# Model-based Optimization of the Energy Efficiency of Multi-threaded Applications

Thomas Rauber
Computer Science Department,
Universität Bayreuth,
95440 Bayreuth, Germany
rauber[at]uni-bayreuth.de

Gudula Rünger
Computer Science Department,
Technische Universität Chemnitz,
09107 Chemnitz, Germany
ruenger[at]cs.tu-chemnitz.de

Matthias Stachowski
Computer Science Department,
Universität Bayreuth,
95440 Bayreuth, Germany
stachowski[at]uni-bayreuth.de

**To cite this version**:
Thomas Rauber, Gudula Rünger, Matthias Stachowski: Model-based Optimization of the Energy Efficiency of Multi-threaded Applications. Sustainable Computing: Informatics and Systems, Vol. 22: pp. 44-61. Elsevier, ISSN 2210-5379, 2019. DOI: 10.1016/j.suscom.2019.01.022

**Abstract**

Energy efficiency is considered to be a critical concern for modern hardware and a variety of hardware features have been developed to improve the energy balance for executing applications. This article focuses on the dynamic voltage and frequency scaling (DVFS) technique, which is available for many platforms. Analytical models for capturing the energy efficiency are considered and it is investigated whether such an analytical model is able to support an a priori selection of the operational frequency that leads to a near optimal energy consumption for the application code to be executed. Furthermore, the possible influence of the number of threads executing a multi-threaded application is taken into account. Models for the energy-delay product (EDP) weighting the power against the square of execution time are also covered. The experimental evaluation is performed on the basis of the multi-threaded PARSEC and SPLASH-2 benchmarks executed on four different recent multi-cores. The investigations show that the operational frequency selected according to the analytical models leads to an energy consumption that is near the minimum energy consumption over all frequencies available.

## 1 Introduction

A low energy consumption as well as a good performance are important properties of today‘s application codes. These properties are influenced by the software structure and execution characteristics, such as the operational frequency or the number of threads chosen, and the way in which the code exploits the hardware details of the execution platform. The internal organization of the processor architecture can have a large influence on the resulting execution time and energy consumption, depending on how well the operations defined by the application code are mapped onto the functional units of the processor. However, the influence of the multitude of different software and hardware specifics as well as their interactions on the quantitative and qualitative behavior of performance and energy consumption can hardly be captured in its entirety. In such cases, it is more worthwhile to develop an abstract model based on observed data with which a priori estimations of performance and energy are possible. In this article, we are concerned with the determination of such models for multi-threaded applications executed on multi-cores with frequency scaling.

The growing importance of energy concerns has encouraged hardware manufacturers to develop a variety of power-aware system features, including multicore-on-a-chip processors, core-independent functional units, dynamic voltage and frequency scaling (DVFS), or clock gating [37, 13, 14]. This article concentrates on the DVFS technique and the effect of frequency scaling on the energy consumption of application executions. It can be observed that it is not a priori clear whether the use of a smaller operational frequency leads to a reduction or an increase of the energy consumption. For multi-threaded applications, the number of threads used for an execution can also have an effect on the time and energy consumed. Again, it is not a priori clear whether the use of more threads may lead to a shorter execution time and/or a lower energy consumption or to the opposite effect. Instead, for a given application there is typically an optimal number of threads beyond which the execution time cannot be reduced further. Similarly, we have observed that there is often an optimal operational frequency beyond which a further frequency reduction does not lead to a further energy reduction. In this article, we investigate the effect of an operational frequency chosen and/or a number of threads used on the performance of time and the energy consumption for two popular multi-threaded benchmark suites, which are the PARSEC and the SPLASH-2 benchmarks.

To control the usage of the DVFS feature, the interactions between the operational frequency used and the resulting energy consumption and runtime performance have to be known. An ideal situation would be to have an analytical model that exactly captures the effect of all influencing parameters. However, no such model exists, and it is even difficult to identify all influencing factors or their quantitative effects. The idea of this article is to take a step towards the development of such an analytical model. In particular, we consider a power model that has been proposed to capture the power consumption of CMOS chips and, based on this power model, further models for the energy consumption and the energy-delay product (EDP) are proposed. The EDP has been introduced as a single metric balancing the effects of execution time and energy consumption of applications and ,thus, it is often used as a measure for energy efficiency. We investigate whether the models are able to a priori select a suitable operational frequency that leads to a near-optimal energy consumption or EDP. We also address the question whether and how the optimal frequencies for energy and EDP are correlated, which reflects the relation between energy consumption and energy efficiency.

The investigation is based on the Princeton Application Repository for Shared-Memory Computers (PARSEC)[6] and the Stanford Parallel Applications for Shared-Memory (SPLASH-2) [40], which are both collections of multi-threaded benchmarks with different parallel workloads and execution characteristics. The PARSEC benchmark suite provides programs from a wide range of applications based on the latest techniques in the specific domains and with an emphasis on large workloads as well as multi-core parallelism. Thus, the PARSEC benchmarks are ideal for demonstrating and testing the analytical energy models proposed for a large variety of applications. To have a wider experimental basis, we also investigate the SPLASH-2 benchmarks, which cover applications from several areas of scientific computing. As hardware platforms, four Intel processors with different architecture are used for the experimental evaluation. Model-based power, energy and EDP values are calculated and are compared with corresponding measurements on these processors.

In this article, two analytical models are proposed, which are a so-called local modeling providing an application-specific model and a global modeling providing an application-independent modeling based on an independent test set for determining the parameters for the energy model. Our investigations have shown that the frequencies derived a priori with the energy models have the property to lead to a near-optimal energy consumption. The resulting average deviation from real measured data is below 6 % for the energy consumption and below 11 % for the EDP for almost all of the benchmarks. The results indicate that a quite accurate application-specific model-based prediction of energy-efficient frequencies is possible for a wide range of multi-threaded applications.

The frequency investigation is combined with an investigation of the number of threads used for the execution. This means that the determination of an optimal frequency depends on the number of threads used, providing a whole range of optimal frequency values for the same application. An important result is that the development of the application-specific energy models and the derivation of the energy-optimal frequency is independent from knowledge about the internal coding of the application. Thus, the model-based prediction method of an energy-optimal frequency can be applied to other multi-threaded applications.

The contributions of the article comprise:

- Models for power and energy from [31] are extended to the multi-threaded case resulting in power and energy functions of two variables, which are the operational frequency and the number of executing threads.

- The analytical energy model is used to a priori determine an energy-optimal frequency for varying numbers of threads.
- The energy model is extended to the EDP to determine an energy-optimal frequency for the energy efficiency.
- The energy consumption for the PARSEC and SPLASH-2 benchmark suites is measured and documented.
- Application-specific optimal frequencies are determined with the analytical models for energy and for the EDP, both for the PARSEC and the SPLASH-2 benchmarks and the good prediction quality of the models is shown.
- Application-independent optimal frequencies are derived with a small test set of benchmarks and are validated for the remaining benchmarks.
- The dependence on the number of threads is is investigated and scalabilty results are presented.

The rest of the article is structured as follows: Section 2 describes the energy model and derives optimal frequency scaling factors for energy consumption and for the EDP. Section 3 describes the experimental environment and the evaluation strategy used. Section 5 presents the experimental evaluation for the PARSEC and for the SPLASH-2 benchmarks. Section 6 discusses related work and Section 7 concludes the article.

# 2 Parallel energy models

This section is devoted to energy models for the parallel execution of multi-threaded application programs. In particular, we consider the dependence of the energy consumption and the runtime performance from two parameters, which are the number of threads employed and the operational frequency chosen. Usually, the runtime performance improves with an increasing number of threads but decreases with a down-scaling of the frequency. Also the energy consumed for an application execution varies when varying the number of threads or when scaling the frequencies. And although the energy consumed depends on the execution time, it can be observed that the lowest execution time achieved for a specific setting of the number of threads and operational frequency does not necessarily lead to the lowest energy consumption. This effect can be explained by an application-specific power consumption. In this section, we investigate an energy model which captures the quantitative energy effects of frequency scaling.

The energy $E$ consumed for the execution of an application code depends on the execution time $T$ and the power drawing $P$ during the execution. The execution time $T$ varies with the number of threads $p$ used for the execution and the operational frequency $f$ chosen, as does the power drawing $P$. To express the dependence of the energy consumption on the number of threads and the operational frequency, we develop an analytical energy model which involves the two variables $p$ and $f$.

Section 2.1 revisits a power and energy model from [31] which had only the frequency as parameter. In Section 2.2, the models are extended to a second parameter, which is the number $p$ of threads used for the execution. For each $p$, an optimal frequency is determined analytically. In Section 2.3, the energy delay product depending on the two parameters is considered and optimal frequencies are calculated as well. The notation used throughout the section is compactly depicted in Table 1.

## 2.1 Power and energy model for frequency scaling

When considering the energy consumption of application programs for DVFS processors, it is useful to introduce frequency scaling factors [31]: The frequency scaling for a DVFS processor is expressed by a dimensionless scaling factor $s \geq 1$ which describes a smaller frequency $\tilde{f} < f_{max}$ as $\tilde{f} = f_{max}/s$ where $f_{max}$ is the maximum frequency available.

Power models for DVFS processors distinguish the dynamic power consumption and the static power consumption [42]. The dynamic power consumption $P_{dyn}(f)$ is related to the supply voltage and the switching activity during the computing activity of the processor and can be expressed by

Table 1: Summary of Notation

| Notation | | |
|---|---|---|
| SYMBOL | UNIT | MEANING |
| $E$ | Joule | energy consumption |
| $T$ | seconds | execution time |
| $P$ | Watt | power consumption |
| $p$ | scalar | number of threads |
| $f$ | 1/seconds | operational frequency |
| $f_{max}$ | 1/seconds | maximum processor specific frequency |
| $s \geq 1$ | dimensionless | scaling factor with $f = f_{max}/s$ |
| $P_{dyn}(f)$ | Watt | dynamic power depending on $f$ |
| $P_{dyn}(s)$ | Watt | dynamic power depending on $s$ |
| $P_{stat}(f)$ | Watt | static power depending on $f$ |
| $P_{static}$ | Watt | static power for unscaled case $s = 1$ |
| $T(1)$ | seconds | unscaled sequential execution time, $s = 1$ |
| $T(s)$ | seconds | scaled sequential execution time $T(s) = s \cdot T(1)$ |
| $E(s)$ | Joule | energy for scaling factor $s$ |
| $E(p, s)$ | Joule | energy for parallel execution |
| $s_{opt}(p)$ | dimensionless | optimal scaling factor for parallel execution |
| $P_{dyn}(p, 1)$ | Watt | dynamic power, unscaled parallel execution |
| $P_{static}(p, 1)$ | Watt | static power, unscaled parallel execution |
| $T(p, 1)$ | seconds | parallel execution for unscaled case |
| $EDP(p, s)$ | Joule $\cdot$ seconds | energy-delay product EDP parallel and scaled |
| $s_{opt}^{EDP}(p)$ | dimensionless | optimal scaling factor of EDP for parallel execution |

Table 2: Comparison of the frequency scaling factor $s$ and the performance factor $\tilde{s} = T(s)/T(1)$ for selected SPLASH-2 benchmarks on Haswell.

| **frequencies** | | **barnes (p=1)** | | | **cholesky (p=1)** | | | **raytrace (p=8)** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| GHz | $s = f_{max}/f$ | $run[sec]$ | $\tilde{s}$ | $Dev(\tilde{s}, s)$ | $run[sec]$ | $\tilde{s}$ | $Dev(\tilde{s}, s)$ | $run[sec]$ | $\tilde{s}$ | $Dev(\tilde{s}, s)$ |
| 0.8 | **4.25** | 442.73 | **4.12** | 3.06% | 1.53 | **4.21** | 0.88% | 96.92 | **4.24** | 0.25% |
| 1.0 | **3.40** | 354.31 | **3.30** | 3.02% | 1.26 | **3.47** | 1.98% | 77.07 | **3.37** | 0.86% |
| 1.2 | **2.83** | 295.11 | **2.75** | 3.08% | 1.04 | **2.88** | 1.56% | 64.71 | **2.83** | 0.11% |
| 1.4 | **2.43** | 253.93 | **2.36** | 2.70% | 0.89 | **2.47** | 1.55% | 55.33 | **2.43** | 0.36% |
| 1.5 | **2.27** | 237.15 | **2.21** | 2.64% | 0.84 | **2.30** | 1.63% | 52.14 | **2.28** | 0.62% |
| 1.7 | **2.00** | 210.57 | **1.96** | 2.02% | 0.73 | **2.03** | 1.29% | 45.47 | **1.99** | 0.55% |
| 1.9 | **1.79** | 187.67 | **1.75** | 2.42% | 0.66 | **1.82** | 1.97% | 40.80 | **1.78** | 0.26% |
| 2.1 | **1.62** | 170.23 | **1.58** | 2.16% | 0.60 | **1.66** | 2.71% | 36.85 | **1.61** | 0.45% |
| 2.3 | **1.48** | 157.11 | **1.46** | 1.10% | 0.55 | **1.52** | 3.15% | 33.54 | **1.47** | 0.74% |
| 2.5 | **1.36** | 143.42 | **1.33** | 1.87% | 0.51 | **1.41** | 3.50% | 30.85 | **1.35** | 0.78% |
| 2.7 | **1.26** | 132.84 | **1.24** | 1.83% | 0.46 | **1.27** | 0.53% | 28.70 | **1.26** | 0.28% |
| 2.8 | **1.21** | 128.99 | **1.20** | 1.15% | 0.46 | **1.26** | 3.46% | 27.76 | **1.21** | 0.01% |
| 3.0 | **1.13** | 120.52 | **1.12** | 1.04% | 0.43 | **1.17** | 3.55% | 26.12 | **1.14** | 0.81% |
| 3.2 | **1.06** | 114.09 | **1.06** | 0.07% | 0.40 | **1.11** | 4.40% | 24.54 | **1.07** | 1.02% |
| 3.4 | **1.00** | 107.46 | **1.00** | 0.0% | 0.36 | **1.00** | 0.0% | 22.86 | **1.00** | 0.0% |

$$P_{dyn}(f) = \alpha \cdot C_L \cdot V^2 \cdot f,$$

where $\alpha$ is the switching probability, $C_L$ is the load capacitance, and $V$ is the supply voltage. The frequency $f$ depends linearly on the supply voltage $V$, which means that $V = \beta \cdot f$ with some appropriate constant $\beta$. Thus, the dependence of the dynamic power consumption on the frequency $f$ can be expressed as $P_{dyn}(f) = \gamma \cdot f^3$ with $\gamma = \alpha \cdot C_L \cdot \beta^2$ or when using the corresponding scaling factor $s$ as

$$P_{dyn}(s) = s^{-3} \cdot P_{dyn}(1) \tag{1}$$

where $P_{dyn}(1) = \gamma \cdot f_{max}^3$ is the dynamic power consumption in the unscaled case.

The static power consumption $P_{stat}(f)$ is intended to capture the leakage power consumption and can be expressed by $P_{stat}(f) = V \cdot N \cdot k_{\text{design}} \cdot I_{\text{leak}}$, where $N$ is the number of transistors, $k_{\text{design}}$ is a design dependent parameter, and $I_{\text{leak}}$ is a technology-dependent parameter [7]. Using $V = \beta \cdot f$ again leads to a linear dependence of the static power on $f$, i.e., $P_{stat}(f) = \delta \cdot f$ with $\delta = N \cdot k_{\text{design}} \cdot I_{\text{leak}} \cdot \beta$ or $P_{stat}(s) = s^{-1} \cdot P_{stat}(1)$ where $P_{stat}(1)$ is the static power consumption in the unscaled case. Other authors have also proposed to make the simplified assumption that $P_{static}$ is independent of the voltage or frequency scaling [42], which means that

$$P_{stat}(s) = P_{stat}(1) = P_{static}. \tag{2}$$

In the following, we use Equ.(2). The total power consumption adds up both power components:

$$P_{total}(s) = s^{-3} P_{dyn}(1) + P_{static}. \tag{3}$$

Reducing the operational frequency of a processor by a scaling factor of $s$ usually decreases the power consumption. However, it increases the execution time $T(1)$ of an application program by about the same factor compared to an unscaled execution. This has been confirmed by runtime measurements on DVFS architectures [32]. As example, Table 2 shows the performance scaling factors $\tilde{s} = T(s)/T(1)$ and the frequency scaling factors $s = f_{max}/f$ for three selected SPLASH-2 benchmarks on the Haswell processor along with the deviation between $s$ and $\tilde{s}$, denoted as $Dev(\tilde{s}, s)$. It can be observed that the deviations between the two factors are quite small. Similar deviations can be observed for other PARSEC or SPLASH-2 benchmarks and other processors. There is no significant difference of the deviations between a sequential execution and a parallel execution. For the energy model, we therefore assume $s = \tilde{s}$ and $T(s) = s \cdot T(1)$. This leads to a simple energy model that facilitates an analytical treatment.

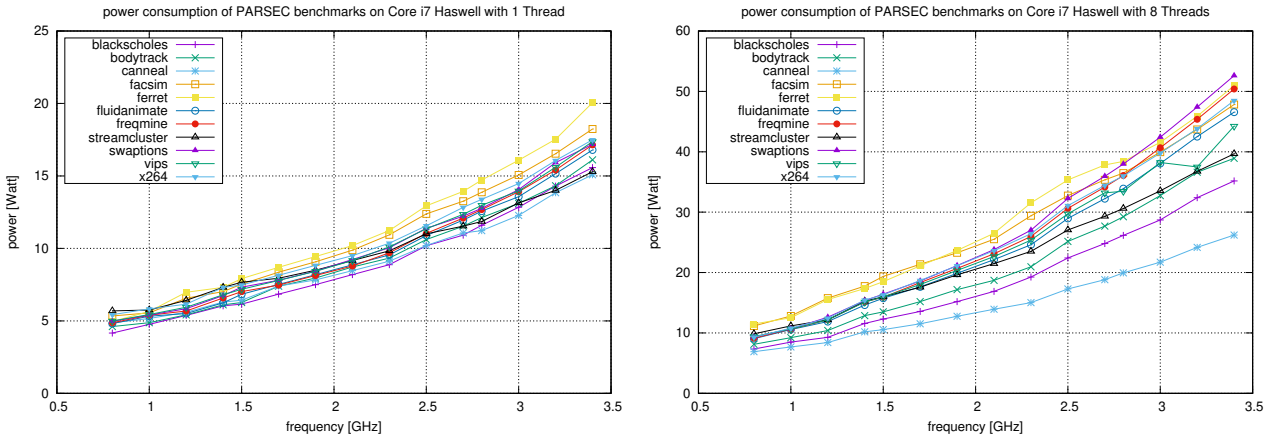


Figure 1: Power consumption of the PARSEC benchmarks executed with one thread (top) and eight threads (bottom) on an Intel Core i7 Haswell architecture.

Using the power and the execution time depending on the scaling factor used for the execution of the application program as given above leads to the following energy model for $E(s)$ depending on the scaling factor $s$:

$$\begin{aligned} E(s) &= (P_{dyn}(s) + P_{static}) \cdot T(s) \\ &= (s^{-3} \cdot P_{dyn}(1) + P_{static}) \cdot s \cdot T(1) \\ &= (s^{-2} \cdot P_{dyn}(1) + s \cdot P_{static}) \cdot T(1), \end{aligned} \tag{4}$$

which describes the energy consumption as a product of the execution time in the unscaled case and the power consumption, containing the values for the dynamic power consumption in the unscaled case, the static power consumption and the scaling parameter.

## 2.2 Energy model for parallel execution

The execution of multi-threaded programs introduces a further variable into the model which is the number $p$ of threads used for a specific code execution. The execution time usually decreases with an increasing number of threads (typically in a non-linear way) until a saturation point is reached, which strongly depends on the application. On the other hand, also the power drawing varies with the number of threads, usually in a way that the power drawing increases with an increasing number of threads. Taking the dependence of the execution time and power drawing on the number of threads and the frequency scaling into consideration results in an energy model with two parameters. More precisely, Equ. (4) is extended to take varying numbers $p$ of executing threads into consideration in addition to the scaling factor $s$, yielding

$$E(p, s) = (s^{-2} \cdot P_{dyn}(p, 1) + s \cdot P_{stat}(p, 1)) \cdot T_{par}(p, 1) \tag{5}$$

where $T_{par}(p, 1)$ denotes the parallel execution time with $p$ threads for scaling factor $s = 1$ and $P_{dyn}(p, 1)$ and $P_{stat}(p, 1)$ denote the dynamic and the static power consumption when using $p$ threads.

The energy model (5) is a continuously differentiable function in $s$, which makes it possible to analytically derive an optimal value for the scaling factor $s$, when $p$ is set to a fixed value. The optimal scaling factor $s_{opt}(p)$ which minimizes $E(p, s)$ can be determined by building the derivative of $E(p, s)$ and looking for the root of the derivative. The derivative of $E(p, s)$ is

$$\frac{\partial E(p, s)}{\partial s} = \left(-2s^{-3} P_{dyn}(p, 1) + P_{static}(p, 1)\right) \cdot T_{par}(p, 1).$$

The function for $E(p, s)$ in Equ. (5) is convex, since the second derivative $\partial^2 E(p, s)/\partial^2 s$ exists and $\partial^2 E(p, s)/\partial^2 s \geq 0$. Thus, the optimal scaling factor can be obtained by setting $\partial E(p, s)/\partial s$ to zero. The resulting optimum scaling

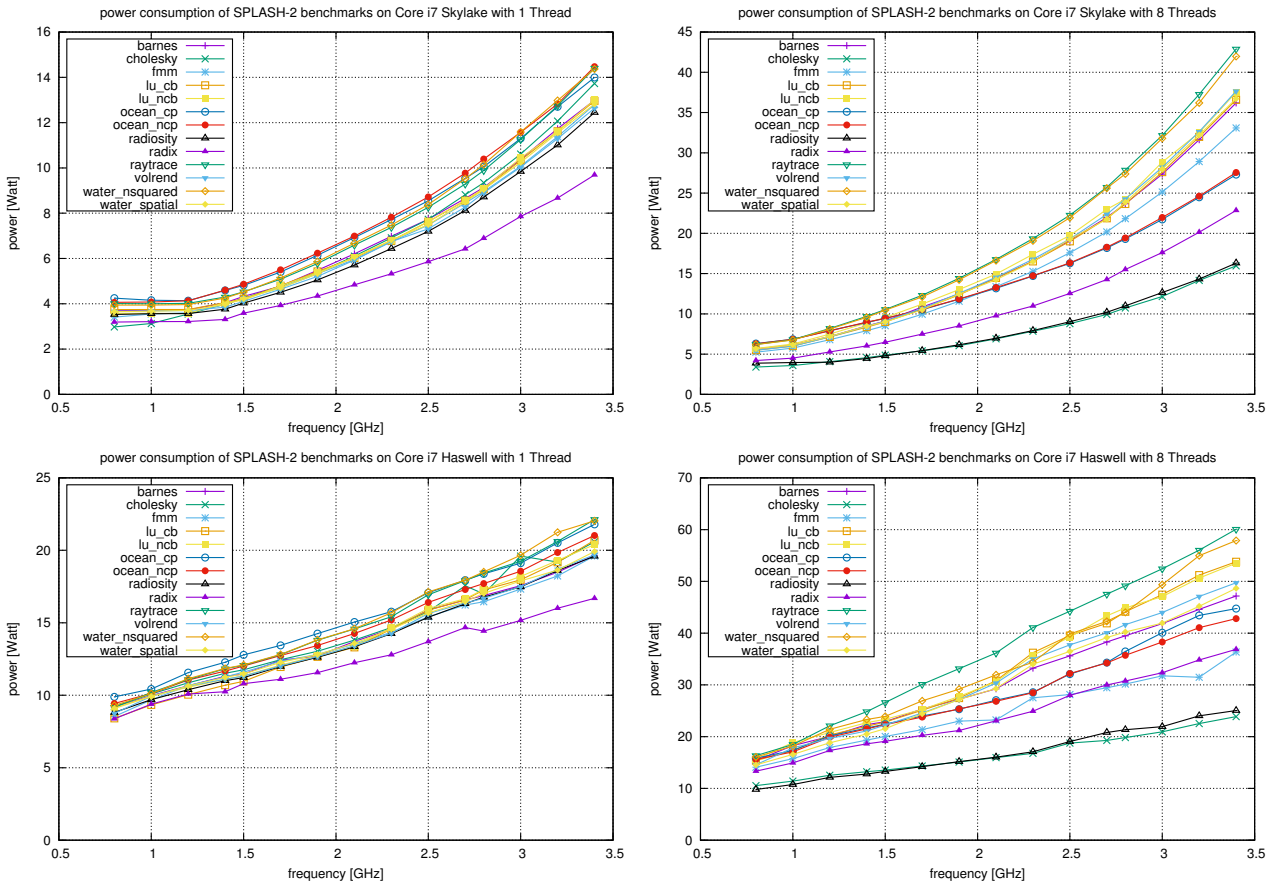


Figure 2: Power consumption of the SPLASH-2 benchmarks executed with one thread (left) and eight threads (right) on an Intel Core i7 Skylake (top) and Haswell (bottom) architecture using different frequencies.

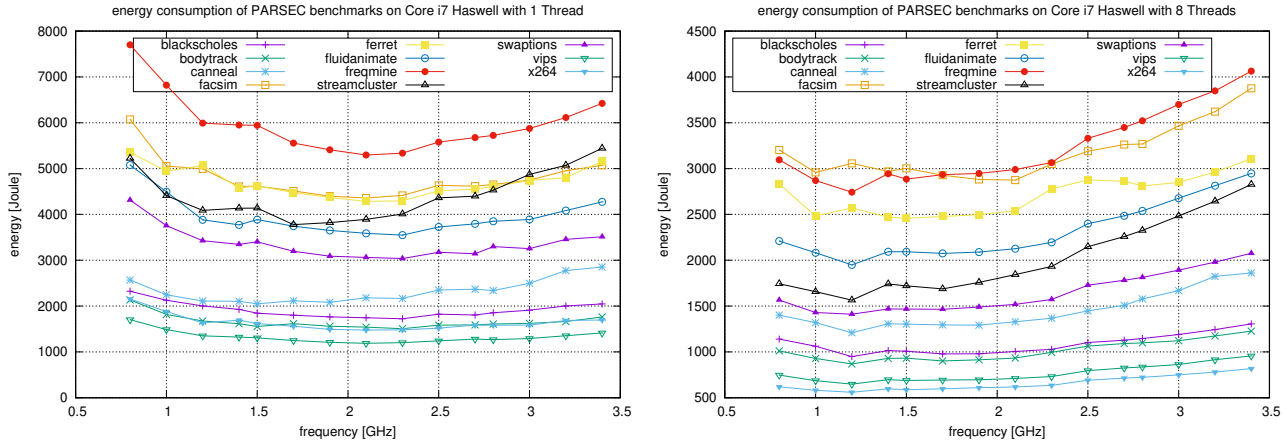


Figure 3: Energy consumption of the PARSEC benchmarks corresponding to Fig. 1.

factor is

$$s_{opt}(p) = \left( \frac{2 \cdot P_{dyn}(p,1)}{P_{static}(p,1)} \right)^{1/3}, \tag{6}$$

assuming that this scaling factor is kept fixed during the execution of the application program. According to Equ. (6), the value of $s_{opt}(p)$ is independent of the actual execution time of the application considered and at first glance seems to be fixed. However, different applications may have different values $P_{dyn}(p,1)$ and $P_{static}(p,1)$ for a fixed $p$ due to a different usage of the hardware resources of the processor employed. Moreover, using a different number of threads may lead to different values for $P_{dyn}(p,1)$ and $P_{static}(p,1)$ due to the exploitation of parallelism. Thus, different applications may have different values for $s_{opt}(p)$ when $p$ is fixed and different values for $s_{opt}(p)$ may result for different values of $p$, even for the same application.

### 2.3 Energy model for the energy-delay product

In this subsection, we extend the parallel energy model from the previous subsection to the energy-delay product (EDP). The EDP is defined as the energy consumed by an application program multiplied by its execution time [34]. Since the energy consumption also contains the execution time as multiplicative factor, see Equ. (5), the EDP is actually the product of the power consumption and the square of the execution time. When considering the scaling factor $s$ explicitly, the EDP can be expressed as

$$EDP(p,s) = E(p,s) \cdot (T(p,1) \cdot s) \qquad [Watt \cdot s^2] \tag{7}$$

$$= (s^{-1} \cdot P_{dyn}(p,1) + s^2 \cdot P_{static}(p)) \cdot T^2(p,1) \tag{8}$$

The EDP is a metric that combines effects of execution time and energy consumption and captures the translation of energy into useful work. Using the derivative

$$\frac{\partial EDP(p,s)}{\partial s} = \left( -s^{-2} P_{dyn}(p,1) + 2s \cdot P_{static}(p,1) \right) \cdot T(p,1)^2$$

and applying the same approach as for the energy consumption yields the optimum scaling factor

$$s_{opt}^{EDP}(p) = \left( \frac{P_{dyn}(p,1)}{2 \cdot P_{static}(p,1)} \right)^{1/3} \tag{9}$$

for the EDP for a given $p$. The optimum scaling factors from Equs. (6) and (9) have the property that $s_{opt}^{EDP}(p) < s_{opt}(p)$, since $(1/2)^{1/3} < 2^{1/3}$. This means that the scaling factor minimizing the EDP is smaller than the scaling factor minimizing the energy consumption. In particular, it is $1.58 \cdot s_{opt}^{EDP}(p) \approx 4^{1/3} \cdot s_{opt}^{EDP}(p) = s_{opt}(p)$.

The significance of the energy delay product can be seen when looking at the energy efficiency (EE). The energy efficiency of an application is defined as performance (flop/s) per energy unit ($Ws$) and is measured as $flop/Ws^2$. The reciprocal value $1/EDP_1$ has the unit $1/Ws^2$. Given two EDP values $EDP_1$ and $EDP_2$ with $EDP_1 < EDP_2$ yields $1/EDP_1 > 1/EDP_2$, both sides measured in $1/Ws^2$. This means that a smaller EDP value $EDP_1$ indicates a better energy efficiency, i.e., a larger performance per energy unit. Therefore, the EDP is a good measure for the energy efficiency of an application, and $s_{opt}^{EDP}$ optimizes the energy efficiency of an application program for a given execution platform.

## 3 Experimental environment

In this section, we describe the experimental setting used for the experimental evaluation and the benchmarks that are investigated.

Table 3: Characteristics of the Intel processors used for the experimental evaluation.

| | **Haswell** | **Skylake** | **Haswell-EP** | **Broadwell-E** |
|---|---|---|---|---|
| Model | i7-6700 | i7-4770 | 2 × E5-2630 v3 | i7-6950X |
| Cores | 4 (8) | 4 (8) | 2 × 8 (16) | 10 (20) |
| min. fequ. | 0.8 GHz | 0.8 GHz | 1.2 GHz | 1.2 GHz |
| max. fequ. | 3.4 GHz | 3.4 GHz | 2.4 GHz | 3.0 GHz |
| step size | 100/200 MHz | 100/200 MHz | 100 MHz | 100/200 MHz |
| Hyperthreading | Yes | Yes | Yes | Yes |
| TPD | 84 W | 65 W | 85 W | 140 W |
| L1d Cache | 32 KB | 32 KB | 32 KB | 64 KB |
| L1i Cache | 32 KB | 32 KB | 32 KB | 64 KB |
| L2 Cache | 256 KB | 256 KB | 256 KB | 256 KB |
| L3 Cache | 8 MB | 8 MB | 20 MB | 25 MB |
| RAM | 16 GB | 16 GB | 64 GB | 32 GB |
| OS | Suse Leap 42.2 | Suse Leap 42.2 | SLES 13 | Ubuntu 16.04 |
| Drive | SSD | SSD | HDD | SSD |

## 3.1 The PARSEC and SPLASH-2 Benchmark Suite

The SPLASH-2 benchmark suite is a mature but still very popular benchmark suite released in 1995 and mainly comprises applications from scientific computing and graphics, including Cholesky and LU factorization or ray-tracing and radiosity algorithms. Due to the architectures at that time, the codes have been optimized for multi-nodes with high latencies between nodes, so that communication between them had been avoided when possible.

The PARSEC Benchmark Suite is a collection of benchmarks aiming at the investigation of thread parallelism on recent chip multiprocessors. The PARSEC benchmarks emphasize on larger workloads typical for today's application programs. Also, while the SPLASH-2 benchmarks primarily come from scientific computing, the PARSEC benchmarks comprise a wider range of applications, including financial analysis, animation, data mining, or enterprise storage. The computational characteristics of the PARSEC benchmarks are intensively studied in [6, 5] where properties concerning the instructions and shared data are investigated.

The experiments described in the following have been performed with the `native` input sets of the PARSEC and SPLASH-2 benchmarks, which are the largest input sets available and which are intended for performance measurements on real machines [6].

## 3.2 Experimental setting

The experimental evaluation has been performed on four Intel multicore processors with different architectures, including three desktop processors (Haswell, Skylake, Broadwell-E) and a server processor (Haswell-EP). Information about the processors used are given in Table 3. All these processors support DVFS. The usage of Intel Turbo Boost has been disabled on all systems due to the usage of cpufreq-set to set the DVFS frequencies. The Haswell-EP is a server system containing two Intel Xeon E5-2630 v3 processors, each providing eight physical cores and supporting hyperthreading. This system has a smaller range of frequency selection than the desktop processors. The three desktop Intel Core i7 systems used have the operating system as well as the benchmarks installed on an solid state drive (SSD). Therefore, network and writing limitations are negligible. The Haswell-EP uses a normal hard disk drive (HDD), so the access times are expected to be longer. For the experimental evaluation, the systems were used as stand-alone system with no other users on the system. The number of processes were reduced as far as possible to reduce influences on the measurements as much as possible.

The time and energy measurements have been performed by using the Running Average Power Limit (RAPL) interface and sensors of the Intel architecture [35, 21]. RAPL sensors can be accessed by control registers, known as Model Specific Registers (MSRs), which are updated in intervals of about 1 $ms$ [21]. In particular, we have used the likwid tool-set, especially the likwid-powermeter (Version 4.0) [39], which provides access to the MSRs from user space using the Linux MSR module. The likwid tool provides pre-configured event sets with selected performance counters and derived metrics. To unlock the MSR registers, `modprobe msr` has been used. All

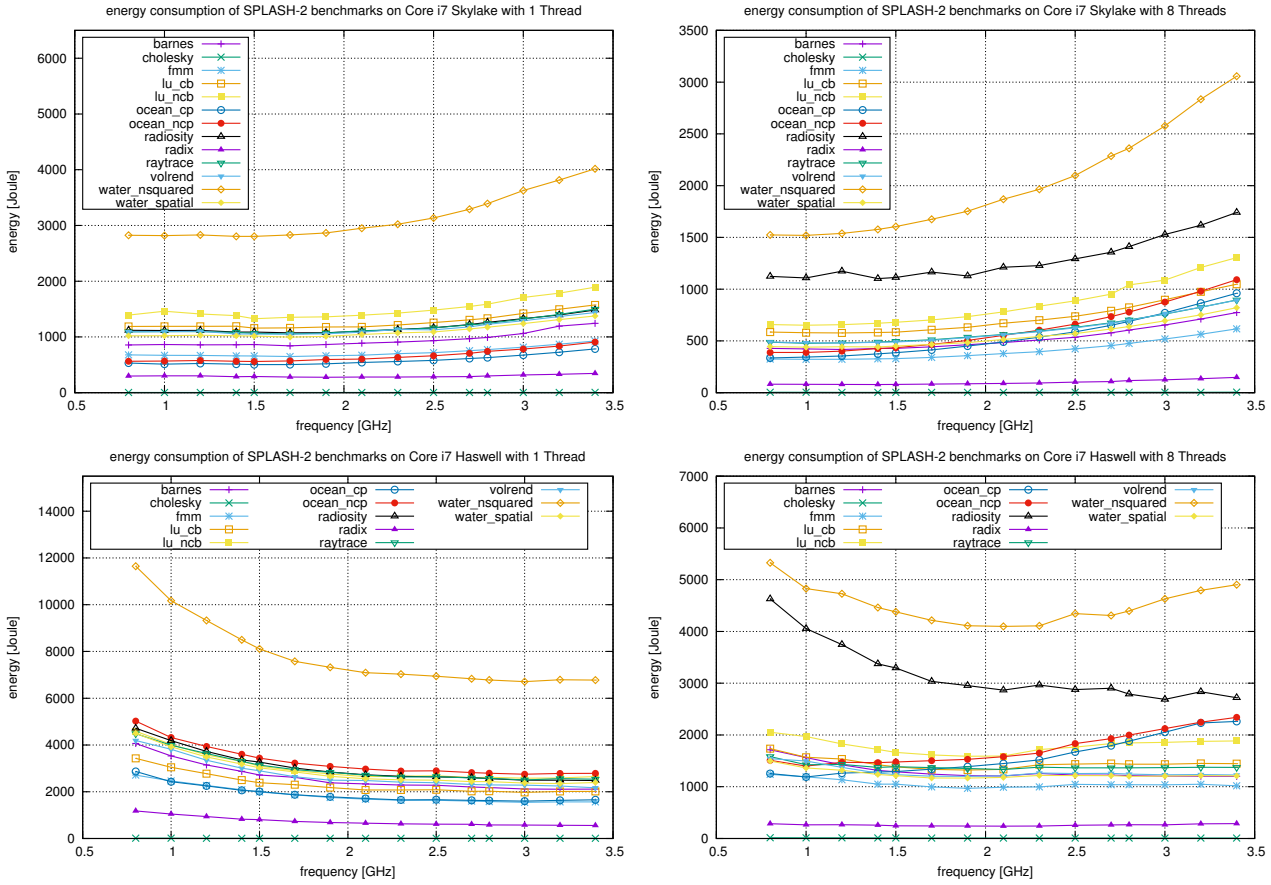


Figure 4: Energy consumption of the SPLASH-2 benchmarks executed with one thread (left) and eight threads (right) on an Intel Core i7 Skylake (top) and Haswell (bottom) architecture using different frequencies.

measurements have been performed as end-to-end measurements of the entire application selected. The energy measurements have been performed for the package power plane. An exception is the Broadwell-E, where only the core power plane could be accessed, capturing only the energy of the CPU cores. The accuracy of the RAPL measurements has been shown in [35].

# 4 Power and energy measurements

The power and energy consumption for executing application programs depends both on the number of threads used and the operational frequency. The diagrams and tables presented in the following mainly show the results for one and eight threads. The dependence on a larger range of the number of threads is investigated in Section 4.4.

## 4.1 Measured power consumption

Table 4 gives the power consumption values for the freqmine program from the PARSEC benchmark suite as example, which shows that the power draw increases with the frequency for a fixed number of threads. For a fixed frequency, the power drawing increases with the number of threads. Figure 1 shows the power consumption of all PARSEC applications on the Haswell architecture for $p = 1$ (top) and $p = 8$ (bottom). Figure 2 shows the power consumption of all SPLASH-2 applications on the Haswell and Skylake architectures for $p = 1$ and $p = 8$. Both figures show that all application programs have the same qualitative power consumption behavior: The power

Table 4: Power Consumption in Watt of the PARSEC benchmark freqmine on Skylake for different frequencies and different numbers of threads

| | **power consumption** | | | |
|---|---|---|---|---|
| **Freq** | $p = 1$ | $p = 2$ | $p = 4$ | $p = 8$ |
| 0.8 GHz | 3.73 | 3.59 | 4.56 | 5.57 |
| 1.0 GHz | 3.69 | 3.60 | 5.71 | 6.10 |
| 1.2 GHz | 3.70 | 4.25 | 5.91 | 7.34 |
| 1.4 GHz | 4.04 | 4.95 | 7.96 | 8.69 |
| 1.5 GHz | 4.33 | 5.28 | 8.48 | 9.38 |
| 1.7 GHz | 4.83 | 5.99 | 10.21 | 11.06 |
| 1.9 GHz | 5.43 | 6.94 | 11.94 | 12.96 |
| 2.1 GHz | 5.91 | 8.03 | 13.78 | 15.05 |
| 2.3 GHz | 6.65 | 9.23 | 13.51 | 17.42 |
| 2.5 GHz | 7.49 | 10.59 | 18.40 | 20.01 |
| 2.7 GHz | 8.42 | 12.31 | 17.72 | 23.20 |
| 2.8 GHz | 9.03 | 13.18 | 19.20 | 25.18 |
| 3.0 GHz | 10.42 | 15.21 | 22.20 | 29.22 |
| 3.2 GHz | 11.74 | 11.93 | 30.95 | 33.64 |
| 3.4 GHz | 13.00 | 13.26 | 28.63 | 38.49 |

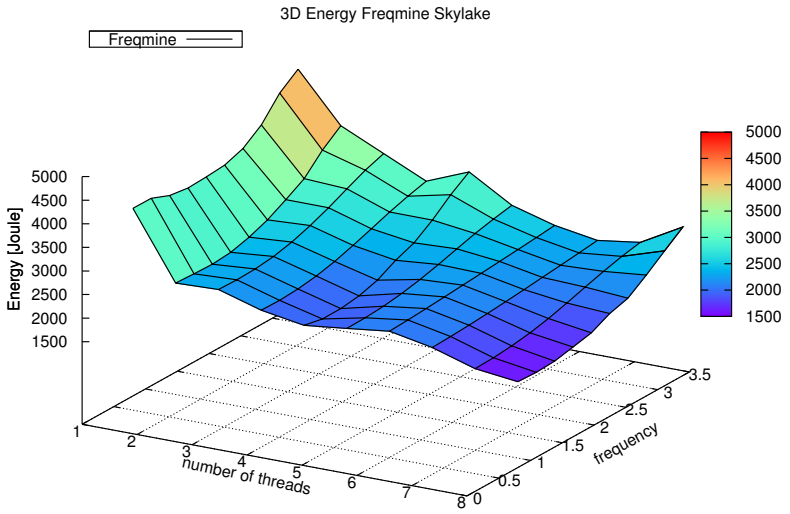


Figure 5: Energy surface plot for the PARSEC benchmark freqmine with different frequencies and different numbers of threads on an Intel Core i7 Skylake.

consumption increases with the operational frequency and there is also an increase of the power consumption with the number of threads. However, it can also be observed that the quantitative amount of power consumption strongly varies between the different applications. For the highest frequency of 3.4 GHz, the following intervals of power consumption can be observed from Figure 1 for the PARSEC benchmarks: for $p = 1$ roughly between 15 and 24 Watt and for $p = 8$ roughly between 20 and 52 Watt. For the SPLASH-2 benchmarks, the following power consumption intervals can be seen from Figure 2 for 3.4 GHz:

- Skylake (upper left): for $p = 1$: between 10 and 15 Watt
- Haswell (lower left): for $p = 1$: between 16 and 23 Watt
- Skylake (upper right) for $p = 8$: between 15 and 43 Watt
- Haswell (lower right) for $p = 8$: between 22 and 60 Watt.

Interestingly, different applications change their power consumption behavior in different quantitative ways: E.g. the PARSEC benchmark `facesim` has a slightly lower power consumption on eight threads than on one thread with the effect that `facesim` has the highest power consumption on one thread compared to the other applications but has the lowest power consumption on eight threads compared to the other applications, see Figure 1. Another example is the PARSEC benchmark `ferret`, which has a higher power consumption on the Haswell architecture than almost every other benchmark when using only one thread. It can also be observed that there is a larger variation between the power consumption of the different applications when the number of threads is increased, see Figures 1 and 2.

## 4.2 Measured energy consumption

Figures 3 and 4 present the corresponding energy consumption of the application programs from Fig. 1 and 2. The diagrams show that for most of the programs, there is an operational frequency for which the energy consumption is at a minimum. The U-shape of the energy curves is especially present for the Haswell processor. The energy curves confirm the energy model described in Subsection 2.1, which predicts such a frequency with minimum energy consumption in Formula (6).

Figure 5 shows the energy consumption surface depicting how the energy consumption depends on the use of multi-threading and frequency scaling for the PARSEC program freqmine. Most other applications show a similar qualitative behavior.

## 4.3 Measured EDP values

The EDP combines the qualitative effects of power and energy, which results in decreasing EDP values for increasing frequency. As an example, Table 5 gives the EDP for the PARSEC benchmark freqmine on Skylake as example. The minimum EDP values for the different number of threads are highlighted. It can be seen that the operational frequency for the minimum EDP value decreases with the number of threads employed. All other applications show a similar behavior.

## 4.4 Dependence on the number of threads

The energy model presented in Subsection 2.1 mainly emphasizes on the minimization of the energy by choosing the appropriate frequency $f_{max}/s_{opt}$. In Subsection 2.2 we have extended the energy model by including the number of threads as an independent variable. However, so far, the number of threads is set to a fixed value so that a family of curves, each curve for a fixed number of threads having its own minimum, is considered. We now consider the dependence on the number of threads for a fixed frequency value.

Figure 6 (left) shows the dependence of the power consumption (top), the execution time (middle) and the energy consumption (bottom) of the different PARSEC programs on the number of threads for the maximum operational frequency $f$ =3.4 GHz. The Skylake processor has been used. Figure 6 (right) shows the same information for the Haswell-EP processor for the maximum operational frequency $f$ =2.4 GHz. The following observations can be made for most of the benchmarks:

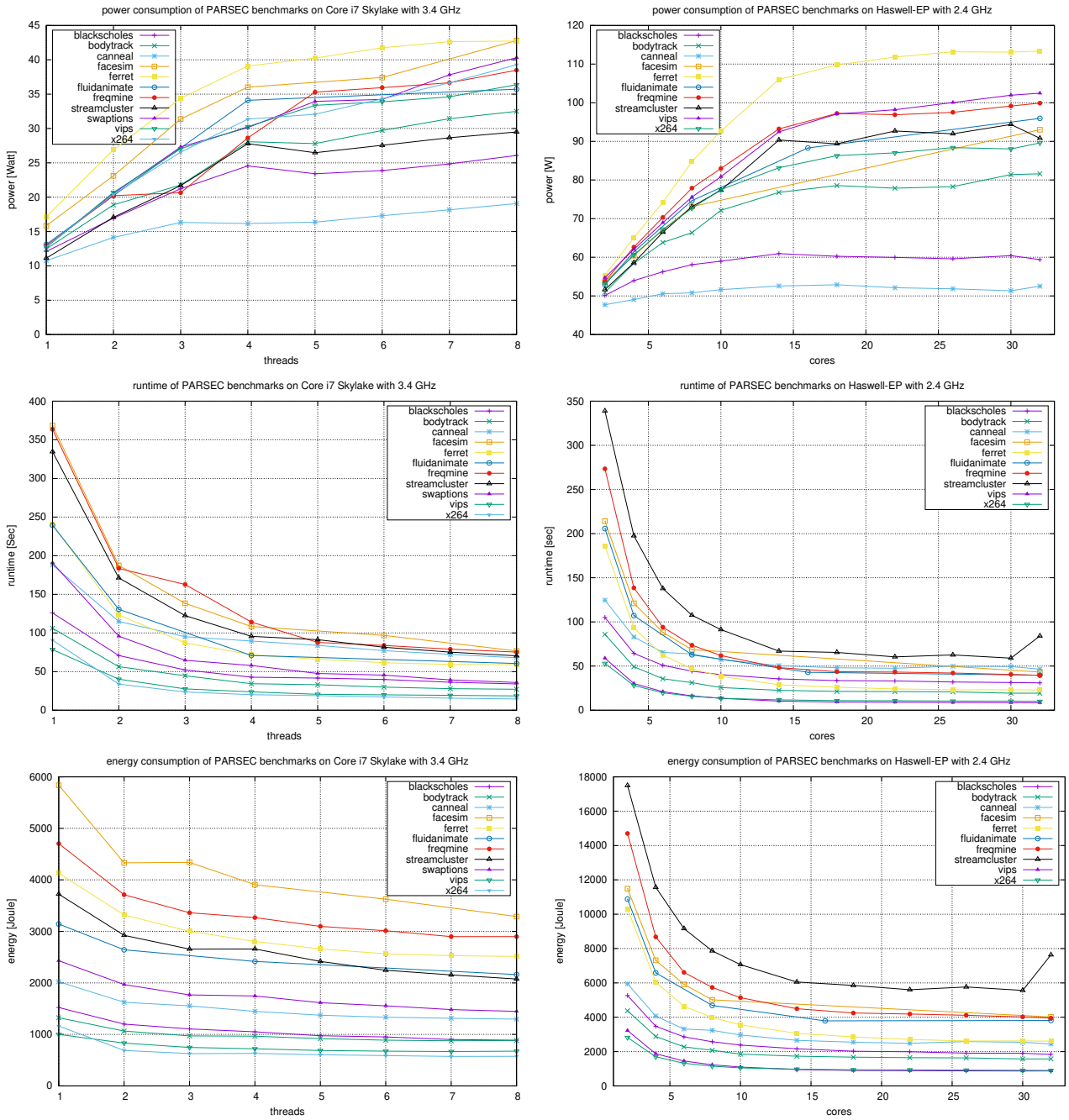


Figure 6: Dependence of the power consumption (top), execution time (middle) and energy consumption (bottom) on the number of threads for the PARSEC benchmarks on Skylake (left) and Haswell-EP (right).

Table 5: EDP for the freqmine benchmark on Skylake.

| | energy-delay product | | | |
|---|---|---|---|---|
| **Freq** | $p = 1$ | $p = 2$ | $p = 4$ | $p = 8$ |
| 0.8 GHz | 3723095.16 | 1373633.25 | 824922.42 | 441310.04 |
| 1.0 GHz | 3777602.27 | 1371858.56 | 557796.35 | 390715.00 |
| 1.2 GHz | 3757455.40 | 1128545.10 | 604685.25 | 331431.44 |
| 1.4 GHz | 3144239.67 | 966765.90 | 417609.50 | 284399.55 |
| 1.5 GHz | 2918132.72 | 897168.38 | 400445.90 | 268099.84 |
| 1.7 GHz | 2548585.34 | 799809.38 | 350197.07 | 247316.78 |
| 1.9 GHz | 2295347.88 | 740150.73 | 329960.00 | 231725.31 |
| 2.1 GHz | 2028699.78 | 698448.61 | 311274.87 | 221811.76 |
| 2.3 GHz | 1909394.77 | 669977.56 | 377899.42 | 214698.15 |
| 2.5 GHz | 1821256.42 | 653056.97 | **293312.49** | **207365.11** |
| 2.7 GHz | 1755470.91 | 649779.96 | 360036.75 | 208173.11 |
| 2.8 GHz | 1751893.85 | **647770.16** | 364071.61 | 212228.70 |
| 3.0 GHz | 1764447.10 | 650671.30 | 369381.94 | 211446.83 |
| 3.2 GHz | 1751533.01 | 1143415.82 | 300938.57 | 213749.20 |
| 3.4 GHz | **1717953.01** | 1115712.35 | 372650.08 | 218278.50 |

- The power consumption increases with the number of threads. However, different applications show quite different increase rates, which vary from a slight increase to a sharp increase. A sharp power increase may be caused by an intensive usage of the physical cores by the additional threads, whereas a slight increase indicates a less intensive usage. The power increase effect is especially significant for up to four threads on the Skylake and up to 16 threads on the Haswell-EP, which corresponds to the number of physical cores of these processors. For more than four or 16 threads, respectively, the power increase slows down and for some of the benchmarks, it even drops slightly.

- The execution time decreases with the number of threads, which can be expected for a parallel execution. The speedup for the PARSEC benchmarks on the Haswell and Skylake architecture using eight threads on four cores is about 4 or larger for many of the benchmarks, see Table 7. This is also true for the SPLASH benchmarks, see Table 8. Thus, most of the applications show a quite good scalability behavior.

- The energy consumption decreases when the number of threads is increased, especially as long as the number of threads is smaller than the number of physical cores available. Most of the applications show a quite similar decrease rate for the energy. For some of the benchmarks the energy consumption slightly increases again when increasing the number of threads beyond the number of physical cores.

A similar behavior can be observed for other frequencies than 3.4 GHz, not shown in a figure.

## 4.5 Summarizing data observations

The measurements performed show a diverse behavior for power, energy and performance with respect to varying frequency and number of threads. However, we can extract the following patterns of behavior:

- **power**: For all benchmarks, the power consumption increases with the operational frequency for each specific number of threads, see Figs. 1 and 2. All power curves show the same qualitative behavior, which is linear or slightly convex, however the quantitative power values vary between the different applications up to a factor of three.

- **energy**: The energy consumption diagrams exhibit a convex behavior, which has a U-shape for the Haswell processor and a slightly increasing shape for the Skylake processor with respect to the frequency, see Figs. 3 and 4. Again, the qualitative behavior is similar for all benchmarks, but the quantitative behavior can differ significantly.

- **parallelism and scalability**: The number of threads employed have an impact on power, energy, and EDP as follows: For each frequency, the power consumption increases for an increasing number of threads, however with different increase rates, see Table 4 for a specific benchmark and Fig. 6 and **??** (top) for all PARSEC benchmarks. Most applications have a good scalability behavior, see Fig. 6 and **??** (middle), which leads to a decreasing energy consumption with the number of threads despite the increasing power, see Fig. 6 and **??** (bottom) and Fig. 5 for a specific benchmark (freqmine).
- **processor architecture**: The measurements on the Haswell and Skylake processor show a similar qualitative behavior. However there are quantitative differences due to the more recent design of the Skylake processor. Especially, the Skylake has a smaller energy consumption in all cases, see Fig. 4, a lower power consumption with a higher increase rate with respect to the frequency, see Fig. 2, and a smaller execution time, not shown in a figure.
- **applications**: Although the PARSEC and the SPLASH-2 benchmarks exhibit a similar behavior in all aspects, it can be observed that the SPLASH-2 benchmarks have generally a slightly higher power consumption than the PARSEC benchmarks for all number of threads, which might be caused by a larger number of numerical computations, leading to a larger computational load.

# 5 Modeling and validation

In this section, we investigate whether the optimal frequencies derived with the analytical model from Section 2.2, see Equ. (6) for the energy consumption and the EDP, see Equ. (9), correspond to the optimal frequencies measured on different platforms using different numbers of threads. We analyze how well the energy and EDP models from Section 2 are able to capture the effects observed for the performance and energy measurements.

## 5.1 Modeling methodology

The application-specific power modeling has the purpose to provide application-specific values for the parameters $P_{dyn}(1)$ and $P_{static}$ from Equ. (3). This is done by the least-squares method and the power model from Equ. (3), for which the values for $P_{dyn}(1)$ and $P_{static}$ are calculated based on the measured data from Subsection 4.1. In the following, we pursue two approaches: The first approach is an application-dependent local modeling that derives an individual model for each of the PARSEC and SPLASH benchmarks. We consider how well the model is suitable to describe the observed energy behavior of the corresponding application. In particular, we consider how well the models are able to capture the optimal operational frequency that leads to a minimum energy consumption. The second approach is an application-independent global modeling based on an independent test set of applications that is used for determining the parameters $P_{dyn}(1)$ and $P_{static}$. For this modeling, we use the first three applications of the PARSEC suite (blackscholes, bodytrack, canneal) and the SPLASH-2 suite (barnes, cholesky, fmm) as test set. Experiments have shown that the usage of a different test set leads to quite similar results. Using the model derived with the test set, we investigate how well the corresponding model is able to describe the energy behavior of the remaining applications (verification set), whose measured power and energy consumption have not been taken into consideration for the derivation of the model, thus exploring the predictive quality of the energy model. Again, the emphasis lies on the question how well the optimal frequency derived with the model fit to the optimal frequencies obtained with the measurements.

The modeling is done for both the energy consumption and the EDP. Since different functions are used for the energy consumption and the EDP, different optimal frequencies result for both approaches. The optimal frequency computed for the energy consumption should be used, if the goal is a reduction of the overall energy consumption. If the runtime of the application should also be taken into consideration, the optimal frequency computed for the EDP should be used, since the EDP balances the execution time and the energy consumption of applications.

## 5.2 Determination of application-specific power parameters

For the SPLASH-2 benchmarks, the use of the least-squares method results in the power parameter values given in Table 6 for $p = 1$ and $p = 8$. The data in Table 6 show the following characteristics: For a specific processor, the

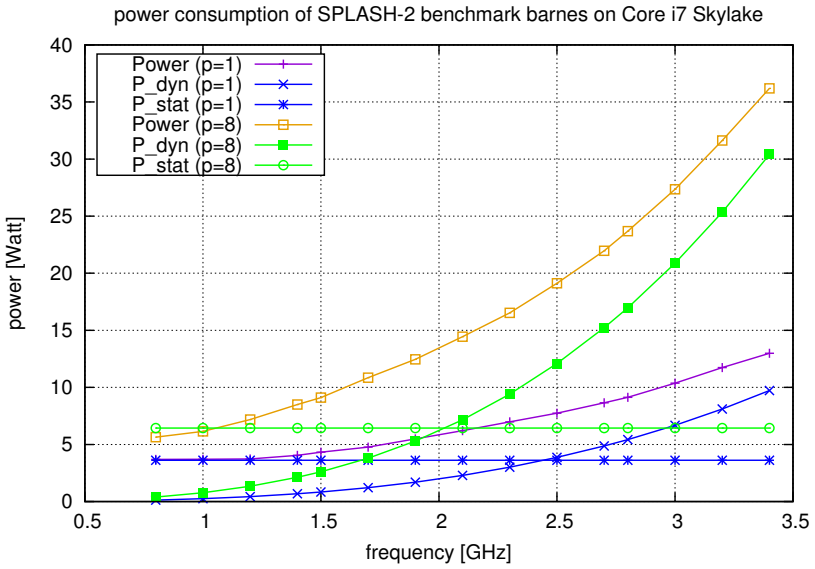


Figure 7: Measured and modeled power consumption of the SPLASH-2 benchmark barnes for one thread ($p = 1$) and eight threads ($p = 8$) on Skylake.

static power values $P_{static}$ are quite similar for all benchmarks on this processor. Comparing different processors, the static power values on the Skylake are much smaller than those on the Haswell processor. On the other hand, larger variations can be observed for the application-specific $P_{dyn}(1)$ values, which can be attributed to a different usage of the hardware resources by these application benchmarks. For each selected benchmark, the values for $P_{dyn}(1)$ are often quite similar for the Haswell and Skylake processors. Thus, the variations observed for the measured overall power consumption on the different processors have to be mainly caused by differences in the values of $P_{static}$.

For eight threads, there is a slightly larger variation in the static power consumption for the different benchmarks, but again the main difference is caused by the dynamic power consumption $P_{dyn}(1)$. As already observed in Fig. 1, there is a large dependence of $P_{dyn}(f)$ on the operational frequency. A dependence on the runtime speedup of the benchmarks can also be observed. More precisely, benchmarks with a large speedup have a larger dynamic power consumption than benchmarks with a small speedup. Nevertheless, there are significant variations also between benchmarks with a good scalability.

The general observations are illustrated by the example in Figure 7, which shows the power modeling for the SPLASH-2 benchmark barnes for varying operational frequencies using one thread ($p = 1$) and eight threads ($p = 8$) on the Skylake processor. The diagram exhibits the horizontal lines of $P_{static}$ with a parallel shift to a higher level for a higher number of threads ($p = 8$), the slightly increasing convex curve for $P_{dyn}(1)$ and the total power for $p = 1$ and the more strongly increasing curve for $P_{dyn}(1)$ and the total power for $p = 8$.

## 5.3 Application-specific modeling of optimal scaling factors

The values of $P_{dyn}(p, 1)$ and $P_{static}(p, 1)$ derived for each of the benchmarks are used to compute application-specific values for $s_{opt}(p)$. Table 7 shows the $s_{opt}(p)$ values for each of the PARSEC benchmarks on the Haswell (top) and the Skylake (bottom) architecture and summarizes measured and modeled performance and energy characteristics. The measured data are the minimum and maximum measured execution time (in seconds) resulting for different frequencies and different numbers of threads (columns 1 and 2), the runtime speedup obtained for the minimum ($f$=0.8 GHz) and maximum ($f$=3.4 GHz) frequency using different numbers of threads (columns 3 and 4), the minimum and maximum measured energy consumption (in Joule) over all frequencies and numbers of threads (columns 5 and 6), and the frequency for which the energy consumption is at a minimum for $p$=1 and $p$=8 threads (columns 7 and 8). The modeled data are the optimal scaling factors derived with Equ. (6) from Section 2.2

Table 6: Measured and modeled Power Consumption of the SPLASH-2 Benchmarks on Haswell (first half) and Skylake (second half)

| | | $Power$ | | $P_{dyn}(1)$ | | $P_{static}$ | |
|---|---|---|---|---|---|---|---|
| **Benchmark** | **Freq** | **p=1** | **p=8** | **p=1** | **p=8** | **p=1** | **p=8** |
| barnes | 0.8 | 9.18 | 15.40 | 0.13 | 0.40 | 10.66 | 20.30 |
| cholesky | 0.8 | 9.21 | 10.53 | 0.15 | 0.17 | 10.69 | 12.28 |
| fmm | 0.8 | 8.54 | 14.09 | 0.13 | 0.26 | 10.49 | 17.96 |
| lu_cb | 0.8 | 8.40 | 15.38 | 0.15 | 0.51 | 9.99 | 19.81 |
| lu_ncb | 0.8 | 9.14 | 15.62 | 0.14 | 0.50 | 10.42 | 20.38 |
| ocean_cp | 0.8 | 9.89 | 15.84 | 0.15 | 0.37 | 11.52 | 19.19 |
| ocean_ncp | 0.8 | 9.43 | 15.62 | 0.15 | 0.35 | 10.94 | 19.33 |
| radiosity | 0.8 | 8.82 | 9.80 | 0.14 | 0.20 | 10.31 | 11.82 |
| radix | 0.8 | 8.38 | 13.33 | 0.10 | 0.30 | 9.87 | 16.69 |
| raytrace | 0.8 | 9.16 | 16.33 | 0.16 | 0.56 | 10.93 | 22.77 |
| volrend | 0.8 | 8.75 | 14.57 | 0.13 | 0.45 | 10.41 | 19.73 |
| water_nsquared | 0.8 | 9.29 | 16.07 | 0.16 | 0.54 | 10.96 | 20.26 |
| water_spatial | 0.8 | 9.09 | 14.55 | 0.14 | 0.44 | 10.55 | 19.21 |
| barnes | 3.4 | 19.66 | 47.17 | 10.08 | 30.82 | 10.66 | 20.30 |
| cholesky | 3.4 | 20.59 | 23.86 | 11.19 | 12.70 | 10.69 | 12.28 |
| fmm | 3.4 | 19.61 | 36.36 | 9.99 | 19.74 | 10.49 | 17.96 |
| lu_cb | 3.4 | 20.69 | 53.79 | 11.74 | 39.09 | 9.99 | 19.81 |
| lu_ncb | 3.4 | 20.39 | 53.52 | 11.13 | 38.17 | 10.42 | 20.38 |
| ocean_cp | 3.4 | 21.79 | 44.74 | 11.31 | 28.71 | 11.52 | 19.19 |
| ocean_ncp | 3.4 | 21.02 | 42.80 | 11.17 | 26.64 | 10.94 | 19.33 |
| radiosity | 3.4 | 19.59 | 25.05 | 10.42 | 15.01 | 10.31 | 11.82 |
| radix | 3.4 | 16.69 | 36.86 | 7.73 | 22.69 | 9.87 | 16.69 |
| raytrace | 3.4 | 22.12 | 60.09 | 12.26 | 42.70 | 10.93 | 22.77 |
| volrend | 3.4 | 19.69 | 49.72 | 10.35 | 34.82 | 10.41 | 19.73 |
| water_nsquared | 3.4 | 22.03 | 57.88 | 12.53 | 41.33 | 10.96 | 20.26 |
| water_spatial | 3.4 | 19.92 | 48.66 | 10.42 | 33.56 | 10.55 | 19.21 |
| | | | | | | | |
| barnes | 0.8 | 3.70 | 5.63 | 0.13 | 0.40 | 3.61 | 6.43 |
| cholesky | 0.8 | 2.98 | 3.40 | 0.14 | 0.16 | 3.28 | 3.67 |
| fmm | 0.8 | 3.41 | 5.24 | 0.12 | 0.36 | 3.47 | 6.05 |
| lu_cb | 0.8 | 3.74 | 5.49 | 0.13 | 0.40 | 3.58 | 6.24 |
| lu_ncb | 0.8 | 3.58 | 5.69 | 0.13 | 0.41 | 3.49 | 6.69 |
| ocean_cp | 0.8 | 4.25 | 6.32 | 0.13 | 0.27 | 4.13 | 7.48 |
| ocean_ncp | 0.8 | 4.07 | 6.28 | 0.14 | 0.27 | 4.10 | 7.46 |
| radiosity | 0.8 | 3.53 | 3.88 | 0.12 | 0.17 | 3.36 | 3.76 |
| radix | 0.8 | 3.19 | 4.20 | 0.09 | 0.24 | 3.07 | 4.79 |
| raytrace | 0.8 | 4.01 | 6.21 | 0.14 | 0.47 | 3.79 | 7.19 |
| volrend | 0.8 | 3.66 | 5.47 | 0.12 | 0.42 | 3.49 | 6.22 |
| water_nsquared | 0.8 | 3.94 | 6.19 | 0.14 | 0.46 | 3.79 | 7.19 |
| water_spatial | 0.8 | 3.65 | 5.66 | 0.13 | 0.41 | 3.54 | 6.27 |
| barnes | 3.4 | 12.99 | 36.19 | 9.72 | 30.40 | 3.61 | 6.43 |
| cholesky | 3.4 | 13.73 | 15.96 | 10.68 | 12.48 | 3.28 | 3.67 |
| fmm | 3.4 | 12.80 | 33.09 | 9.58 | 27.66 | 3.47 | 6.05 |
| lu_cb | 3.4 | 12.96 | 36.58 | 9.64 | 30.92 | 3.58 | 6.24 |
| lu_ncb | 3.4 | 13.02 | 37.51 | 9.83 | 31.52 | 3.49 | 6.69 |
| ocean_cp | 3.4 | 14.00 | 27.30 | 10.31 | 20.63 | 4.13 | 7.48 |
| ocean_ncp | 3.4 | 14.47 | 27.56 | 10.75 | 20.89 | 4.10 | 7.46 |
| radiosity | 3.4 | 12.45 | 16.30 | 9.27 | 12.73 | 3.36 | 3.76 |
| radix | 3.4 | 9.69 | 22.84 | 6.75 | 18.57 | 3.07 | 4.79 |
| raytrace | 3.4 | 14.40 | 42.90 | 10.78 | 36.36 | 3.79 | 7.19 |
| volrend | 3.4 | 12.64 | 37.66 | 9.39 | 31.90 | 3.49 | 6.22 |
| water_nsquared | 3.4 | 14.37 | 41.97 | 11.03 | 35.49 | 3.79 | 7.19 |
| water_spatial | 3.4 | 12.82 | 36.74 | 9.60 | 31.12 | 3.54 | 6.27 |
| | | **measured** | **measured** | **modeled** | **modeled** | **modeled** | **modeled** |

Table 7: Evaluation of different **PARSEC** Benchmarks on Haswell (first half) and Skylake (second half) using runtime in seconds, energy in Joule and frequency in GHz. **E-Diff** and **E-Diff-glob** denote percentage differences in energy for application-dependent and application-independent modeling

| | **Runtime** | | **Speedup** | | **Energy Consumed** | | **Best Freq** | | $s_{opt}$ (f) | | **E-Diff** [%] | | **E-Diff-glob** [%] | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Benchmark** | min | max | f=0.8 | f=3.4 | min | max | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 |
| blackscholes | 37.12 | 558.67 | 3.60 | 3.54 | 948.56 | 2323.36 | 2.3 | 1.2 | 1.83 (1.9) | 2.13 (1.5) | 2.3 | 6.2 | 2.3 | 3.0 |
| bodytrack | 31.52 | 463.37 | 3.72 | 3.47 | 868.69 | 2134.53 | 2.3 | 1.2 | 1.80 (1.9) | 2.15 (1.5) | 3.5 | 7.2 | 3.5 | 3.8 |
| canneal | 71.00 | 527.51 | 2.59 | 2.66 | 1209.09 | 2851.98 | 1.5 | 1.2 | 1.68 (2.1) | 1.86 (1.9) | 6.3 | 6.9 | 1.6 | 7.0 |
| facesim | 81.05 | 1136.57 | 3.43 | 3.95 | 2958.25 | 6071.95 | 2.1 | 2.1 | 1.36 (2.5) | 1.64 (2.1) | 6.3 | 0 | 0.9 | 1.8 |
| ferret | 60.91 | 1080.28 | 1.00 | 0.94 | 2482.52 | 5357.96 | 2.1 | 1.7 | 1.50 (2.3) | 1.57 (2.1) | 0.3 | 2.5 | 2.1 | 0 |
| fluidanimate | 63.30 | 1054.09 | 4.32 | 4.02 | 1949.83 | 5077.73 | 2.3 | 1.2 | 1.80 (1.9) | 2.20 (1.5) | 2.9 | 7.3 | 2.9 | 6.4 |
| freqmine | 80.63 | 1581.32 | 4.67 | 4.65 | 2742.39 | 7700.87 | 2.1 | 1.2 | 1.81 (1.9) | 2.29 (1.5) | 2.1 | 5.1 | 2.1 | 6.9 |
| streamcluster | 71.25 | 918.56 | 5.19 | 4.99 | 1564.56 | 5222.48 | 1.7 | 1.2 | 1.55 (2.1) | 1.89 (1.9) | 3.0 | 12.4 | 1.1 | 7.8 |
| swaptions | 39.46 | 867.48 | 5.18 | 5.18 | 1411.24 | 4313.39 | 2.3 | 1.2 | 1.78 (1.9) | 2.35 (1.4) | 1.7 | 4.0 | 1.7 | 3.8 |
| vips | 21.63 | 341.41 | 4.30 | 3.76 | 648.34 | 1703.58 | 2.1 | 1.2 | 1.78 (1.9) | 2.03 (1.7) | 1.8 | 6.9 | 1.7 | 6.8 |
| x264 | 16.87 | 396.76 | 5.95 | 5.75 | 560.11 | 2155.81 | 2.1 | 1.2 | 1.73 (1.9) | 2.18 (1.5) | 1.0 | 5.0 | 0.9 | 6.5 |
| blackscholes | 34.02 | 333.86 | 2.97 | 3.70 | 539.53 | 1546.02 | 2.1 | 1.2 | 1.97 (1.7) | 2.53 (1.4) | 2.3 | 1.3 | 2.3 | 1.3 |
| bodytrack | 27.15 | 294.10 | 2.91 | 3.91 | 490.99 | 1314.92 | 1.7 | 1.2 | 1.97 (1.7) | 2.85 (1.2) | 0 | 0 | 0 | 0.5 |
| canneal | 67.94 | 456.88 | 2.80 | 2.77 | 650.01 | 2015.27 | 1.5 | 0.8 | 1.92 (1.7) | 2.28 (1.5) | 2.8 | 2.8 | 2.8 | 1.9 |
| facesim | 76.79 | 1043.79 | 3.70 | 4.84 | 1686.56 | 5837.58 | 1.7 | 1.4 | 1.80 (1.9) | 2.23 (1.5) | 0.7 | 1.5 | 0 | 0 |
| ferret | 58.79 | 746.15 | 3.55 | 4.13 | 1262.20 | 4129.74 | 1.9 | 1.2 | 1.18 (1.9) | 2.01 (1.7) | 0 | 4.3 | 1.2 | 0.3 |
| fluidanimate | 60.51 | 636.68 | 2.97 | 3.95 | 1151.72 | 3151.76 | 1.9 | 1.0 | 2.02 (1.7) | 2.79 (1.2) | 0.4 | 0 | 0.4 | 1.1 |
| freqmine | 75.31 | 1011.15 | 3.55 | 4.83 | 1543.42 | 4726.40 | 2.1 | 1.0 | 2.02 (1.7) | 2.97 (1.2) | 1.3 | 1.0 | 1.3 | 1.8 |
| streamcluster | 70.36 | 878.20 | 5.29 | 4.77 | 878.56 | 3851.71 | 1.5 | 0.8 | 1.96 (1.7) | 2.31 (1.4) | 3.5 | 4.1 | 3.5 | 1.8 |
| swaptions | 35.91 | 516.00 | 3.68 | 5.31 | 779.57 | 2529.71 | 1.9 | 1.0 | 1.98 (1.7) | 2.96 (1.2) | 6.3 | 1.6 | 6.3 | 2.0 |
| vips | 18.54 | 221.89 | 3.28 | 4.27 | 367.38 | 1067.31 | 2.1 | 1.0 | 2.03 (1.7) | 2.86 (1.2) | 0.2 | 0.1 | 0.2 | 0.4 |
| x264 | 14.61 | 255.89 | 4.92 | 6.20 | 304.57 | 1197.61 | 1.7 | 1.0 | 2.01 (1.7) | 2.86 (1.2) | 0 | 0.1 | 0 | 1.5 |
| | **measured** | | **measured** | | **measured** | | **measured** | | **modeled** | **modeled** | **modeled** | | **modeled** | |

for $p$=1 and $p$=8 threads (columns 9 and 10, corresponding frequencies shown in parentheses), and the percentage difference between the minimum energy consumption measured and the energy consumption for the frequency for which the minimum energy consumption is expected according to the analytical energy model (columns 11 and 12). Columns 13 and 14 present results for an application-independent modeling, which will be addressed in Subsection 5.5.

From Table 7 it can be seen that for each application there is a large variation of the runtime and energy consumption when using different frequencies and different numbers of threads, i.e., varying the operational frequency and the number of executing threads has a significant impact and provides a good source for saving energy. For the different benchmarks, there is a large difference in the frequencies at which the minimum energy consumption results (between 1.0 GHz and 2.3 GHz). Moreover, for $p$=8 threads a smaller frequency is required to obtain the minimum energy consumption than for $p$=1, see columns 7 and 8. This corresponds to a larger scaling factor in columns 9 and 10.

An important result is shown in columns 11 and 12 containing the percentage difference between the measured and predicted energy values. An entry 0 indicates an exact fit. The percentage deviation lies well below 10% except for one case. The maximum percentage deviations are 12.4% on the Haswell architecture (streamcluster application for $p = 8$) and 6.3% on the Skylake (swaptions application for $p = 1$). The average percentage deviation is 3.4% and 6.2% for $p$=1 and $p$=8, respectively, for the Haswell processor. For the Skylake, these deviations are 1.9% and 1.0%, respectively, meaning that the model fits significantly better for the Skylake than for the Haswell architecture.

Comparing the Haswell and the Skylake architecture, the best frequencies are typically smaller for Skylake than for Haswell. The smallest energy consumptions typically result for frequencies that are larger than the minimum frequency provided. For the Haswell, the frequency for which the minimum energy consumption results lies between 1.2 GHz and 2.3 GHz, with smaller frequencies for a larger number of threads. For the Skylake, these frequencies lie between 0.8 GHz and 2.1 GHz, again with smaller frequencies for a larger number of threads.

Table 8 contains the corresponding information as Table 7 for the SPLASH-2 benchmarks. The application-specific modeling (columns 11 and 12) is usually very accurate both for $p = 1$ and $p = 8$, except for the two ocean benchmarks for $p = 8$. This effect is caused by an irregular behavior of the parallel execution time of these

Table 8: Evaluation of the **SPLASH-2** benchmarks on Haswell (first half) and Skylake (second half) using runtime in seconds, energy in Joule and frequency in GHz

| | **Runtime** | | **Speedup** | | **Energy Consumed** | | **Best Freq** | | $s_{opt}$ (f) | | **E-Diff**[%] | | **E-Diff-glob**[%] | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Benchmark** | min | max | f=0.8 | f=3.4 | min | max | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 |
| barnes | 25.436 | 442.727 | 3.975 | 4.224 | 1199.80 | 4063.08 | 3.2 | 3.4 | 1.23 (2.7) | 1.44 (2.3) | 4.4 | 4.7 | 0.2 | 2.0 |
| cholesky | 0.3567 | 1.52708 | 1.111 | 1.850 | 7.71460 | 14.4705 | 3.2 | 3.0 | 1.14 (3.0) | 1.27 (2.7) | 8.0 | 2.4 | 8.0 | 5.2 |
| fmm | 27.988 | 317.544 | 3.611 | 2.855 | 969.552 | 2710.94 | 3.0 | 1.9 | 1.23 (2.7) | 1.30 (2.7) | 3.1 | 7.2 | 0.0 | 7.9 |
| lu_cb | 26.889 | 408.141 | 3.614 | 3.616 | 1323.18 | 3430.18 | 3.0 | 1.9 | 1.32 (2.5) | 1.58 (2.1) | 5.8 | 0.95 | 0.0 | 8.6 |
| lu_ncb | 35.209 | 502.018 | 3.819 | 3.691 | 1588.19 | 4588.54 | 3.0 | 1.9 | 1.28 (2.7) | 1.55 (2.1) | 0.9 | 0.35 | 0.0 | 11.2 |
| ocean_cp | 50.570 | 289.784 | 3.660 | 1.503 | 1188.19 | 2865.26 | 3.0 | 1.0 | 1.25 (2.7) | 1.44 (2.3) | 2.1 | 27.8 | 0.0 | 40.6 |
| ocean_ncp | 54.661 | 532.340 | 5.509 | 2.425 | 1400.14 | 5020.83 | 3.0 | 1.0 | 1.26 (2.7) | 1.40 (2.5) | 2.6 | 17.9 | 0.0 | 31.0 |
| radiosity | 108.56 | 534.572 | 1.131 | 1.165 | 2478.76 | 4713.34 | 3.4 | 3.0 | 1.26 (2.7) | 1.36 (2.5) | 4.9 | 7.0 | 0.7 | 7.0 |
| radix | 7.8211 | 140.550 | 6.553 | 4.278 | 238.103 | 1177.16 | 3.4 | 2.1 | 1.16 (3.0) | 1.39 (2.5) | 2.5 | 7.1 | 2.6 | 7.3 |
| raytrace | 22.863 | 489.276 | 5.047 | 5.015 | 1332.35 | 4482.98 | 3.0 | 2.1 | 1.30 (2.5) | 1.55 (2.1) | 5.5 | 0.0 | 0.0 | 2.4 |
| volrend | 24.689 | 479.362 | 4.534 | 4.461 | 1201.91 | 4192.04 | 3.4 | 1.9 | 1.25 (2.7) | 1.52 (2.3) | 6.5 | 5.3 | 5.2 | 4.4 |
| water_nsquared | 84.675 | 1253.17 | 3.782 | 3.631 | 4097.42 | 11647.5 | 3.0 | 2.1 | 1.31 (2.5) | 1.59 (2.1) | 3.4 | 0.0 | 0.0 | 5.6 |
| water_spatial | 24.841 | 495.010 | 4.812 | 4.838 | 1166.13 | 4497.56 | 3.2 | 1.9 | 1.25 (2.7) | 1.51 (2.3) | 3.6 | 4.5 | 1.1 | 4.3 |
| | | | | | | | | | | | | | | |
| barnes | 21.39 | 232.84 | 3.05 | 4.48 | 417.23 | 1244.07 | 1.7 | 1.2 | 2.01 (1.7) | 2.79 (1.2) | 0.0 | 0.0 | 0.0 | 2.2 |
| cholesky | 0.32 | 1.32 | 1.10 | 1.13 | 3.39 | 5.14 | 1.7 | 1.5 | 2.22 (1.5) | 2.26 (1.5) | 0.2 | 0.0 | 0.0 | 2.9 |
| fmm | 18.64 | 198.87 | 3.21 | 3.88 | 322.22 | 926.49 | 1.7 | 1.0 | 2.03 (1.7) | 2.73 (1.2) | 0.0 | 0.2 | 0.0 | 1.4 |
| lu_cb | 28.65 | 317.63 | 2.97 | 4.24 | 576.70 | 1573.95 | 1.5 | 1.2 | 2.00 (1.7) | 2.89 (1.2) | 0.4 | 0.0 | 0.4 | 0.8 |
| lu_ncb | 34.78 | 407.64 | 3.36 | 4.18 | 652.01 | 1892.70 | 1.5 | 1.0 | 2.05 (1.7) | 2.79 (1.2) | 1.8 | 0.8 | 1.8 | 2.9 |
| ocean_cp | 34.98 | 125.07 | 2.36 | 1.59 | 335.16 | 784.12 | 1.5 | 0.8 | 1.94 (1.7) | 2.04 (1.7) | 0.2 | 23.2 | 0.2 | 11.6 |
| ocean_ncp | 39.60 | 139.62 | 2.24 | 1.58 | 388.21 | 1103.04 | 1.5 | 1.0 | 1.99 (1.7) | 2.06 (1.7) | 1.4 | 20.2 | 1.4 | 9.9 |
| radiosity | 98.38 | 317.35 | 1.13 | 1.21 | 1073.33 | 1740.68 | 1.7 | 1.5 | 2.03 (1.7) | 2.25 (1.5) | 0.0 | 0.9 | 0.0 | 0.0 |
| radix | 6.48 | 94.27 | 4.77 | 5.52 | 78.92 | 346.79 | 1.9 | 1.5 | 1.82 (1.9) | 2.45 (1.4) | 0.0 | 0.0 | 1.4 | 0.0 |
| raytrace | 20.88 | 273.69 | 3.46 | 4.98 | 477.11 | 1498.99 | 1.7 | 1.0 | 2.06 (1.7) | 2.94 (1.2) | 0.0 | 0.9 | 0.0 | 2.3 |
| volrend | 23.71 | 298.27 | 3.33 | 4.80 | 480.75 | 1437.68 | 1.7 | 1.2 | 2.00 (1.7) | 2.97 (1.2) | 0.0 | 0.0 | 0.0 | 1.1 |
| water_nsquared | 72.83 | 715.78 | 2.91 | 3.84 | 1519.57 | 4015.32 | 1.5 | 1.0 | 2.09 (1.7) | 2.89 (1.2) | 0.9 | 1.2 | 0.9 | 3.7 |
| water_spatial | 22.35 | 279.36 | 3.56 | 4.80 | 439.87 | 1376.12 | 1.7 | 1.0 | 2.01 (1.7) | 2.89 (1.2) | 0.0 | 0.8 | 0.0 | 0.5 |
| | **measured** | | **measured** | | **measured** | | **measured** | | **modeled** | **modeled** | **modeled** | | **modeled** | |

two benchmarks for $p = 8$ for varying frequencies: when increasing the frequency, the execution time reduction is much smaller than expected by the model and much smaller than it can be observed for all other PARSEC and SPLASH-2 applications. On the other hand, the power consumption of the two ocean benchmarks increases normally. This behavior is not covered by the energy model, and therefore larger deviations result. The effect cannot be observed for $p = 1$. This effect will be leveled out by using an application-independent modeling, see Subsection 5.5.

Figure 8 shows the optimal frequencies that lead to the smallest measured energy consumption for one and eight threads and compares the measured optimal frequencies with the frequencies that have been computed according to Equ. (6) for $s_{opt}$ for the i7 Skylake. The left diagram covers the PARSEC benchmarks, the right diagram covers the SPLASH-2 benchmarks. The application-independent values for $s_{opt}$ are also shown as horizontal lines. It can be seen that for some of the benchmarks, the modeled values for the optimal frequency are quite different from the measured values. However, the variations of the energy consumption with the frequency are not very large, see Fig. 3, and larger differences in the frequency do not necessarily lead to large differences in the energy consumption, see Tables 7 and 8 for the resulting deviations in the energy consumption. Figure 9 contains the same information as Figure 8 for the Haswell processor.

## 5.4 Application-specific optimal EDP frequencies

The modeling for the optimal EDP frequencies are given in Table 9 for the PARSEC benchmarks and in Table 10 for the SPLASH-2 benchmarks. Both tables contain the minimum and maximum measured EDP (in Joule · seconds) for different frequencies and different numbers of threads (columns 1 and 2), the frequency for which the measured EDP is at a minimum for $p$=1 and $p$=8 threads (columns 3 and 4), the optimum EDP scaling factors derived with Equ. (9) (columns 5 and 6), and the percentage difference between the minimum EDP measured and the EDP for the analytically determined EDP frequency (columns 7 and 8).

In Table 9, two benchmarks (facesim, ferret) have very small EDP values, which is caused by their short execution times, given in Table 7. A comparison with the best energy frequencies from Tables 7 and 9 show that the smallest EDP values require larger frequencies than the smallest energy values, since the execution time plays a larger role for the EDP. This confirms the EDP model from Section 2.3 and the optimum scaling factors computed with this model, see columns 5 and 6 in Table 9, which are smaller than the scaling factors in columns 9 and 10 of Table 7.

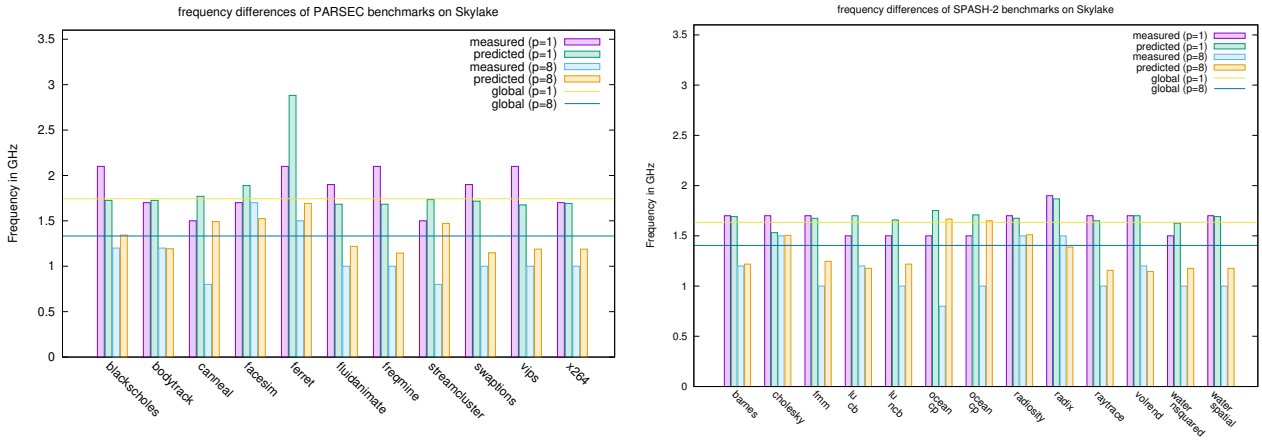


Figure 8: Comparison of measured and predicted optimum frequency values for PARSEC (left) and SPLASH-2 (right) benchmark suites for Skylake.

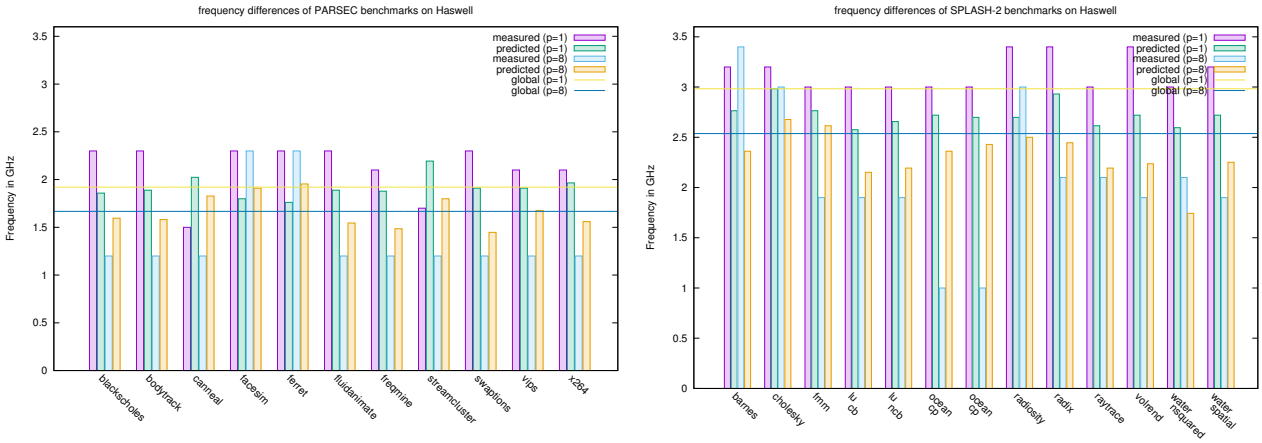


Figure 9: Comparison of measured and predicted optimum frequency values for PARSEC (left) and SPLASH-2 (right) benchmark suites for Haswell.

Table 9: Evaluation of the EDP for the **PARSEC** Benchmarks on Haswell (first half) and Skylake (second half)

| | **EDP** | | **Best Freq** | | $s_{opt}^{EDP}$ (f) | | **EDP-Diff**[%] | | **EDP-Diff-glob**[%] | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Benchmark** | min | max | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 |
| blackscholes | 47745.13 | 1297994.89 | 3.4 | 3.2 | 1.15 (3.0) | 1.34 (2.5) | 5.4 | 11.9 | 5.7 | 7.3 |
| bodytrack | 37647.80 | 989076.15 | 3.4 | 3.2 | 1.13 (3.0) | 1.35 (2.5) | 4.1 | 16.2 | 4.2 | 14.2 |
| canneal | 120885.86 | 1354092.07 | 2.8 | 2.7 | 1.06 (3.2) | 1.17 (3.0) | 14.1 | 5.9 | 4.3 | 0 |
| facesim | 292995 | 6901199 | 3.4 | 2.8 | 0.85 (3.4) | 1.03 (3.2) | 0 | 2.4 | 6.1 | 2.6 |
| ferret | 189153 | 5788118 | 3.2 | 3.4 | 0.94 (3.4) | 0.98 (3.4) | 1.2 | 0 | 6.0 | 14.5 |
| fluidanimate | 5352385.18 | 186174.95 | 3.4 | 3.2 | 1.13 (3.0) | 1.38 (2.5) | 2.4 | 6.2 | 2.5 | 2.8 |
| freqmine | 326402.63 | 12177540.17 | 3.4 | 3.2 | 1.14 (3.0) | 1.44 (2.3) | 3.1 | 9.7 | 3.1 | 6.6 |
| streamcluster | 157654.12 | 4797148.67 | 2.3 | 1.9 | 0.97 (3.4) | 1.19 (2.8) | 18.5 | 12.1 | 10.4 | 10.4 |
| swaptions | 81911.89 | 3741768.23 | 3.4 | 3.4 | 1.12 (3.0) | 1.48 (2.3) | 4.8 | 10.4 | 5.0 | 7.77 |
| vips | 19470.45 | 581622.91 | 3.4 | 3.0 | 1.12 (3.0) | 1.28 (2.7) | 4.1 | 4.9 | 4.1 | 5.1 |
| x264 | 13792.44 | 855348.67 | 3.4 | 3.4 | 1.09 (3.2) | 1.37 (2.5) | 7.6 | 11.1 | 5.3 | 7.4 |
| | | | | | | | | | | |
| blackscholes | 29693.18 | 408471.95 | 3.4 | 2.7 | 1.24 (2.7) | 1.59 (2.1) | 3.0 | 9.3 | 1.9 | 10.3 |
| bodytrack | 22756.40 | 314972.63 | 3.4 | 2.7 | 1.24 (2.7) | 1.79 (1.9) | 3.5 | 12.0 | 2.6 | 7.7 |
| canneal | 68628.75 | 569316.84 | 2.7 | 2.3 | 1.21 (2.8) | 1.43 (2.3) | 2.9 | 0 | 2.9 | 1.0 |
| facesim | 224735 | 4467468 | 2.8 | 2.5 | 1.13 (3.0) | 1.4 (2.5) | 2.6 | 0 | 0 | 4.2 |
| ferret | 127648 | 2244714 | 2.7 | 2.5 | 1.16 (2.8) | 1.26 (2.7) | 0.6 | 0.6 | 2.4 | 4.4 |
| fluidanimate | 120781.00 | 1529633.80 | 3.4 | 2.5 | 1.27 (2.7) | 1.76 (1.9) | 0.3 | 8.5 | 1.4 | 4.7 |
| freqmine | 207365.11 | 3777602.27 | 3.4 | 2.5 | 1.27 (2.7) | 1.87 (1.9) | 2.1 | 10.5 | 1.9 | 6.9 |
| streamcluster | 94014.90 | 2456987.16 | 2.3 | 1.7 | 1.23 (2.7) | 1.45 (2.3) | 0.8 | 3.7 | 3.3 | 1.8 |
| swaptions | 49561.53 | 1031320.00 | 3.4 | 2.7 | 1.25 (2.7) | 1.86 (1.9) | 2.9 | 11.8 | 4.3 | 7.0 |
| vips | 11839.69 | 183456.93 | 3.2 | 2.7 | 1.28 (2.7) | 1.80 (1.9) | 0.4 | 11.2 | 1.3 | 7.0 |
| x264 | 7760.78 | 244215.62 | 3.4 | 2.5 | 1.26 (2.7) | 1.80 (1.9) | 6.5 | 8.7 | 1.8 | 5.0 |
| | **measured** | | **measured** | | **modeled** | **modeled** | **modeled** | | **modeled** | |

The percentage deviation between the minimum EDP value and the EDP value for the frequency computed with the EDP model are slightly larger than for the energy values: For the PARSEC benchmarks, the average percentage deviations are 6.8 % and 8.5 % for the Haswell processor for $p = 1$ and $p = 8$, respectively, and 3.8 % and 9.3 % for the Skylake for $p = 1$ and $p = 8$, respectively. Thus, the average deviations are larger than the deviations for the energy, especially for the Skylake processor for $p = 8$. This can be explained by the strong influence of the application-specific execution time, which seems to be more difficult to capture by the model, especially for an increasing number of threads.

## 5.5 Application-independent optimal frequencies

The goal of an application-independent modeling is to derive only one optimum scaling factor $s_{glob}$ that is reasonable suitable for all benchmarks. The method to derive such an application-independent scaling factor is to subdivide the PARSEC and SPLASH-2 benchmark sets into a test set, containing three benchmarks, and a validation set, containing the remaining benchmarks. We select the first three benchmarks (blackscholes, bodytrack, canneal for PARSEC and barnes, cholesky and fmm for SPLASH-2) as test set. For each benchmark in the test set, the values for $P_{dyn}$ and $P_{static}$ are determined by the least-squares method as it was done in Subsection 5.3. These values are used to calculate mean values $P_{dyn}^{mean}$ and $P_{static}^{mean}$, which are then used to calculate an optimum scaling factor $s_{glob}$ according to Equ. (6). Tables 11 and 12 shows the resulting values for the global optimum scaling factor $s_{glob}$ along with the corresponding operational frequencies for $p = 1$ and $p = 8$ for the PARSEC and SPLASH-2 benchmarks. The global optimum scaling factor $s_{glob}$ is then applied to the validation set to which the other benchmarks belong. We calculate two kinds of percentage difference to evaluate the quality of $s_{glob}$ for the benchmarks in the validation set, which are (i) the percentage difference between the minimum measured energy consumption and the energy consumption for the frequency to $s_{glob}$ and (ii) the percentage difference between the individual $s_{opt}$ values and $s_{glob}$. The percentage deviations of the energy are given in the last two columns of Tables 7 and 8. Selecting other benchmarks for the test set leads to very similar results.

For the PARSEC benchmarks, the percentage deviation lies below 10% except for one case, which has a maximum percentage deviation of 11.3% on the Haswell architecture (ferret application for $p = 1$) and 6.3% on the Skylake (swaptions application for $p = 1$). The average percentage deviation for the Haswell processor is 2.9% and 5.7% for $p = 1$ and $p = 8$, respectively. Thus, surprisingly, the application-independent modeling leads to smaller

Table 10: Evaluation of the EDP for the **SPLASH-2** Benchmarks on Haswell (first half) and Skylake (second half)

| | **EDP** | | **Best Freq** | | $s_{opt}^{EDP}$ (f) | | **EDP-Diff**[%] | | **EDP-Diff-glob**[%] | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Benchmark** | min | max | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 | p=1 | p=8 |
| barnes | 30518.1 | 1798840 | 3.4 | 3.4 | 0.78 (3.4) | 0.91 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| cholesky | 2.70503 | 21.4811 | 3.4 | 3.2 | 0.71 (3.4) | 0.80 (3.4) | 0.0 | 2.39 | 0.0 | 2.39 |
| fmm | 28483.4 | 860843 | 3.4 | 3.4 | 0.78 (3.4) | 0.81 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| lu_cb | 38890.3 | 1400000 | 3.4 | 3.4 | 0.83 (3.4) | 0.99 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| lu_ncb | 66348.6 | 2303530 | 3.4 | 3.4 | 0.81 (3.4) | 0.97 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| ocean_cp | 74499.6 | 830306 | 3.4 | 1.5 | 0.78 (3.4) | 0.90 (3.4) | 0.0 | 53.58 | 0.0 | 53.58 |
| ocean_ncp | 92290.7 | 2672790 | 3.4 | 1.9 | 0.80 (3.4) | 0.80 (3.4) | 0.0 | 38.58 | 0.0 | 38.58 |
| radiosity | 295224 | 2519620 | 3.4 | 3.4 | 0.79 (3.4) | 0.85 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| radix | 2158.91 | 165450 | 3.4 | 3.0 | 0.73 (3.4) | 0.87 (3.4) | 0.0 | 4.4 | 0.0 | 4.4 |
| raytrace | 31409.5 | 2193410 | 3.4 | 3.4 | 0.82 (3.4) | 0.97 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| volrend | 30308.2 | 2009500 | 3.4 | 3.4 | 0.79 (3.4) | 0.95 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| water_nsquared | 415024 | 14596300 | 3.4 | 3.4 | 0.83 (3.4) | 1.00 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| water_spatial | 30029 | 2226340 | 3.4 | 3.4 | 0.79 (3.4) | 0.95 (3.4) | 0.0 | 0.0 | 0.0 | 0.0 |
| | | | | | | | | | | |
| barnes | 14989.6 | 201077 | 2.8 | 2.5 | 1.10 (3.0) | 1.33 (2.5) | 1.33 | 0.0 | 1.33 | 1.7 |
| cholesky | 1.56215 | 5.20819 | 2.8 | 3.2 | 1.17 (2.8) | 1.19 (2.8) | 0.0 | 1.75 | 1.83 | 3.24 |
| fmm | 10192.8 | 134910 | 3.0 | 2.5 | 1.11 (3.0) | 1.31 (2.5) | 0.0 | 0.0 | 0.0 | 0.7 |
| lu_cb | 28632.6 | 378218 | 3.4 | 2.5 | 1.10 (3.0) | 1.35 (2.5) | 2.2 | 0.0 | 2.2 | 0.03 |
| lu_ncb | 39448 | 595360 | 3.2 | 2.7 | 1.12 (3.0) | 1.33 (2.5) | 1.6 | 1.15 | 1.6 | 0.0 |
| ocean_cp | 15691.6 | 66498.7 | 2.5 | 1.4 | 1.07 (3.2) | 1.11 (3.0) | 5.65 | 74.3 | 2.45 | 48.42 |
| ocean_ncp | 20022.8 | 80603.1 | 2.5 | 1.5 | 1.09 (3.2) | 1.11 (3.0) | 7.93 | 74.08 | 5.09 | 47.51 |
| radiosity | 176397 | 355392 | 3.2 | 2.7 | 1.11 (3.0) | 1.19 (2.8) | 1.82 | 0.31 | 1.82 | 0.0 |
| radix | 801.203 | 28531.2 | 3.4 | 2.3 | 1.03 (3.2) | 1.24 (2.7) | 0.86 | 0.87 | 3.52 | 0.87 |
| raytrace | 17713.1 | 299872 | 3.2 | 2.8 | 1.12 (3.0) | 1.36 (2.5) | 0.43 | 0.98 | 0.43 | 0.18 |
| volrend | 20138.8 | 327530 | 3.4 | 2.7 | 1.10 (3.0) | 1.37 (2.5) | 1.63 | 0.76 | 1.63 | 0.0 |
| water_nsquared | 200446 | 2021020 | 3.2 | 2.5 | 1.13 (3.0) | 1.35 (2.5) | 1.34 | 0.0 | 1.25 | 1.82 |
| water_spatial | 16879.3 | 285822 | 3.4 | 2.5 | 1.10 (3.0) | 1.35 (2.5) | 1.58 | 0.0 | 1.58 | 2.97 |
| | **measured** | | **measured** | | **modeled** | **modeled** | **modeled** | | **modeled** | |

Table 11: Application-independent optimum scaling factors $s_{glob}$ and corresponding frequencies for PARSEC benchmarks.

| | **optimum scaling factors energy PARSEC** | |
|---|---|---|
| **processor** | $p = 1$ | $p = 8$ |
| Haswell | $s_{glob}$=1.77 (1.9 GHz) | $s_{glob}$=2.04 (1.7 GHz) |
| Skylake | $s_{glob}$=1.95 (1.7 GHz) | $s_{glob}$=2.55 (1.4 GHz) |
| Haswell-EP | $s_{glob}$=0.79 (2.4 GHz) | $s_{glob}$=1.04 (2.3 GHz) |
| Broadwell | $s_{glob}$=1.86 (1.6 GHz) | $s_{glob}$=2.49 (1.2 GHz) |

Table 12: Application-independent optimum scaling factors $s_{glob}$ and corresponding frequencies for SPLASH-2 benchmarks.

| | **optimum scaling factors energy SPLASH-2** | |
|---|---|---|
| **processor** | $p = 1$ | $p = 8$ |
| Haswell | $s_{glob}$=1.14 (3.0 GHz) | $s_{glob}$=1.34 (2.5 GHz) |
| Skylake | $s_{glob}$=2.08 (1.7 GHz) | $s_{glob}$=2.42 (1.4 GHz) |

Table 13: Application-independent optimum scaling factors $s_{glob}^{EDP}$ and corresponding frequencies for PARSEC benchmarks.

| | **optimum scaling factors EDP PARSEC** | |
|---|---|---|
| **processor** | $p = 1$ | $p = 8$ |
| Haswell | $s_{glob}^{EDP}$=1.11 (3.0 GHz) | $s_{glob}^{EDP}$=1.28 (2.7 GHz) |
| Skylake | $s_{glob}^{EDP}$=1.23 (2.8 GHz) | $s_{glob}^{EDP}$=1.60 (2.1 GHz) |
| Haswell-EP | $s_{glob}^{EDP}$=0.47 (2.4 GHz) | $s_{glob}^{EDP}$=0.66 (2.4 GHz) |
| Broadwell | $s_{glob}^{EDP}$=1.23 (2.5 GHz) | $s_{glob}^{EDP}$=1.58(1.9 GHz) |

Table 14: Application-independent optimum scaling factors $s_{glob}^{EDP}$ and corresponding frequencies for SPLASH-2 benchmarks.

| | **optimum scaling factors EDP SPLASH-2** | |
|---|---|---|
| **processor** | $p = 1$ | $p = 8$ |
| Haswell | $s_{glob}^{EDP}$=0.79 (3.4 GHz) | $s_{glob}^{EDP}$=0.86 (3.4 GHz) |
| Skylake | $s_{glob}^{EDP}$=1.13 (3.0 GHz) | $s_{glob}^{EDP}$=1.29 (2.7 GHz) |

average deviations than the application-specific modeling. For the Skylake, the average deviations are 1.7% and 1.4%, respectively. Thus, they are similar to the deviations for application-specific modeling. Again, the model fits significantly better for the Skylake than for the Haswell architecture. Simular results are obtained for the SPLASH-2 benchmarks: For $p = 1$, the percentage deviations are even smaller than for the PARSEC benchmarks. For $p = 8$, the ocean benchmarks lead to larger deviations due to their irregular behavior.

## 5.6 Application-independent optimal EDP frequencies

For the application-independent modeling of the EDP, the same approach has been used, and the resulting percentage deviations are given in the last two columns of Tables 9 and 10. The resulting EDP optimum scaling factors are given in Tables 13 and 14.

Larger deviations between the measured and modeled EDP values can be observed for two programs (facesim, ferret) on Skylake for $p = 8$, which is mainly caused by an irregular runtime behavior of these programs: The runtime is quite short compared to the other programs, see Table 7, and sudden runtime leaps occur between neighboring frequencies for $p = 8$, which cannot be captured by the model. These runtime leaps are not visible for the Haswell. The average deviations are (i) for Haswell 4.8% and 8.5% for $p = 1$ and $p = 8$, respectively, and (ii) for Skylake 1.8% and 10.8% for $p = 1$ and $p = 8$, respectively. For the SPLASH-2 benchmarks, the situation is similar with larger deviations again for the two ocean benchmarks.

## 5.7 Scalability analysis

This section investigates the development of the optimal frequencies for the energy consumption and the EDP with respect to the number of threads used for the execution of the different benchmarks. Figure 10 (left) shows this development of the optimal frequencies for the energy consumption for a varying number of threads. The optimal frequencies have been computed with the global application-independent model from Section 5.5 for the PARSEC benchmarks. The figure shows the optimal frequencies for the different processors that have been used for the experimental evaluation. Figure 10 (right) shows the optimal frequencies for the EDP again using the global application-independent model for the PARSEC benchmarks. The following observations can be made from the Figure 10:

- For both the energy consumption and the EDP, there is no large variation of the optimal frequency when increasing the number of threads for a specific processor. This is especially true for the Haswell-EP and for the energy consumption on the Broadwell processors. For the EDP on the Broadwell processor, an increase of the optimal frequency with the number of threads can be observed beyond two threads.

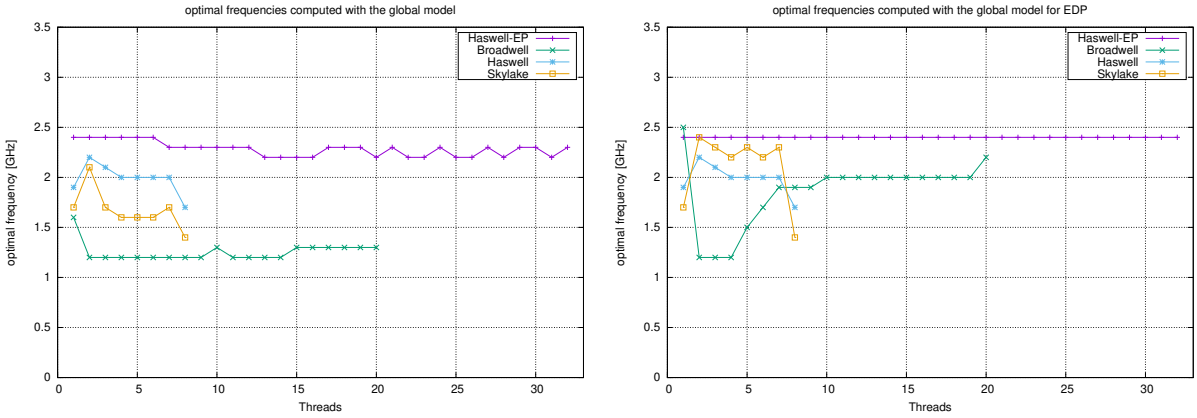


Figure 10: Development of the optimal frequencies for the energy consumption (left) and EDP (rigt) computed with the global model for the different processors considered using different numbers of threads.

- For the energy consumption on the Haswell and Skylake processors, a slight decrease of the optimal frequency with the number of threads can be observed beyond two threads.
- Especially for the energy consumption, the different processors require a different frequency to obtain the minimum energy consumption. For the desktop processors Haswell and Skylake, which have a frequency range between 0.8 GHz and 3.4 GHz, the optimal frequency typically lies between 1.4 GHz and 2.2 GHz, which is significantly below the maximum frequency. For the Broadwell processor, the optimal frequency is even smaller. In comparison, the server processor Haswell-EP requires a frequency between 2.2 GHz and 2.4 GHz for the minimum energy consumption, which is close to the maximum frequency 2.4 GHz available for this processor.

## 5.8 Summarizing modeling observations

In this section 5, we have applied the analytical power and energy models from Section 2 to derive application-specific and application-independent power, energy, and EDP models. The application-specific models are derived by calculating values for the dynamic and static power consumption, given in Table 6 and by computing optimal frequency scaling factors, given in Tables 7 – 10 according to the formulas in Section 2. For the calculation, we have proposed a methodology which can be applied in an application-specific or an application-independent way. The resulting modeling and parameter values exhibit the following behavior and quality:

- **benchmark suites**: The calculated optimum scaling factors for the PARSEC and SPLASH-2 benchmarks are quite similar for the Skylake architecture. This can be seen in Fig. 8 where the application-independent scaling factors are nearly identical. The situation is different for the Haswell architecture, see Fig. 9, where the scaling factors for the SPLASH-2 benchmarks are significantly larger than the scaling factors for the PARSEC benchmarks. This behavior can be attributed to the higher computational intensity of the SPLASH-2 benchmarks, which has a much larger effect on the Haswell than on the Skylake.
- **processor architecture**: The optimal scaling factors computed for the Skylake architecture are generally smaller than the optimal scaling factors computed for the Haswell. This can be observed for the energy and the EDP modeling. This effect is epecially significant for the SPLASH-2 benchmarks. Correspondingly, the minimum energy consumption is obtained for a smaller operational frequency on the Skylake than on the Haswell. This is confirmed by the measurements in Figures 3 and 4 and the modeling shown in Tables 7 – 10. The scaling factors for the Broadwell-E processor are similar than the scaling factors for the Skylake processor, see Tables 11 and 13. For the server processor Haswell-EP, the scaling factors are such that the maximum frequencies should be used in most of the cases for both the energy consumption and the EDP.
- **energy consumption and EDP**: Comparing Tables 11 and 12 with Tables 13 and 14, it can be seen that the application-independent scaling factors for the EDP are smaller than the corresponding scaling factors for the

energy consumption. Thus, a larger frequency leads to the optimization of the EDP. The same observation can be made for the application-specific scaling factors, see Tables 7–10, both for the measured and the modeled optimal frequencies. This confirms the theoretical prediction in Section 2.3 about the two scaling factors.

# 6 Related Work

Power-management techniques and features have been integrated in computer systems of different sizes and classes, from handheld devices to large servers [37, 13, 14]. An important feature for energy saving is the DVFS technique that trades off performance for power consumption by lowering the operating voltage and frequency if this is possible [42]. The approach to determine the voltage scaling factor that minimizes the total CPU energy consumption by taking both the dynamic power and the leakage power into consideration has been discussed in [23, 22, 42] for sequential executions. The influence of voltage scaling has been investigated in [24].

Power models for multicore designs have been developed in [13]. A detailed model for capturing the leakage currents on recent CMOS devices has been developed in [18]. The model takes DVFS into consideration and is much more detailed than the model in Section 2, since a quite accurate gate-level modeling is intended. The transition overhead for DVFS is addressed in [30] and a formal analysis of these overheads as well as a detailed analysis of the various components of the overhead are given. A similar power model as the one described in Section 2 has been used in [8] to evaluate the effect of DVFS on the execution of Lattice-Bolzmann simulations for computational fluid dynamics. A DVFS energy model similar to our model has been used in [10] for analyzing and tuning the energy efficiency of the fast-multipole method for n-body simulations. The experimental evaluation has been performed on a NVIDIA Jetson TK1 mobile system-on-chip (SoC) development board. The DVFS performance predictor DEP+BURST for managed multithreaded applications has been proposed in [2]. DEP+BURST takes into account synchronization, inter-thread dependencies, and store bursts. However, the energy consumption is not considered.

Performance prediction for DVFS processors is addressed in [36] with an emphasis on green supercomputing. Predictive models for multi-dimensional power-performance optimization on many-core processors is investigated in [12]. Especially, the interaction between DVFS and dynamic concurrency throttling (DCT) is investigated and a library targeting power-performance adaptation of OpenMP applications is presented. The experimental evaluation has been performed for the OpenMP versions of some of the NAS parallel benchmarks. The model used for the prediction is based on sample configurations and does not use a power model as presented in Section 2. The performance impact of DVFS for realistic memory systems is explored in [28]. The experimental evaluation is done with the SPEC CPU 2006 benchmarks, which are based on a sequential workload. The energy consumption of parallel algorithms for shared memory architectures based on the parallel external memory (PEM) model [4] has been discussed in [25]. [38] proposes a systemlevel iso-energy-efficiency model to analyze, evaluate and predict energy-performance of data intensive parallel applications running on cluster systems.

A detailed analytical treatment of energy models in the context of scheduling algorithms is given in [27]. Fundamental aspects of speed scaling and power-down scheduling are described in [11]. Power-down mechanisms and dynamic speed scaling techniques with a focus on algorithmic solutions are addressed in [3].

In the domain of realtime scheduling, many techniques for utilizing available waiting times based on DVFS have been considered, see, e.g., [19, 29, 41]. These approaches are usually based on heuristics and are not based on an analytical model as presented in this work. The effects of dynamic concurrency throttling (DCT) and DVFS in the context of a hybrid MPI/OpenMP programming model are considered in [26]. In particular, frequency selection is formulated as a variant of the 01 knapsack problem and dynamic programming is used to compute an approximation.

The energy delay product (EDP) has first been introduced in [17] to capture both energy consumption and execution time simultaneously. However, as it has been pointed out in [1], the EDP is not able to distinguish whether a shorter execution time or a better energy utilization leads to a smaller EDP value when comparing different version of a program. The biased effects of the EDP and its generalization $\mathrm{ED}^n\mathrm{P}$ for a nonnegative integer $n$ are also discussed in [20]. To separate energy and performance when considering different program versions, [1] introduces powerup and greenup metrics. These metrics are then used together with the speedup to identify different energy categories

of software. In the context of frequency scaling, [15] defines the power-aware speedup, which incorporates the frequency when computing the runtime speedup of programs. However, the energy consumption is not explicitly taken into consideration.

A detailed energy roofline model, which also takes details of the processor architecture such as memory hierarchies into consideration, has been presented in [9]. The model has been evaluated with micro-benchmarks for different architectures including x86, ARM, and GPU. In contrast to this model, our model is application-oriented and can be used for an application-specific as well as for an application-independent modeling. Our model abstracts from the details of the architecture considered. Instead, application-specific effects are captured by the parameters of the energy model, which are determined from measured runtime and energy values. An earlier version of this paper [33] contained preliminary results for the PARSEC benchmarks on the Skylake processor. Application-specific thermal energy models using both hardware-specific and application-specific parameters are described in [16]. The paper describes a detailed thermal hardware model along with an experimental verification by hardware measurements.

# 7 Conclusions

This article has demonstrated that analytical models can help to explain the dependence of the energy consumption of DVFS processors on the operational frequency and that such models can also be used for the EDP. Based on an existing sequential power model, this article presents parallel models for the energy consumption and the EDP of application codes and shows that the models can be used to compute optimal frequency values. The analytical models are derived from power formulas modeling the power activity of processors in general and without any assumption about the application to be executed. We have performed a detailed experimental evaluation of these models using the PARSEC and SPLASH-2 benchmark programs on four Intel processors.

A comparison with measured energy and EDP values shows that for most of the applications, the models can support an a priori selection of a frequency for which the energy consumption is near the minimum energy consumption over all frequencies. This has been done for an application-specific as well as an application-independent modeling for the energy consumption and the EDP. The application-independent modeling is based on a test set of applications for determining the parameters of the energy model and an independent validation set for the experimental comparison. The comparison shows that the modeling process is very accurate for the energy consumption and quite accurate for the EDP. This shows that the analytical models are suitable to explain and control the frequency scaling on recent DVFS processors.

Although we have taken the PARSEC and SPLASH-2 benchmark suites for our investigations, the method to derive application specific power, energy and EDP models is entirely independent from the specific multi-threaded applications or the internal behavior of the application and the hardware details. Instead, the derivation of the models relies entirely on the provision of measured performance data.

# Acknowledgement

This work was performed within the Federal Cluster of Excellence EXC 1075 "MERGE Technologies for Multifunctional Lightweight Structures" and supported by the German Research Foundation (DFG). This work is also supported by the German Ministry of Science and Education (BMBF) under project number 01IH16012A/B (SeASiTe).

# References

[1] S. Abdulsalam, Z. Zong, Q. Gu, and Meikang Qiu. Using the Greenup, Powerup, and Speedup metrics to evaluate software energy efficiency. In *Green Computing Conference and Sustainable Computing Conference (IGSC), 2015 Sixth International*, pages 1–8, Dec 2015.

[2] S. Akram, J. B. Sartor, and L. Eeckhout. Dvfs performance prediction for managed multithreaded applications. In *2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 12–23, April 2016.

[3] S. Albers. Energy-efficient algorithms. *Commun. ACM*, 53(5):86–96, May 2010.

[4] L. Arge, M.T. Goodrich, M. Nelson, and N. Sitchinava. Fundamental parallel algorithms for private-cache chip multiprocessors. In *SPAA '08: Proc. of the 20th Ann. Symp. on Parallelism in Algorithms and Architectures*, pages 197–206. ACM, 2008.

[5] C. Bienia, S. Kumar, and K. Li. PARSEC vs. SPLASH-2: A Quantitative Comparison of Two Multithreaded Benchmark Suites on Chip-Multiprocessors. In *Proc. of the 2008 Int. Symp. on Workload Characterization*, September 2008.

[6] C. Bienia, S. Kumar, J.P. Singh, and K. Li. The PARSEC Benchmark Suite: Characterization and Architectural Implications. In *Proc. of the 17th Int. Conf. on Parallel Architectures and Compilation Techniques*, October 2008.

[7] J.A. Butts and G.S. Sohi. A static power model for architects. In *In Proc. of the 33rd Int. Symp. on Microarchitecture (MICRO-33)*, 2000.

[8] E. Calore, A. Gabbana, S. F. Schifano, and R. Tripiccione. Evaluation of DVFS techniques on modern HPC processors and accelerators for energy-aware applications. *CoRR*, abs/1703.02788, 2017.

[9] J. Choi, M. Dukhan, X. Liu, and R. Vuduc. Algorithmic Time, Energy, and Power on Candidate HPC Compute Building Blocks. In *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*, pages 447–457, May 2014.

[10] J. W. Choi and R. W. Vuduc. Analyzing the energy efficiency of the fast multipole method using a dvfs-aware energy model. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 79–88, May 2016.

[11] M. Chrobak. Algorithmic Aspects of Energy-Efficient Computing. In I. Ahmad and S. Ranka, editors, *Handbook of Energy-Aware and Green Computing*, pages 311–329. CRC Press, 2012.

[12] M. Curtis-Maury, A. Shah, F. Blagojevic, D. S. Nikolopoulos, B. R. de Supinski, and M. Schulz. Prediction models for multi-dimensional power-performance optimization on many cores. In *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques*, PACT '08, pages 250–259. ACM, 2008.

[13] H. Esmaeilzadeh, E. Blem, R. Amant, K. Sankaralingam, and D. Burger. Power challenges may end the multicore era. *Commun. ACM*, 56(2):93–102, February 2013.

[14] A. Fedorova, J.C. Saez, D. Shelepov, and M. Prietol. Maximizing Power Efficiency with Asymmetric Multicore Systems. *Commun. ACM*, 52(12):48–57, December 2009.

[15] R. Ge and K. W. Cameron. Power-Aware Speedup. In *2007 IEEE International Parallel and Distributed Processing Symposium*, pages 1–10, March 2007.

[16] V. Getov, D. J. Kerbyson, M. Macduff, and A. Hoisie. Towards an Application-specific Thermal Energy Model of Current Processors. In *Proceedings of the 3rd International Workshop on Energy Efficient Supercomputing*, E2SC '15, pages 5:1–5:10, New York, NY, USA, 2015. ACM.

[17] R. Gonzalez and M. Horowitz. Energy dissipation in general purpose microprocessors. *IEEE Journal of Solid-State Circuits*, 31(9):1277–1284, Sep 1996.

[18] D. Helms, R. Eilers, M. Metzdorf, and W. Nebel. Leakage models for high-level power estimation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(8):1627–1639, Aug 2018.

[19] T. Horvath, T. Abdelzaher, K. Skadron, and X. Liu. Dynamic Voltage Scaling in Multitier Web Servers with End-to-End Delay Control. *IEEE Trans. Comput.*, 56(4):444–458, 2007.

[20] C. H. Hsu, W. C. Feng, and J. S. Archuleta. Towards efficient supercomputing: a quest for the right metric. In *19th IEEE Int. Parallel and Distributed Processing Symposium*, page 8, April 2005.

[21] Intel. *Intel 64 and IA-32 Architecture Software Developer's Manual, System Programming Guide*, 2011.

[22] S. Irani, S. Shukla, and R. Gupta. Algorithms for power savings. *ACM Trans. Algorithms*, 3(4):41, 2007.

[23] R. Jejurikar, C. Pereira, and R. Gupta. Leakage aware dynamic voltage scaling for real-time embedded systems. In *Proc. of the 41st Annual Design Automation Conference*, pages 275–280. ACM, 2004.

[24] T. Kim. Power Saving by Task-level dynamic Voltage Scaling: A Theoretical Aspect. In I. Ahmad and S. Ranka, editors, *Handbook of Energy-Aware and Green Computing*, pages 361–383. CRC Press, 2012.

[25] V.A. Korthikanti and G. Agha. Towards optimizing energy costs of algorithms for shared memory architectures. In *SPAA '10: Proceedings of the 22nd ACM symposium on Parallelism in algorithms and architectures*, pages 157–165, New York, NY, USA, 2010. ACM.

[26] D. Li, B.R. de Supinski, M. Schulz, D. Nikolopoulos, and K.W. Cameron. Strategies for Energy Efficient Resource Management of Hybrid Programming Models. *IEEE Transaction on Parallel and Distributed Systems*, 2012.

[27] Keqin Li. Performance analysis of power-aware task scheduling algorithms on multiprocessor computers with dynamic voltage and speed. *IEEE Trans. Parallel Distrib. Syst.*, 19(11):1484–1497, 2008.

[28] R. Miftakhutdinov, E. Ebrahimi, and Y. N. Patt. Predicting Performance Impact of DVFS for Realistic Memory Systems. In *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-45, pages 155–165, Washington, DC, USA, 2012. IEEE Computer Society.

[29] R. Mishra, N. Rastogi, D. Zhu, D. Mossé, and R. Melhem. Energy Aware Scheduling for Distributed Real-Time Systems. In *IPDPS '03: Proc. of the 17th Int. Symp. on Parallel and Distributed Processing*, page 21.2. IEEE Computer Society, 2003.

[30] S. Park, J. Park, D. Shin, Y. Wang, Q. Xie, M. Pedram, and N. Chang. Accurate modeling of the delay and energy overhead of dynamic voltage and frequency scaling in modern microprocessors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(5):695–708, May 2013.

[31] T. Rauber and G. Rünger. Modeling and Analyzing the Energy Consumption of Fork-Join-based Task Parallel Programs. *Concurrency and Computation: Practice and Experience*, 27(1):211–236, 2015.

[32] T. Rauber, G. Rünger, M. Schwind, H. Xu, and S. Melzner. Energy Measurement, Modeling, and Prediction for Processors with Frequency Scaling. *The Journal of Supercomputing*, 2014.

[33] T. Rauber, G. Rünger, and M Stachowski. Model-based Optimization of the Energy Efficiency of Multithreaded Applications. In *Proceedings of the 8th International Green and Sustainable Computing Conference (IGSC'17)*. IEEE, 2017.

[34] G. Rong, F. Xizhou, S. Shuaiwen, C. Hung-Ching, L. Dong, and K.W. Cameron. PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications. *IEEE Transactions on Parallel and Distributed Systems*, 21(5):658 –671, may 2010.

[35] E. Rotem, A. Naveh, A. Ananthakrishnan, D. Rajwan, and E. Weissmann. Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge. *IEEE Micro*, 32(2):20–27, 2012.

[36] B. Rountree, D. K. Lowenthal, M. Schulz, and B. R. de Supinski. Practical performance prediction under Dynamic Voltage Frequency Scaling. In *Proc. of the 2011 Int. Green Computing Conference and Workshops*, IGCC '11, pages 1–8, Washington, DC, USA, 2011. IEEE Computer Society.

[37] E. Saxe. Power-efficient software. *Commun. ACM*, 53(2):44–48, 2010.

[38] S. Song, C.-Y. Su, R. Ge, A. Vishnu, and K.W. Cameron. Iso-energy-efficiency: An approach to power-constrained parallel computation. In *Proc. of the 25th IEEE Int. Parallel and Distributed Processing Symp. (IPDPS 11)*. IEEE, 2011.

[39] J. Treibig, G. Hager, and G. Wellein. LIKWID: A Lightweight Performance-Oriented Tool Suite for x86 Multicore Environments. In *39th Int. Conf. on Parallel Processing Workshops*, ICPP '10, pages 207–216. IEEE Computer Society, 2010.

[40] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, and A. Gupta. The SPLASH-2 Programs: Characterization and Methodological Considerations. In *Proceedings of the 22nd Annual International Symposium on Computer Architecture*, pages 24–36, 1995.

[41] D. Zhu, R. Melhem, and D. Mossé. Energy efficient redundant configurations for real-time parallel reliable servers. *Real-Time Syst.*, 41(3):195–221, 2009.

[42] J. Zhuo and C. Chakrabarti. Energy-efficient dynamic task scheduling algorithms for DVS systems. *ACM Trans. Embed. Comput. Syst.*, 7(2):1–25, 2008.