency of both homogeneous and heterogeneous workloads. We evaluate load distribution on four levels of the computational hierarchy (servers, sockets, CPU cores, and SMT units). For each of those levels one

of three basic strategies is selected, creating a hierarchical strategy composition. The three basic strategies are: load balancing, load consolidation, and an energy-efficient consolidation strategy, introduced in this paper. We show that a strategy composition using our new efficient distribution can save up to 10.7% power, depending on load level and workload.

The observations in this paper enable the creation of new load balancers on both the application and operating system levels. These balancers would be capable of maximizing energy efficiency by dynamically selecting appropriate load distribution strategies based on hardware, workload, and load level.

VIII. Acknowledgments

The authors also wish to acknowledge current and past members of the SPECpower Committee who have contributed to the design, development, testing, and overall success of SERT: Ashok Emani, Sanjay Sharma, Mike Tricker, Greg Darnell, Karl Huppler, Van Smith, Nathan Totura, Cloyce Spradling, Paul Muehr, David Ott, Cathy Sandifer, Jason Glick, and Dianne Rice, as well as the late Alan Adamson, and Larry Gray.

References

[1] C. Babcock. NY Times data center indictment misses the big picture. 2012.

[2] L. Barroso and U. Holzle. The Case for Energy-Proportional Computing. *Computer*, 40(12):33–37, Dec 2007.

[3] F. Chen, J. Grundy, Y. Yang, J.-G. Schneider, and Q. He. Experimental Analysis of Task-based Energy Consumption in Cloud Computing Systems. In *Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering*, ICPE '13, pages 295–306, New York, NY, USA, 2013. ACM.

[4] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing Server Energy and Operational Costs in Hosting Centers. In *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '05, pages 303–314, New York, NY, USA, 2005. ACM.

[5] B. Dorronsoro, S. Nesmachnow, J. Taheri, A. Y. Zomaya, E.-G. Talbi, and P. Bouvry. A hierarchical approach for energy-efficient scheduling of large workloads in multicore distributed systems . *Sustainable Computing: Informatics and Systems*, 4(4):252 – 261, 2014. Special Issue on Energy Aware Resource Management and Scheduling (EARMS).

[6] M. Gomaa, M. D. Powell, and T. N. Vijaykumar. Heat-and-run: Leveraging SMT and CMP to Manage Power Density Through the Operating System. In *Proceedings of the 11th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS XI, pages 260–270, New York, NY, USA, 2004. ACM.

[7] J. L. Henning. SPEC CPU2000: Measuring CPU Performance in the New Millennium. *Computer*, 33(7):28–35, Jul 2000.

[8] S. Heo, K. Barr, and K. Asanovic. Reducing power density through activity migration. In *Low Power Electronics and Design, 2003. ISLPED '03. Proceedings of the 2003 International Symposium on*, pages 217–222, Aug 2003.

[9] D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, and G. Jiang. Power and Performance Management of Virtualized Computing Environments Via Lookahead Control. In *Autonomic Computing, 2008. ICAC '08. International Conference on*, pages 3–12, June 2008.

[10] K.-D. Lange. Identifying Shades of Green: The SPECpower Benchmarks. *Computer*, 42(3):95–97, March 2009.

[11] K.-D. Lange, J. A. Arnold, H. Block, N. Totura, J. Beckett, and M. G. Tricker. Further Implementation Aspects of the Server Efficiency Rating Tool (SERT). In *Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering*, ICPE '13, pages 349–360, New York, NY, USA, 2013. ACM.

[12] K.-D. Lange and M. G. Tricker. The Design and Development of the Server Efficiency Rating Tool (SERT). In *Proceedings of the 2nd ACM/SPEC International Conference on Performance Engineering*, ICPE '11, pages 145–150, New York, NY, USA, 2011. ACM.

[13] K.-D. Lange, M. G. Tricker, J. A. Arnold, H. Block, and C. Koopmann. The Implementation of the Server Efficiency Rating Tool. In *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, ICPE '12, pages 133–144, New York, NY, USA, 2012. ACM.

[14] L. Lefèvre and A.-C. Orgerie. Designing and Evaluating an Energy Efficient Cloud. *J. Supercomput.*, 51(3):352–373, Mar. 2010.

[15] R. Nathuji and K. Schwan. VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems. In *Proceedings of Twenty-first ACM SIGOPS Symposium on Operating Systems Principles*, SOSP '07, pages 265–278, New York, NY, USA, 2007. ACM.

[16] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath. Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems, 2001.

[17] A. Podzimek, L. Bulej, L. Y. Chen, W. Binder, and P. Tůma. Analyzing the Impact of CPU Pinning and Partial CPU Loads on Performance and Energy Efficiency. In *Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, CCGRID 2015, 2015. Accepted for publication.

[18] M. Poess, R. O. Nambiar, K. Vaid, J. M. Stephens Jr, K. Huppler, and E. Haines. Energy benchmarks: a detailed analysis. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, pages 131–140. ACM, 2010.

[19] D. Quan, R. Basmadjian, H. Meer, R. Lent, T. Mahmoodi, D. Sannelli, F. Mezza, L. Telesca, and C. Dupont. Energy Efficient Resource Allocation Strategy for Cloud Data Centres. In E. Gelenbe, R. Lent, and G. Sakellari, editors, *Computer and Information Sciences II*, pages 133–141. Springer London, 2012.

[20] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu. No "Power" Struggles: Coordinated Multi-level Power Management for the Data Center. In *Proceedings of the 13th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS XIII, pages 48–59, New York, NY, USA, 2008. ACM.

[21] S. Rivoire, M. A. Shah, P. Ranganathan, and C. Kozyrakis. JouleSort: A Balanced Energy-efficiency Benchmark. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, pages 365–376, New York, NY, USA, 2007. ACM.

[22] S. Srikantaiah, A. Kansal, and F. Zhao. Energy Aware Consolidation for Cloud Computing. In *Proceedings of the 2008 Conference on Power Aware Computing and Systems*, HotPower'08, pages 10–10, Berkeley, CA, USA, 2008. USENIX Association.

[23] Standard Performance Evaluation Corporation. SPEC Power and Performance Benchmark Methodology. http://spec.org/power/docs/SPEC-Power_and_Performance_Methodology.pdf.

[24] A. Verma, P. Ahuja, and A. Neogi. pMapper: Power and Migration Cost Aware Application Placement in Virtualized Systems. In V. Issarny and R. Schantz, editors, *Middleware 2008*, volume 5346 of *Lecture Notes in Computer Science*, pages 243–264. Springer Berlin Heidelberg, 2008.

[25] J. von Kistowski, H. Block, J. Beckett, K.-D. Lange, J. A. Arnold, and S. Kounev. Analysis of the Influences on Server Power Consumption and Energy Efficiency for CPU-Intensive Workloads. In *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (ICPE 2015)*, ICPE '15, New York, NY, USA, February 2015. ACM.

[26] J. von Kistowski, N. R. Herbst, D. Zoller, S. Kounev, and A. Hotho. Modeling and Extracting Load Intensity Profiles. In *Proceedings of the 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2015)*, May 2015. Accepted for publication.