The Journal of Supercomputing (2021) 77:2458–2485
https://doi.org/10.1007/s11227-020-03349-0

Energy consumption model in multicore architectures with variable frequency

Amilcar Meneses-Viveros[1] · Mireya Paredes-López[2] · Erika Hernández-Rubio[3] · Isidoro Gitler[4]

Published online: 17 June 2020

Abstract
Models extending Amdahl's law have been developed to study the behavior of parallel programs energy consumption. In addition, it has been shown that energy consumption of those programs also relies on the layout of the resources on the chip, such as power supply. Other extensions over Amdahl's law have been conducted to study the behavior of parallel programs speedup for frequency variable processors. Previous models have focused on the use of Turbo Boost in the parallel regions of a program, without considering that Turbo Boost also affects the sequential regions. Hence, we present a model to analyze energy consumption of parallel programs executed on Intel multicore processors with Turbo Boost frequencies to cover this gap. The model is an extension to Amdahl's law, and it is validated with a double-precision matrix multiplication running on Intel multicore processors that enable Turbo Boost technology.

Keywords Amdahl's law · Energy model · Multicore processors · Parallel programs · Turbo Boost · Variable frequency

✉ Amilcar Meneses-Viveros
ameneses@cs.cinvestav.mx

Mireya Paredes-López
mireya.paredes@udlap.mx

Erika Hernández-Rubio
ehernandezru@ipn.mx

[1] Department of Computer Science, Center for Research and Advanced Studies, Mexico City, Mexico

[2] Universidad de las Américas-Puebla, Cholula, Puebla, Mexico

[3] Instituto Politécnico Nacional, SEPI-ESCOM, Mexico City, Mexico

[4] Department of Mathematics, Center for Research and Advanced Studies, Mexico City, Mexico

1 Introduction

Generation, consumption and energy saving are not a minor issue. Although sources of clean energy are being sought, the generation of energy still causes large emissions of CO_2. Approximately 72% of global energy consumption goes to transportation, lighting and electrical heating, while other energy consumptions are mainly due to electrical and electronic equipment, including the computing devices, among others [1–6].

Energy consumption in computing devices is an area of research and development that is rapidly growing. This growth is due to the fact that there are several critical problems that must be thoroughly addressed, including the energy gap problem, energetic consumption for autonomous systems and the problem of exaflop scaling, among others [7–13]. Firstly, regarding the energy gap problem, Nokia (2010) reported that the power consumption used by new applications increases annually more than what mobile device batteries can yield. Requiring that new smartphones need to be recharged more often, limiting users mobility [7, 14]. Secondly, autonomous vehicles, such as robots and drones, require efficient energy management. In fact, if the algorithms that control or use them for specific tasks are not energetically optimal, the battery is at risk and can affect the tasks of the vehicle, even to its integrity [9–11]. Thirdly, related to the exaflop problem, it is known that there is a steady growth in supercomputer performance as can be seen from the Top500.org reports [12]. With nowadays computer technology, the increment of the number of floating point operations per second (flops) is directly proportional to power consumption; hence, exaflop computers would be energetically unviable. However, around the world, teams of engineers and scientists are aiming, by 2020, to manufacture exaflop computers with the target in energy consumption of 20 Mwatts per hour [12, 13].

Various strategies have been developed and implemented to save energy. For mobile devices: use of processors with low power consumption, mobile computation offloading strategies, ensuring that applications have sequential rather than parallel executions, and scheduler optimization for both processor and antennas, to name a few [15]. Some strategies to save energy have been developed, such as the use of approximate computing, to the effect that autonomous vehicles can save energy in the expensive computing operations they perform [16, 17]. For exaflop scaling: use of processing units that have a good flop-to-watt ratio including low power processors, use of accelerators such as GPUs or Xeon Phi, use of multicore processors with variable frequency, cooling technology and energy capture/recovering [12, 13].

Most of energy models based on extensions to Amdahl's law are not considering the use of variable frequency as Turbo Boost provides [18]. Meneses [19] presents an extension to Amdahl's law for Intel processors with Turbo Boost, but only to study the performance behavior and not the energy behavior of parallel programs. Verner [20] presents a model that considers frequency variation but only in the parallel regions of a program. On the other hand, it is necessary to have a model that helps to analyze the energy behavior of parallel programs running on multicore processors with Turbo Boost technology. The main contribution of this work is a model to analyze energy consumption of parallel programs executed on Intel multicore processors with Turbo Boost, which is an extension to Amdahl's law.

2 Related work

Power consumption is an important factor during the designing phase of new processors. Recently, the use of multicore processors has increased from 2 to 48 cores. In addition, it is common to manufacture processors with more memory cache and with frequencies higher than 3.6 Ghz. This frequency has a limit and it is due to the fact that energy consumption generates heat. For instance, when an instruction flow is being executed the energy that is consumed by the processor might cause overheat, ending requiring additional energy to cool it down [21–25].

Fuller and Miller [26] in 2010 addressed the power consumption problem, exploring new technologies for the design of modern processors. Furthermore, different strategies have been used to reduce power consumption of these new processor architectures, heterogeneous or asymmetric, including specialized cores, vectorization, scaled dynamic voltage frequency (DVFS) and systematic techniques for power activation, among others [27, 28].

The study of the multicore energy behavior has been focused on two different paths. The first one has been through mathematical models and the second one through experimenting. Table 1 shows all models related to power consumption analysis of multicore processors.

Traditionally, Amdahl's law model has been used to analyze parallel programming speedup [47] and it has been taken as basis for the study of power consumption of this programming in multicore architectures. Research on energy models based on extensions to Amdahl's law can be found in [29–32, 32–35]. Most of these studies have been focused on power consumption of multicore architectures without considering frequency variations. Kim et al. [48] conducted their research focused on frequency variations on the sequential regions of the parallel program. Pei et al. [35] present a model for heterogenous multicore processors;

Table 1 Models related to the study of power consumption

References	Model	Approach
[29–32, 32–35]	Amdahl's law extension	Based on Amdahl's law to implement a energy speedup model
[36–40]	Energy proportionality	Models to obtain indicators for the energy proportionality
[41–44]	Energy complexity	To extend the concepts of computational complexity of algorithm analysis to have an energy consumption model
[33, 45, 46]	Other models	Models based on sharing resources and energy delay product (EDP)

in this model, the Data Preparation Overhead (ODP) is focused to generate its extension to Amdahl's law.

Energy proportionality refers to a model that establishes that a server consumes power proportionally to its use [36–40]. Since its introduction in 2007, the model has been embraced by the server industry as a design goal for further optimizations of energy efficiency. In [36], the authors found that modern servers have taken the proportionality of server power consumption from linear to quadratic. Hsu et al. [37] reviewed different energy proportionality metrics and they proposed a unique unified indicator to represent them all, called EP (energy proportionality). Furthermore, the authors identified that while EP is considered better than other existing metrics, EP does not capture an important aspect of the server power proportionality, which is linearity. Energy proportionality is still under research for data centers environment [38–40].

Some models can be categorized as energy complexity models. Some authors [41–44] have formalized the problem of energy consumption by extending the ideas of computational complexity to a new area called energy complexity to explain its behavior. Actually, computational complexities have been obtained for traditional algorithms such as sorting algorithms. The main idea is to know the energy used as a function of problem size.

There are other models using different strategies for the analysis of energetic behavior based on processors with independent power supply resources [33, 45]. These resources can be cores, cache memory and network interconnection, among others. Basmadjian and de Meer [45] worked on the design of a software-based model of power consumption for multicore processors. They suggested that it is important to bear in mind that the presence of more than one processing unit directly affects the processor power behavior, because computational resources are shared. They also mentioned that measuring power consumption from hardware-level is not trivial for multiprocessors with a high quantity of circuits inside. Cho and Melhem [33] studied the effects of parallelism over a program performance and energy consumption. Their proposed model was tested on a machine that could apply core off-lining. They predicted that the introduction of more cores combined with a high percentage of parallel code in a process helps to reduce the energy consumption. Their main conclusion is that the more energy savings, the more common elements processing units there are, energized by the same power source.

Finally, another energy consumption model is presented in [46]. The authors take the energy delay product (EDP) as the metric for energy efficiency. The lowest EDP means the best energy efficiency. For a shared memory multicore processor system, they used processor overclocking and memory frequency scaling to achieve better performance and lower power consumption, providing better energy efficiency. Furthermore, the authors stated that it cannot be ensured that better energy efficiency can be achieved anytime we use processor overclocking. However, they also found that reducing memory frequency combined with processor overclocking cannot ensure that we achieve greater energy efficiency. Hence, they propose a processor overclocking and memory frequency scaling algorithm based on the Holistic Energy-Efficient (HEE) algorithm, which determines when to use processor overclocking to improve energy efficiency and when you could combine processor overclocking with memory frequency scaling for better energy efficiency.

The majority of research that have studied the energy consumption behavior in Intel multicore processors are based on experimentation. They have tested several benchmarks to study the impact of energy consumption on specific tasks of the system, such as memory access, use of DVFS (dynamic voltage frequency scalability), use of Turbo Boost, use of vectorization, use of counters of thermal, power use and operating system configurations, to name a few [18, 28, 49–57].

James et al. [25] present a performance analysis of Intel Core i7 processors (Nehalem). Their analysis combines different workload scenarios. The authors determine that Turbo Boost activation is due to inherent features of the application. They note that Turbo Boost mitigates Amdahl's law. This article reports that the reduction in time is 6%, while the energy consumption increases by 16%. They report that with Core i7 the step size for frequency increases is 133.33 MHz. Turbo Boost is made possible by a processor feature called power gating. Traditionally, an idle processor core consumes zero active power while dissipating static power due to leakage current. Power activation also aims to reduce the leakage current, further reducing the power consumption of the idle core. The additional available power margin can be diverted to the active cores to increase their voltage and frequency without violating the power, voltage and thermal enveloped [25]. Turbo Boost makes Nehalem a dynamically asymmetric processor; this means that different cores can execute the same set of instructions, while frequencies vary independently between cores. The experiments they do make them CPU intensive, floating point and integer, and with large memory accesses. In this article, they present a frequency-based processor power consumption model.

Acun et al. [49] present a series of experiments on 4 supercomputers in order to see the behavior of HPC applications on supercomputers with nodes that have Turbo Boost. The test program they use is MK double-precision matrix multiplication, as it represents a benchmark for heavy hardware use. The authors report a degradation in the performance of applications as the number of cores increases in processors with dynamic frequency change. The observed problem is that not all nodes increase the frequency at the same time. They tried trying to set the frequency constant but did not get good results. The solution they found was a process migration algorithm to processors with active Turbo Boost and which are observed to increase the frequency in the same way.

Aniruddha et al. in 2017 [50] proposed that the models to predict the performance of energy and temperature in clusters presuppose homogeneous execution systems, that is, processors and cores that run at the same frequencies, so that frequency variations must be carefully studied of processors. This paper presents an empirical study to see the performance of processors for HPC. In particular, they do the family review of Intel processor families with Turbo Boost. The performance problem due to energy variation increases as processors arrive with higher frequency dynamic increases. The variation is further amplified by the restrictions imposed by the hardware (number of cores, power management functionality, among others). In this paper, the authors show that with the hardware power constraint of the processor, the variation in processor performance and energy efficiency has increased up to fourfold in the latest Intel processors (Sandy Bridge, Ivy Bridge and Broadwell). This work suggests that there is a risk of greater variation in performance and energy efficiency, as the number of cores in a processor increases.

Current energy models based on extensions to Amdahl's law are not considering the use of variable frequency as Turbo Boost provides [18]. Verner [20] and Meneses [19] present an extension to Amdahl's law for Intel Processors with Turbo Boost, but only to study the performance behavior and not the energy behavior of parallel programs. On the other hand, it is necessary to have a model that helps to analyze the energy behavior of parallel programs running on multicore processors with Turbo Boost technology. Hence, we present a model to analyze energy consumption of parallel programs executed on multicore processors with variable frequencies, which is an extension to Amdahl's law.

3 Energy consumption model

We give an energy consumption model for multicore processors with variable frequency, based on Amdahl's law. This model extends the model proposed in [34] and covers processors with variable frequency.

The relation between *energy*, *power* and *time* is given by [14]:

$$energy = power \times time. \tag{1}$$

Based on this relation, our energy consumption model consists of two main components: the *power* and the *time*. On one hand, we obtain the power model for multicore processors of variable frequency taking into account two different measurements of power. The first one is the power associated to the processor's base frequency. The second one is the power associated to the processor after variable frequency has been enabled. Due to the fact that frequency is directly related to power, for higher frequency there is higher power consumption. Hence, for multicore processors with variable frequency, it is expected that as frequency increases an increment of power consumption is obtained. On the other hand, we introduce Amdahl's law to model the time of a parallel program running on a multicore processor. Then, we extend the model for multicore processors with variable frequency. Furthermore, this extension predicts not only the speedup in terms of time, but also the energetic speedup of the multicore processor. In Sects. 3.1 and 3.2, we describe both components of the energy consumption model, the power and the time models, respectively. Section 3.3 develops the energy model and Sect. 3.4 develops an extension of Amdahl's law for the energetic speedup.

3.1 Power

The power model is developed based on two cases, which we call W_{off} and W_{on}. Case W_{off} refers to the status when the processor does not have variation in its frequency. Case W_{on} refers to the status when the processor has an active variation in its frequency.

In general, according to [34], it is possible to represent the entire power usage by the addition of three parameters: *base power*, *power of active cores* and *idle power for all cores*. These parameters are denoted by W_{base}, pW_{active} and mW_{idle}, where p refers to the number of active cores and m to the total number of cores in the processor. The parameter pW_{active} can be pW_{aoff} or pW_{aon} depending on the case.

The power of a processor with m cores of which p are active is given by:

$$W_{off}(p) = W_{base} + pW_{aoff} + mW_{idle}, \tag{2}$$

$$W_{on}(p) = W_{base} + pW_{aon} + mW_{idle}. \tag{3}$$

Since $W_c = W_{base} + mW_{idle}$ is constant, we will use W_c in the future reference to this term. As stated in [45], this constant depends on the layout of the resources shared by the multicore processor including, cache memory and interconnection network, among others.

$$W_{off}(p) = W_c + pW_{aoff}, \tag{4}$$

$$W_{on}(p) = W_c + pW_{aon}. \tag{5}$$

When only one core is used, Eqs. (6) and (7) represent the power required by a sequential program in a multicore processor for both cases.

$$W_{off}(1) = W_c + W_{aoff}, \tag{6}$$

$$W_{on}(1) = W_c + W_{aon}. \tag{7}$$

3.2 Time

According to Amdahl [47], the execution time of a sequential program to solve a problem of size n, running on a single processor is given by Eq. (8), where $\sigma(n)$ is the serial portion of the computation and $\varphi(n)$ is the portion of the computation that can be executed in parallel.

$$T(n, 1) = \sigma(n) + \varphi(n). \tag{8}$$

Following the same principle, the execution time for a program running on a multicore processor is given by Eq. (9), where p denotes the number of cores in the multicore processor and $k(n, p)$ is the overhead related to the intercommunication between processors.

$$T(n,p) = \sigma(n) + \frac{\varphi(n)}{p} + k(n,p). \tag{9}$$

Hence, the speedup of a parallel program is the ratio between sequential execution time and parallel execution time as given by Eq. (11). Assuming $k(n,p) > 0$ and $\sigma(n) > 0$, we obtain inequality (10).

$$\Psi(n,p) = \frac{\sigma(n) + \varphi(n)}{\sigma(n) + \frac{\varphi(n)}{p} + k(n,p)}, \tag{10}$$

$$\Psi(n,p) \leq \frac{1}{f + (1-f)/p}, \quad \text{where } f = \frac{\sigma(n)}{\sigma(n) + \varphi(n)}. \tag{11}$$

Note that $\sigma(n) > 0$ because in practice all programs have a sequential code region, even if it is very small. And the inherent sequential proportion is given by f. From (10), it follows that the maximum speedup when $f \to 0$ is given by:

$$\lim_{f\to 0} \Psi(n,p) = p. \tag{12}$$

3.2.1 Amdahl's law for processors with variable frequency

We give an extension for the speedup (10) of multicore processors with variable frequency based on Amdahl's law. We consider the execution time of a program when the frequency change is active or not. Let $T_{off}(n,p)$ be the time a program uses to solve a problem of size n with p cores when the frequency change is not active. We consider that $T_{off}(n,1)$ and $T_{off}(n,p)$ are given by Eqs. (8) and (9), respectively.

To calculate the execution time of a sequential program to solve a problem of size n in a processor that has the frequency change active, we must consider the ratio of frequencies H given by:

$$H = \frac{\text{increased frequency}}{\text{base frequency}}.$$

Then, $T_{on}(n,1)$ and $T_{on}(n,p)$ are defined as

$$T_{on}(n,1) = \frac{\sigma(n)}{H} + \frac{\varphi(n)}{H}, \tag{13}$$

$$T_{on}(n,p) = \frac{\sigma(n)}{H} + \frac{\varphi(n)}{Hp} + \frac{\kappa(n,p)}{H}. \tag{14}$$

The speedup of a parallel program that solves a problem of size n with p cores on a multicore processor with variable frequency is given by:

$$\begin{aligned} \Psi(n,p) &= \frac{T_{off}(n,1)}{T_{on}(n,p)}) \\ &= \frac{\sigma(n) + \varphi(n)}{\frac{\sigma(n)}{H} + \frac{\varphi(n)}{Hp} + \frac{k(n,p)}{H}}, \end{aligned} \tag{15}$$

We obtain (16) from (15) (as in the case above for the time),

$$\begin{aligned}\Psi(n,p) &\leq \frac{\sigma(n)+\varphi(n)}{\frac{\sigma(n)}{H}+\frac{\varphi(n)}{Hp}} \\ &= H\left(\frac{\sigma(n)+\varphi(n)}{\sigma(n)+\frac{\varphi(n)}{p}}\right).\end{aligned} \tag{16}$$

Taking $f = \frac{\sigma(n)}{\sigma(n)+\varphi(n)}$, for $\sigma(n) > 0$, turns (16) into (17).

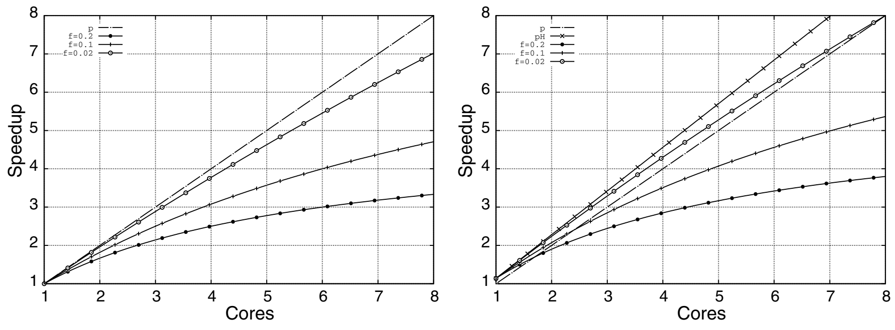
$$\Psi(n,p) \leq \frac{H}{f+\frac{(1-f)}{p}}. \tag{17}$$

Note that the maximum speedup for a parallel program running in a processor with frequency variation is when $\lim_{f\to 0}\Psi(n,p)$, then

$$\lim_{f\to 0}\Psi(n,p) \leq pH. \tag{18}$$

Figure 1a, b shows the speedup for three different values of f: 0.2, 0.1 and 0.02. Figure 1a shows the speedup of a parallel program without variation in the frequency (traditional Amdahl's law), and Fig. 1b shows the speedup of a parallel program using $H = 1.14$. The H value was taken assuming that the processor used increments in frequency from 2.7 to 3.1 GHz ($H \approx 3.1GHz/2.7GHz$). The same values are used in the experiments conducted in Sect. 4.

In Fig. 1b, it can be observed that for the frequency ratio $H = 1.14$, there are some regions where the speedup is higher than the upper limit of Amdahl's [Eq. (12)]. For example, when $f = 0.02$ and $p \in [1, 8]$, where p is the number of cores, the speedup is greater than p (the upper limit of Amdahl's law). Speedup allows to study the behavior of a parallel program with respect to a sequential program, if speedup is positive then it follows that the parallel program has better performance than the sequential program and, if there is a deceleration then the sequential program has better performance



(a) Parallel program speedup without frequency variation ($H = 1$).

(b) Parallel program speedup with frequency variation $H = 1.14$.

Fig. 1 Amdahl's law for multicore processors without and with frequency variation

than the parallel program. In the same way, in the case of processors with frequency change, the speedup allows to study the behavior of a parallel program in a processor with frequency variable with respect to the sequential execution in the multicore processor without change of frequencies. For $f = 0.02$, the speedup exceeds the traditional limit of Amdahl's law, given in Eq. 12, because the parallel program is executed in the multicore processor with frequencies greater than of sequential execution, so H-more instructions per unit of time per streaming is executed in the parallel program. It can also be seen that the acceleration is below pH as indicated in Eq. 18.

3.3 Energy consumption model

In Sects. 3.1 and 3.2, we introduced the models for time and power for multicore processors that can change their frequency. In this section, we will combine both models to generate an energy consumption model. Analogously to the modeling of time, we begin by introducing the general formula for energy of a sequential program. In general, the energy required to solve a problem of size n on a single core is denoted by $J(n, 1)$, where $\sigma(n)$ refers to the inherently sequential portion of the computation and $\varphi(n)$ to the potentially parallel portion. Then, $J(n, 1)$ is given by Eq. (19).

$$\begin{aligned} J(n, 1) &= W(1)\sigma(n) + W(1)\varphi(n) \\ &= W(1)(\sigma(n) + \varphi(n)), \end{aligned} \tag{19}$$

where $W(1)$ depends if the processor frequency is variable or not [Eqs. 4 and 5].

We define $J(n, p)$ as the energy to solve a problem of size n on p cores. We have:

$$J(n,p) = W(1)\sigma(n) + W(p)\frac{\varphi(n)}{p} + W_\kappa \kappa(n,p), \tag{20}$$

where $W(1)$ and $W(p)$ could be taken from Eqs. 4 and 5, depending if the processor have activate the variable frequency, and $W_\kappa \kappa(n,p) > 0$, which refers to the energetic overhead inherent in a parallel program.

Now, to model the energy for a multicore processor of variable frequency, we consider two cases. The first one is when the frequency of the processor does not have any variation, and the second case is when the frequency has variation. We denote these two cases by J_{off} and J_{on}, respectively.

Eqs. (21) and (22) correspond to the sequential and parallel executions for which there is a lack of frequency variation in the multicore processor, where $W_{off}(1)$ is given by Eq. (6) and $W_{off}(p)$ by Eq. (4).

$$\begin{aligned} J_{off}(n, 1) &= W_{off}(1)\sigma(n) + W_{off}(1)\varphi(n) \\ &= W_{off}(1)(\sigma(n) + \varphi(n)). \end{aligned} \tag{21}$$

$$J_{off}(n,p) = W_{off}(1)\sigma(n) + W_{off}(p)\frac{\varphi(n)}{p} + W_\kappa \kappa(n,p). \tag{22}$$

Similarly, for the case when frequency change is enabled, Eqs. (23) and (24) give the energy consumption for solving a problem of size n in one and p cores, respectively.

$$\begin{aligned} J_{on}(n,1) &= W_{on}(1)\frac{\sigma(n)}{H} + W_{on}(1)\frac{\varphi(n)}{H} \\ &= \frac{W_{on}(1)}{H}(\sigma(n)+\varphi(n)), \end{aligned} \tag{23}$$

$$J_{on}(n,p) = W_{on}(1)\frac{\sigma(n)}{H} + W_{on}(p)\frac{\varphi(n)}{Hp} + W_{\kappa}\kappa(n,p). \tag{24}$$

3.4 Amdahl's law extension for energetic speedup

Based on the energy consumption model, it is possible to calculate the energetic speedup using Amdahl's law. By knowing the energetic speedup, the energy saving of a multicore processor of variable frequency can also be known. Again, we will work on two cases, $\Psi_{off}(n,p)$, for multicore processors with no variable frequency and $\Psi_{on}(n,p)$ for multicore processors with variable frequency. This representation will allow us to study the energy consumption behavior of a parallel program in multicore processors. When the frequency change is inactive, as in Amdahl's law, the energy consumption of a sequential program $J_{off}(n,1)$ is divided by the power consumption of a parallel program $J_{off}(n,p)$ as given in Eq. (25).

$$\begin{aligned} \Psi_{off}(n,p) &= \frac{J_{off}(n,1)}{J_{off}(n,p)} \\ &= \frac{W_{off}(1)(\sigma(n)+\varphi(n))}{W_{off}(1)\sigma(n) + W_{off}(p)\varphi(n) + W_{\kappa}\kappa(n,p)} \\ &\le \frac{W_{off}(1)(\sigma(n)+\varphi(n))}{W_{off}(1)\sigma(n) + W_{off}(p)\frac{\varphi(n)}{p}} \\ &\le \frac{1}{f + (1-f)\frac{W_{off}(p)}{pW_{off}(1)}}, \end{aligned} \tag{25}$$

where $f = \frac{\sigma(n)}{(\sigma(n)+\varphi(n))}$.

The maximum energy speedup for a parallel program running in a processor without frequency variation is when $\lim_{f\to 0}\Psi_{off}(n,p)$, then

$$\lim_{f\to 0}\Psi_{off}(n,p) = \frac{pW_{off}(1)}{W_{off}(p)}. \tag{26}$$

If this limit is greater than 1, then we are in a scenario where we can have energy savings. If this value is less than or equal to 1, then we are in a scenario where sequential programs consume less energy than parallel programs.

On the other hand, when the frequency change is active, the energy consumption of a sequential execution when the frequency change is inactive $J_{off}(n,1)$ is divided by the energy consumption of the parallel program in a multicore processor with the frequency change activated $J_{on}(n,p)$ as given by Eq. (27).

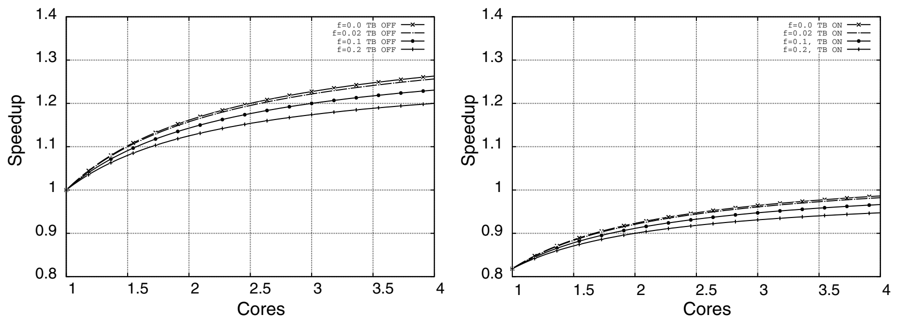
$$\begin{aligned}\Psi_{on}(n,p) &= \frac{J_{off}(n,1)}{J_{on}(n,p)}\\ &= \frac{W_{off}(1)(\sigma(n)+\varphi(n))}{W_{on}(1)\frac{\sigma(n)}{H} + W_{on}(p)\frac{\varphi(n)}{Hp} + W_{\kappa}\kappa(n,p)}\\ &\leq \frac{W_{off}(1)(\sigma(n)+\varphi(n))}{W_{on}(1)\frac{\sigma(n)}{H} + W_{on}(p)\frac{\varphi(n)}{Hp}}\\ &\leq \left(\frac{W_{off}(1)}{W_{on}(1)}\right)\frac{H}{f+(1-f)\frac{W_{on}(p)}{W_{on}(1)p}}.\end{aligned} \tag{27}$$

The maximum energy speedup for a parallel program running in a processor with frequency variation is when $\lim_{f\to 0}\Psi_{off}(n,p)$, then

$$\lim_{f\to 0}\Psi_{on}(n,p) = \frac{pHW_{off}(1)}{W_{on}(p)}. \tag{28}$$

As in the inequality (26), if this limit is greater than 1, we can have energy savings. If this value is less than or equal to 1, then the sequential programs consume less energy than parallel programs. The relevant point is that we can know the maximum energy acceleration if p is equal to the number of cores of the processor, and if we know H, $W_{off}(1)$ and $W_{on}(p)$.

For example, Fig. 2 shows an instance of the model that describes how energy speedup behaves in a processor that can change frequency. In this example we considered an 4-core processor that has its base frequency at 2.75Ghz and can change its frequency to 3.1GHz, thus the value of $H = 1.127$. The power values associated to the processor are: $W_c = 2.5$ watts for the constant power, $W_{aoff} = 6.5$ watts for the active power of the core when Turbo Boost is off, and $W_{aon} = 8.5$ is the active power per core when Turbo Boost is on. With these values it is possible to compute $W_{off}(p) = W_c + pW_{aoff}$ and $W_{on}(p) = W_c + pW_{aon}$. Figure 2a shows the energy speedup when the frequency is constant. Figure 2b shows the energy speedup when there are changes in the frequency. These figures show that the processor without changes in the frequency can have greater speedup than the processor that has changes in the frequency. However, the gain is not so significant, it stays close to 1. This means, parallel programs running in processors with or without change of frequency consume the same energy when they use all cores available. So, from the point of view of the model, a processor with frequency change offers greater performance in parallel programs and uses the same amount of energy as a sequential program.



(a) Energetic speedup without frequency variation ($H = 1$)

(b) Energetic speedup with frequency variation ($H = 1.127$)

Fig. 2 Energy speedup for parallel programs in multicore processor for $J_{off}(n, p)$ and $J_{on}(n, p)$

4 Experiments and discussion

Several experiments have been conducted in two platforms to validate the energy model introduced in Sect. 3.3. These experiments have been executed on three different Intel processors adapted with Turbo boost technology listed below. This technology allows the processor to switch on and off the frequency variation during the processing of instructions.

1. Intel Core i5 Dual core 2.7 Ghz (power book).
2. Intel Core i5 Quad core 2.3Ghz (iMac).
3. Intel Xeon Eight Quad core 3.4Ghz (Server).

To measure the energy consumption in a processor, we used *Power Gadget*, which is an Intel software-based power usage monitoring tool activated for the processors used in these experiments.

To test the proposed model, part of the Linpack benchmark was used, specifically the double-precision matrix multiplication in square matrices. It is well known that the complexity of this task is $O(n^3)$ and that it stresses the processor. Also, the parallel part exceeds 98% of execution time with matrices greater than 1500. The sizes of the matrices that were tested are $1500 \times 1500, 2000 \times 2000, 3000 \times 3000$ and 3500×3500.

OpenMP was used for the tests. The number of threads that were used in the OpenMP programs for the experiments was from one to the number of cores that were on the test platform. That is, for Intel Core i5 dual core processor was experimented with one and two threads. On the Intel Core i5 quad core processor, the experiments were done with one, two, three and four threads. Finally, for Intel Xeon eight-core, the experiments were done from one until eight threads. Special care was taken in that increasing the number of threads would increase the use of cores in the processor. For example, if k threads were used, then k cores of the processor were used. For this reason, we use threads and cores in an indistinct manner in the following sections.

Table 2 Frequency and electric power for Intel Core i5 dual core with Turbo Boost off and on

Turbo Boost	Frequency (Ghz)	W_c (w)	W_a (w)	$W(1)$ (w)	$W(2)$ (w)
Off	2.7	2.5	6.5	9	15.5
On	3.1	2.5	8.5	11	19.5

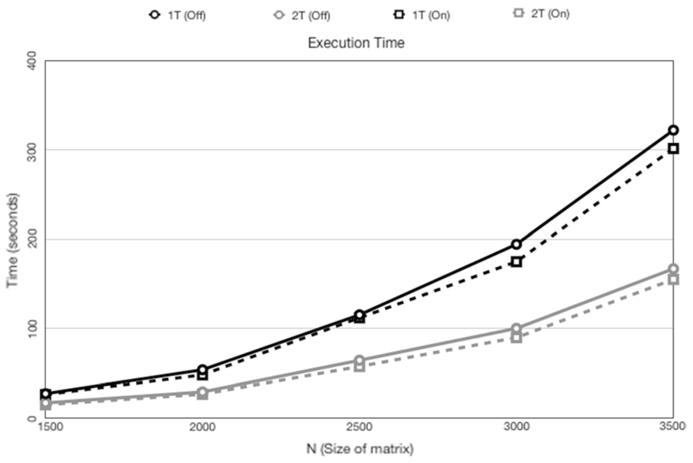


Fig. 3 Double-precision matrix multiplication time performance (dpMM)

4.1 Dual core experiments

Table 2 shows the power behavior of the Intel Core i5 Dual core processor, as measured by the Intel PowerTutor tool. Columns W_c and W_a represent the power constant and the active power for each core [Eqs. (4) and (5)]. Column $W(1)$ is the power of the processor with one active core [Eqs. (6) and (7)], and column $W(2)$ is the power of the processor with two active cores [Eqs. (4) and (5)].

Figure 3 and Table 3 show the execution time behavior and speedup of double-precision matrix multiplication for different sizes of square matrices. In Fig. 3, the solid lines show the consumption time of the program running in one and two cores with Turbo Boost off, while the dashed lines show the behavior of the process in one and two cores with Turbo Boost on. Notice, that when Turbo Boost is activated there are shorter execution times compared to when it is turned off. The process that consumes more time is the sequential one with Turbo Boost off, and the process that consumes less time is the one that uses four cores with Turbo Boost on. This behavior can be deduced from Eqs. (12) and (18), notice that when $H = 3.1/2.7 \approx 1.148 > 1$, the speedup obtained with Turbo Boost on is greater than when Turbo Boost is off.

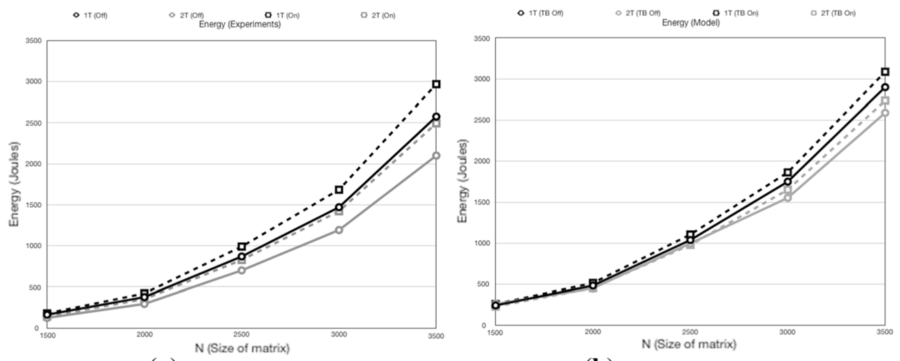
Table 3 shows the experimental speedup and the model associated to Eq. (17), of double-precision matrix multiplication using one and two cores, for the cases when Turbo Boost is on and off. Speedup is calculated with Eq. (27), since we are

Table 3 Speedup on Intel Core i5 dual core processor

Size	f	Turbo Boost Off		Turbo Boost On		
		Experimental two threads	Model two threads	Experi-mental one thread	Experimental two threads	Model two threads
1500	0.001469	1.613564	1.997066	1.037827	1.827173	2.292928
2000	0.001128	1.845377	1.997746	1.118600	2.021029	2.293709
2500	0.000686	1.791813	1.998628	1.029916	2.007168	2.294722
3000	0.000599	1.941497	1.998802	1.111596	2.164335	2.294921
3500	0.000426	1.928455	1.999148	1.068103	2.076377	2.295318

interested in knowing the execution behavior of parallel programs in architectures with changes in the frequency handling of the processor cores. In Table 3, we can observe that the speedup with Turbo Boost on exceeds the traditional linear limit of Amdahl's law [Eq. (12)]. Also, according to the behavior predicted by the model [Eq. (18)], this upper limit is pH, and since $H = 1.148$, the upper limit of the speedup is $2.29x$. This result is consistent with those in column *Model Two Threads* for Turbo Boost On in Table 3, where the maximum speedup reported is 2.2953 for the case of a matrix of size 3500.

Figure 4 shows the energy consumption for double-precision matrix multiplication with different matrix sizes in one and two cores. Figure 4a shows the behavior of energy consumption in the experiments carried out. Figure 4b shows the energy consumption behavior of the model, according to Eqs. (22) and (24), using the processors data found in Table 2 and the sequential time when Turbo Boost is turned off. The solid lines in these graphs correspond to the energy consumption of matrix multiplication when Turbo Boost is off and the dashed lines correspond to the same experiment when Turbo Boost is on. When both graphs in Fig. 4 are compared, it is observed that the behavior predicted by the model corresponds to that of the experiments. Notice that the process consumes more energy when Turbo Boost is



(a) Experimental energy behavior **(b)** Model energy behavior

Fig. 4 Double-precision matrix multiplication (dpMM) energy behavior in Intel Core i5 Dual core

on than when it is off. Also, the execution in two cores consumes less energy than the sequential execution. Another behavior that can be observed from this figure is that the program that consumes most energy is the sequential one with Turbo Boost turned on, while the process that consumes less energy is the one that uses two cores with Turbo Boost off. Finally, in Fig. 4, we can see that the cases of the process running with 1 core with Turbo Boost off and the process that runs with two cores with Turbo Boost on, consume almost the same amount of energy.

Table 4 shows the behavior of energetic speedup for double-precision matrix multiplication when Turbo Boost is on and off. The values on this Table are for experiments with several matrix sizes and for the model given by Eqs. (25) and (27), according to the energy power values found in Table 2. Table 4 shows that for both the experiments and the results of the model, when Turbo Boost is turned off, a greater energy speedup is obtained compared to the speedup obtained when Turbo Boost is on. An interesting behavior observed is, that for both the experiments and the model, when the two cores of the processor are used, the speedup is very close to one. This means that a process using two cores with Turbo Boost on spends the same energy as the sequential process when Turbo Boost is off. Finally, we observe that the speedup that occurs using a core when Turbo Boost is on is less than one, this is because the power consumption of a sequential process with Turbo Boost on spends more energy than a sequential process with Turbo Boost off, as shown in Fig. 4.

4.2 Quad core experiments

Table 5 shows the main attributes of power and frequency of Intel Core i5 quad core processor. Values for the constant power W_c, active power W_a, power of the processor with 1, 2, 3 and 4 active cores are obtained by experiments with the Intel Power Gadget program. Column $W(1)$ is the power of the processor with one

Table 4 Experimental and model energetic speedup for double-precision matrix multiplication on Intel Core i5 dual core processor

Size	One thread	Two threads	Statistical error	One thread	Two threads
	Experimental Turbo Boost Off			*Model Turbo Boost Off*	
1500	1.000000	1.308064	1.40974	1.000000	1.161015
2000	1.000000	1.281765	1.98011	1.000000	1.161079
2500	1.000000	1.241899	4.70266	1.000000	1.161162
3000	1.000000	1.233482	4.81068	1.000000	1.161178
3500	1.000000	1.226390	7.61943	1.000000	1.161211
	Experimental Turbo Boost On			*Model Turbo Boost On*	
1500	0.912849	1.157935	2.91265	0.922314	1.040364
2000	0.894905	1.067509	2.51090	0.922314	1.040409
2500	0.877799	1.049844	4.17080	0.922314	1.040468
3000	0.873333	1.034216	4.93669	0.922314	1.040480
3500	0.867953	1.032789	5.91292	0.922314	1.040503

Table 5 Frequency and electric power for Intel Core i5 quad core with Turbo Boost off and on

Turbo Boost	Frequency (Ghz)	W_c	W_a (w)	$W(1)$	$W(2)$	$W(3)$	$W(4)$
Off	2.7	4.16	6	10.36	13.8	17.20	20.7
On	2.8	4.16	9	13.16	17.1	19.45	22.0

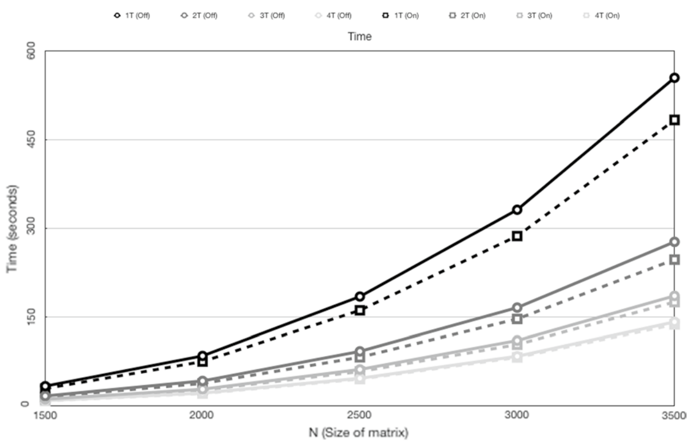


Fig. 5 Double-precision matrix multiplication time performance (dpMM)

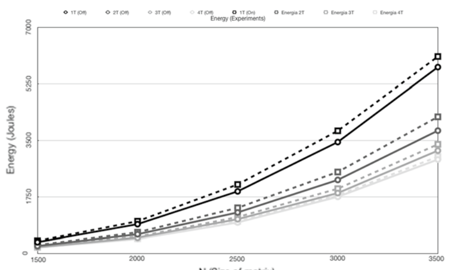
active core [Eqs. (6) and (7)], and column $W(p)$, with $p \in [2, 3, 4]$ is the power of the processor with two, three and four active cores [Eqs. (4) and (5)].

Figure 5 and Table 6 show the execution time behavior of double-precision matrix multiplication for different sizes of square matrices. In Fig. 5, the solid lines show the behavior of the experiment running in one, two, three and four cores with Turbo Boost off, while the dashed lines show the behavior of the process in one, two, three and four cores with Turbo Boost on. In the figure and in the table, it is possible to see that the execution for one, two, three and four cores runs in less time when the frequency is increased. The process that consumes more time is the sequential one with Turbo Boost off, and the process that consumes less time is the one that uses two cores with Turbo Boost on. As in the case of the dual core processor, this behavior is predicted by Eqs. (8), (9), (13) and (14), where $H = 2.8/2.7 \approx 1.037 > 1$.

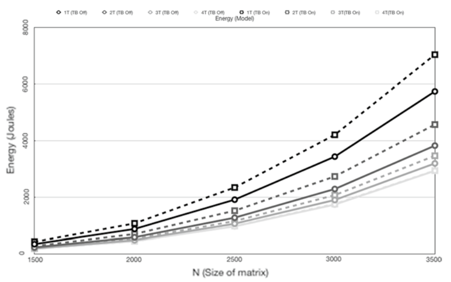
Table 6 shows the experimental speedup and the models speedup, corresponding to Eq. (15). The columns show the speedup with one, two, three and four cores with Turbo Boost on and off for the experimental speedup and the models speedup. As can be seen in the speedup columns for two, three and four cores, the values obtained from the model are very similar to the values obtained by the experiments. As in the case of the two-core processor, the traditional Amdahl's law (10) is fulfilled when Turbo Boost is turned off. But, when Turbo Boost is turned on, the upper limit of Amdahl's law (12) is exceeded, as is indicated by inequality (18). This upper limit is nH, and since $H \approx 1.037$ the upper limit of speedup is $2.074x$ for two cores,

Table 6 Speedup on Intel Core i5 quad core processor

Size	f	One thread	Two threads	Three threads	Four threads	One thread	Two threads	Three threads	Four threads
	Experimental Turbo Boost Off					*Model Turbo Boost Off*			
1500	0.000687	1.0000	1.985900	2.927225	3.708745	1.0000	1.998627	2.995886	3.991772
2000	0.000449	1.0000	1.999850	2.986874	3.859838	1.0000	1.999102	2.997311	3.994619
2500	0.000320	1.0000	1.999228	2.989805	3.943211	1.0000	1.999360	2.998084	3.996163
3000	0.000265	1.0000	1.998340	2.999738	3.945642	1.0000	1.999470	2.998413	3.996822
3500	0.000210	1.0000	1.999796	2.983583	3.910449	1.0000	1.999580	2.998743	3.997481
	Experimental Turbo Boost On					*Model Turbo Boost On*			
1500	0.000687	1.137894	2.188967	3.098018	3.948399	1.037037	2.072650	3.106873	4.139616
2000	0.000449	1.128723	2.224770	3.157378	4.019133	1.037037	2.073143	3.108350	4.142568
2500	0.000320	1.144169	2.241499	3.169859	4.054351	1.037037	2.073410	3.109152	4.144169
3000	0.000265	1.155198	2.259015	3.194838	4.036831	1.037037	2.073524	3.109494	4.144852
3500	0.000210	1.146979	2.242592	3.161721	4.031015	1.037037	2.073638	3.109836	4.145536



(a) Experimental energy behavior



(b) Model energy behavior

Fig. 6 Double-precision matrix multiplication (dpMM) energy behavior on Intel Core i5 Quad core

Table 7 Energetic speedup model and experimental for Intel Core i5 quad core processor

Size	One thread	Two threads	Three threads	Four threads	One thread	Two threads	Three threads	Four threads
	Experimental Turbo Boost Off				*Model Turbo Boost Off*			
1500	1.000000	1.509065	1.792455	1.907389	1.000000	1.500932	1.805976	2.000555
2000	1.000000	1.528925	1.855346	2.016989	1.000000	1.501111	1.806322	2.001032
2500	1.000000	1.516539	1.817077	2.007601	1.000000	1.501208	1.806510	2.001291
3000	1.000000	1.517133	1.848149	1.971367	1.000000	1.501250	1.806590	2.001401
3500	1.000000	1.515949	1.811326	1.982703	1.000000	1.501291	1.806671	2.001511
	Experimental Turbo Boost On				*Model Turbo Boost On*			
1500	0.900746	1.345303	1.684277	1.915193	0.816391	1.256108	1.655955	1.951533
2000	0.909239	1.386067	1.745337	1.968412	0.816391	1.256269	1.656361	1.952180
2500	0.899376	1.361960	1.711827	1.936347	0.816391	1.256357	1.656581	1.952530
3000	0.908324	1.368542	1.722781	1.919630	0.816391	1.256394	1.656674	1.952680
3500	0.946242	1.362532	1.706216	1.924711	0.816391	1.256431	1.656768	1.952830

$3.11x$ for three cores and $4.148x$ for four cores. This behavior can be appreciated from experimental values of speedup that are below this upper limit.

Figure 6 and Table 7 show the behavior of energy consumption and energy speedup of matrix multiplication. Figure 6 shows the energy consumption for double-precision matrix multiplication with different matrix sizes in one, two, three and four cores. Figure 6a shows the behavior of energy consumption from the experiments carried out. Figure 6b shows the energy consumption behavior given by the model, as obtained from Eqs. (22) and (24), with the processors data found in Table 5 and sequential time when Turbo Boost is turned off. The solid lines in these graphs correspond to the energy consumption of matrix multiplication when Turbo Boost is off and the dashed lines correspond to the same experiment when Turbo Boost is on. From Fig. 6a, b it is observed that the behavior predicted by the model corresponds to that of the experiments. As in the case of the dual core processor, in Fig. 6 it is observed that the process consumes more energy when Turbo Boost is on. However, the execution when all cores are used consumes less energy than the sequential execution (1 thread). Moreover, the program that consumes the most energy is the sequential one with Turbo Boost turned on, while the process that consumes less energy is the one that uses all the available cores with Turbo Boost turned off.

Table 7 shows the energetic speedup for double-precision matrix multiplication. These values are for both the experimental and the model cases when Turbo Boost is on and off. The values on this table are for experiments with several matrix sizes and for the model given by Eqs. (27) and (25), according to the energy power values found in Table 5. It should be noted that, although there is an error between the value obtained in the model with respect to the experimental value, the model does show the qualitative behavior of the energy, for the cases in which Turbo Boost is off and on. Unlike the dual core case, where the maximum energy speedup is approximately one, we find that when all the cores of the Quad Core processor are used, we have a speedup close to $2x$, for the cases in which Turbo Boost is on and off. Thus, when we use the multicore processor at its maximum computational capacity and with Turbo Boost on, we save time (more than $4x$ speedup) and spend half of the energy consumed by the sequential program with Turbo Boost off.

4.3 Xeon eight-core experiments

Table 8 shows the main attributes of power and frequency of Intel Xeon eight-core processor. Values for the constant power W_c, power of the processor with 1, 2, 3, 4, 5, 6, 7 and 8 active cores are obtained by experiments with the Intel Power Gadget program. Column $W(1)$ is the power of the processor with one active core [Eqs. (6) and (7)], and columns $W(p)$, with $p \in [2, \dots, 8]$ is the power of the processor with two, three until eight active cores.

For this case, the H element from Eq. 17 is $H = 3.6GHX/3.4GHX = 1.058823$. For visualization purposes, the tables and figures that describe the experimental and model behavior use 1, 2, 4, 6 and 8 threads.

Figure 7 shows the experimental and the models time behavior, corresponding to Eq. (15) for double-precision matrix multiplication. The time behavior is maintained in the model.

Table 9 shows the comparison of the speedup model from (10) and (17) inequalities, with the acceleration obtained from the experiments. The case of Turbo Boost off is the traditional Amdahl's law.

Table 8 Frequency and electric power for Intel Xeon eight-core with Turbo Boost off and on

Turbo Boost	Frequency (Ghz)	W_c	$W(1)$	$W(2)$	$W(3)$	$W(4)$	$W(5)$	$W(6)$	$W(7)$	$W(8)$
Off	3.4	9.1	49.33	56.06	62.15	67	72.1	76.6	79.6	82.1
On	3.6	9.1	68.2	73.5	76.21	81.53	88.5	92.7	97	102.2

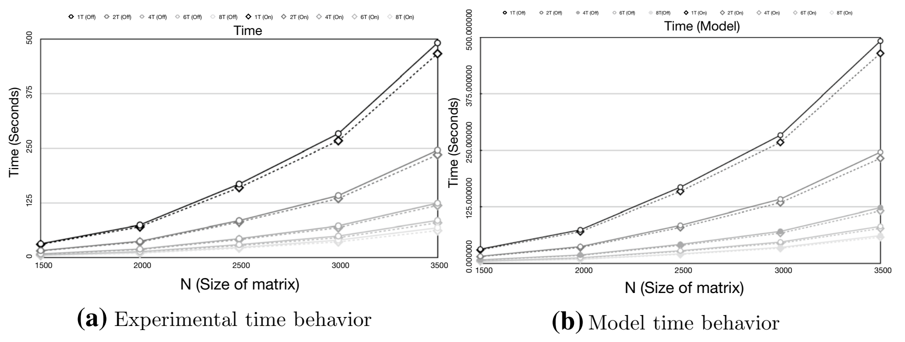


Fig. 7 Double-precision matrix multiplication (dpMM) performance on Intel Xeon eight-core

Figure 8 shows the experimental and the models energy behavior, corresponding to from Eqs. (22) and (24) for double-precision matrix multiplication. The energy consumption behavior is maintained in the model.

Table 10 shows the comparison of the speedup energy model from (25) and (27) inequalities, with the acceleration obtained from the experiments. It is appreciated that while increasing the use of cores, there is more energy savings. The worst performance of power consumption is when using a single core with the Turbo Boost turned on. The maximum energy acceleration is 4.8x and is obtained when using eight cores with the Turbo Boost turned off. The multicore processor saves more energy by using all the cores.

Table 11 shows the values obtained from Verner Model [20]. We notice that the model is equal to our model in the case of Turbo Boost off. And when comparing our model with the Verner model for Turbo Boost on, it is observed that our model is closer to the experiments (Table 10).

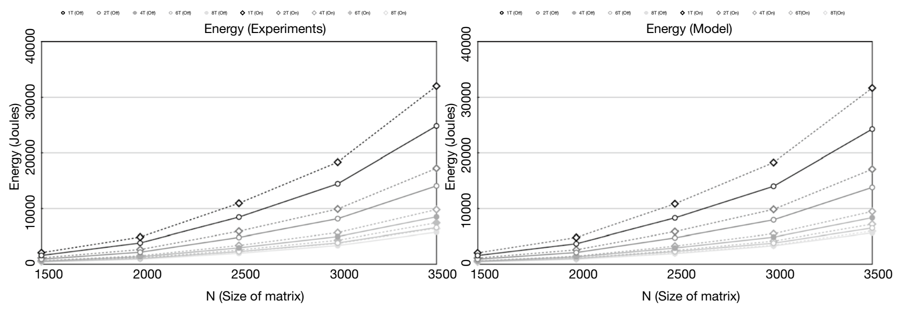
5 Conclusions

It is possible to obtain extensions of Amdahl's law to study the performance and power behavior of parallel programs running on Intel processors with Turbo Boost technology. The model presented in this work can help to understand the behavior of parallel programs on processors with Turbo Boost. When change of frequencies in the processing units is considered, the rate of change in the frequency should be taken into account, as given by Eq. (17). To extend Amdahl's law for the case of energy consumption behavior, it is necessary to establish the power model used by the processors when the frequency change is active and when it is not [Eqs. (4) and (5)]. By combining the model extensions for time and power, it is possible to extend Amdahl's law for energy, by considering constant or variable frequency [Eqs. (25) and (27)].

Several experiments were carried out on platforms with Intel Core i5 processors of two and four cores and Intel Xeon eight-core. These processors allow to enable or disable frequency change through the Turbo Boost technology. The

Table 9 Speedup experimental and model for Intel Xeon eight-core processor

Size	One thread	Two threads	Four threads	Six threads	Eight threads	One thread	Two threads	Four threads	Six threads	Eight threads
	Experimental Turbo Boost Off					*Model Turbo Boost Off*				
1500	1.000000	1.986322	3.792619	5.428659	6.725028	1.000000	1.998627	3.991773	5.981843	7.961712
2000	1.000000	1.991278	3.833580	5.509441	6.929370	1.000000	1.999102	3.994619	5.988949	7.974935
2500	1.000000	1.998067	3.887061	5.700239	7.302066	1.000000	1.999360	3.996164	5.992808	7.982120
3000	1.000000	1.999094	3.921813	5.753170	7.096713	1.000000	1.999470	3.996823	5.994454	7.985187
3500	1.000000	1.999419	3.953518	5.807577	7.204500	1.000000	1.999580	3.997482	5.996102	7.988257
	Model Turbo Boost On					*Model Turbo Boost On*				
1500	1.044967	2.077091	4.074112	6.017063	7.214358	1.058823	2.11619	4.226581	6.333713	8.430044
2000	1.057392	2.090701	4.115944	6.156174	7.454265	1.058823	2.11670	4.229595	6.341237	8.444044
2500	1.051485	2.083223	4.116845	6.187055	7.798846	1.058823	2.11697	4.231230	6.345323	8.451652
3000	1.056104	2.101192	4.146856	6.225864	7.899906	1.058823	2.11708	4.231928	6.347066	8.454900
3500	1.052935	2.086501	4.130702	6.214695	7.955720	1.058823	2.11720	4.232625	6.348811	8.458151



(a) Experimental time behavior **(b)** Model time behavior

Fig. 8 Double-precision matrix multiplication (dpMM) energy behavior on Intel Xeon eight-core

operation considered was double-precision matrix multiplication for different matrix sizes in order to stress the processor. The advantage of this test is that it is highly parallelizable and the values of f given by Eq. (10) are very close to zero, allowing to study the behavior of a highly parallelizable task.

With the experiments, it was possible to verify that the model corresponds to the qualitative behavior in time and energy for parallel programs running in processors that allow frequency change. In addition, the experiments show that the sequential programs are the ones that consume more energy when the processor has frequency change enabled and that the parallel programs that consume less energy are those that occupy the maximum amount of cores available in the processor, in accordance with the model. Parallel programs consume less energy when the processor frequency is not changed compared to parallel programs that run when the processor increases its frequency. From Figs. 4 and 6, it is possible to observe upper and lower limits in the energy consumption, the programs that consume more energy are the sequential ones with the case of Turbo Boost on. The programs that consume less energy are those that have Turbo Boost off and that use the maximum amount of cores available in the processor, however, if they use all of the available cores, the energy difference between the case with Turbo Boost off compared to the case with Turbo Boost on is very small.

As expected, it was validated that at higher frequencies, the processor cores operate with more power. This is noted in Tables 2, 5 and 8. However, the speedup analyses show that the use of more cores decreases the execution time and decreases the energy consumption. In fact, for the case in which all the processor cores are used, the consumption of energy is almost the same with Turbo Boost on and off.

The power model presented in this work considers linear behavior; however, it is useful to deduce the formulas for energy acceleration and it is appreciated in the experiments that it has a good behavior with respect to the experimental results. In the future, a nonlinear correction can be made to the power model to have a better prediction of the energy consumption behavior, taking into account the energy proportionality models.

Table 10 Energetic speedup experimental and model for Intel Xeon eight-core processor

Size	One thread	Two threads	Four threads	Six threads	Eight threads	One thread	Two threads	Four threads	Six threads	Eight threads
	Experimental Turbo Boost Off					*Model Turbo Boost Off*				
1500	1.000000	1.759016	2.815894	3.604298	4.179634	1.000000	1.758982	2.941144	3.856535	4.794283
2000	1.000000	1.772515	2.888767	3.706067	4.300625	1.000000	1.759300	2.942505	3.859160	4.798619
2500	1.000000	1.757830	2.876024	3.731069	4.314419	1.000000	1.759472	2.943243	3.860585	4.800972
3000	1.000000	1.770125	2.918746	3.779421	4.316870	1.000000	1.759546	2.943557	3.861193	4.801977
3500	1.000000	1.765877	2.925379	3.785331	4.338124	1.000000	1.759619	2.943872	3.861800	4.802981
	Experimental Turbo Boost On					*Model Turbo Boost On*				
1500	0.761055	1.401512	2.452704	3.214633	3.643008	0.765861	1.420437	2.55845	3.374129	4.076440
2000	0.784244	1.447997	2.533015	3.357174	3.783814	0.765861	1.420726	2.55988	3.376868	4.080641
2500	0.772071	1.429126	2.511706	3.318781	3.839439	0.765861	1.420883	2.56066	3.378355	4.082922
3000	0.787411	1.458152	2.551198	3.360736	3.891491	0.765861	1.420950	2.56099	3.378989	4.083895
3500	0.775413	1.440657	2.532669	3.345287	3.887327	0.765861	1.421016	2.56132	3.379623	4.084808

Table 11 Energy speedup on Intel Xeon eight-core processor with Verner Model

Size	One thread	Two threads	Four threads	Six threads	Eight threads	One thread	Two threads	Four threads	Six threads	Eight threads
	Turbo Boost Off					*Turbo Boost On*				
1500	1.000000	1.758982	2.941144	3.857920	4.794283	0.944481	1.751778	3.15544	4.161609	5.028006
2000	1.000000	1.759300	2.942505	3.860548	4.798619	0.944468	1.752092	3.15706	4.164745	5.032834
2500	1.000000	1.759472	2.943243	3.861974	4.800972	0.944462	1.752262	3.15794	4.166447	5.035455
3000	1.000000	1.759546	2.943557	3.862582	4.801977	0.944459	1.752335	3.15831	4.167173	5.036573
3500	1.000000	1.759619	2.943872	3.863190	4.802981	0.944456	1.752407	3.15869	4.167899	5.037692

The experimental results validate the model for the case of architectures with Intel Turbo Boost technology. In the future work, tests will be done for AMD's Turbo Core technology and ARM processors.

Acknowledgements The authors thank financial support given by the Mexican National Council of Science and Technology (CONACyT), as well as ABACUS: Laboratory of Applied Mathematics and High-Performance Computing of the Mathematics Department of CINVESTAV-IPN. The authors acknowledge both, the Center for Research and Advance Studies of the National Polytechnic Institute (CINVESTAV-IPN) and the Section of Research and Graduate Studies (SEPI) of ESCOM-IPN, for encouragement and facilities provided to accomplish this publication.

References

1. Clerici A, Assayag M (2013) Recursos energéticos globales. Encuesta 2013: Resumen. Tech. rep., World Energy Council, for sustainable energy
2. Commission WEC (1993) Energy for tomorrow's world: the realities, the real options and the agenda for achievement. St. Martin's Press, New York
3. Nakicenovic N, Jefferson M (1995) Global energy perspectives to 2050 and beyond. Global energy perspectives to 2050 and beyond. Tech. rep
4. Poizot P, Dolhem F (2011) Clean energy new deal for a sustainable world: from non-CO_2 generating energy sources to greener electrochemical storage devices. Energy Environ Sci 4(6):2003–2019
5. Chu S, Majumdar A (2012) Opportunities and challenges for a sustainable energy future. Nature 488(7411):294–303
6. Chow J, Kopp RJ, Portney PR (2003) Energy resources and global development. Science 302(5650):1528–1531
7. Robinson S (2009) Cellphone energy gap: desperately seeking solutions. Strateg Anal
8. D'Andrea R (2014) Guest editorial can drones deliver? IEEE Trans Autom Sci Eng 11(3):647–648
9. Mei Y, Lu YH, Hu YC, Lee CG (2004) Energy-efficient motion planning for mobile robots. In: Proceedings, ICRA'04 2004 IEEE International Conference on Robotics and Automation 2004, vol 5, pp 4344–4349
10. de Santos PG, Garcia E, Ponticelli R, Armada M (2009) Minimizing energy consumption in hexapod robots. Adv Robot 23(6):681
11. Chyba M, Haberkorn T, Singh S, Smith R, Choi S (2009) Increasing underwater vehicle autonomy by reducing energy consumption. Ocean Eng 36(1):62
12. Geller T (2011) Supercomputing's exaflop target. Commun ACM 54(8):16–18
13. Hsu J (2012) Supercomputer 'Titans' face huge energy costs. Blog on LiveScience. https://www.livescience.com/18072-rise-titans-exascale-supercomputers-leap-power-hurdle.html
14. Tarkoma S, Siekkinen M, Lagerspetz E, Xiao Y (2014) Smartphone energy consumption: modeling and optimization. Cambridge University Press, Cambridge
15. Meneses-Viveros A, Hernandez-Rubio E, Mendoza S, Rodriguez J, Quintos ABM (2018) Energy saving strategies in the design of mobile device applications. Sustain Comput Inform Syst 19:86–95
16. Xu Q, Mytkowicz T, Kim NS (2016) Approximate computing: a survey. IEEE Des Test 33(1):8–22
17. Pant YV, Abbas H, Nischal K, Kelkar P, Kumar D, Devietti J, Mangharam R (2015) Power-efficient algorithms for autonomous navigation. In: 2015 International Conference on Complex Systems Engineering (ICCSE), pp 1–6
18. Gunther S, Deval A, Burton T, Kumar R (2010) Energy-efficient computing: power management system on the nehalem family of processors. Intel Technol J 14(3):50–65
19. Meneses-Viveros A, Paredes-López M, Gitler I (2018) In: International Conference on Supercomputing in Mexico, Springer, pp 87–96
20. Verner U, Mendelson A, Schuster A (2017) Extending Amdahl's Law for Multicores with Turbo Boost. IEEE Comput Archit Lett 16(1):30–33
21. Le Sueur E, Heiser G (2010) In: Proceedings of the 2010 International Conference on Power Aware Computing and Systems, USENIX Association, Berkeley, HotPower'10, pp 1–8

22. Haj-Yahya J, Mendelson A, Asher YB, Chattopadhyay A (2018) In: Energy Efficient High Performance Processors, Springer, pp 57–72
23. Conway P, Hughes B (2007) The AMD Opteron Northbridge architecture. IEEE Micro 27(2):10–21. https://doi.org/10.1109/MM.2007.43
24. Rotem E, Naveh A, Ananthakrishnan A, Weissmann E, Rajwan D (2012) Power-management architecture of the intel microarchitecture code-named Sandy bridge. IEEE Micro 32(2):20. https://doi.org/10.1109/MM.2012.12
25. Charles J, Jassi P, Ananth NS (2009) In: Proceedings of IEEE International Symposium on Workload Characterization, 2009. IISWC 2009, IEEE, pp 188–197
26. Fuller SH, Miller LE (2011) The National Academies Press pp 31–38
27. Song W, Mukhopadhyay S, Yalamanchili S (2012) In: Dark Silicon Workshop
28. Cebrian JM, Natvig L, Meyer JC (2012) In: 2012 SC Companion: High Performance Computing, Networking Storage and Analysis, IEEE, pp 675–684
29. Sun XH, Chen Y (2010) Reevaluating Amdahl's Law in the Multicore Era. J Parallel Distrib Comput 70(2):183
30. Woo DH, Lee HHS (2008) Extending Amdahl's law for energy-efficient computing in the Many-Core Era. Computer 41(12):24–31
31. Hill MD, R MM (2008) Amdahl's law in the multicore Era. Computer 41(7):33–38
32. Londoño SM, de Gyvez JP (2010) In: 2010 International Conference on Energy Aware Computing (ICEAC), IEEE, pp 1–4
33. Cho S, Melhem RG (2010) On the interplay of parallelization, program performance, and energy consumption. IEEE Trans Parallel Distrib Syst 21:342–353
34. Isidro-Ramirez R, Viveros AM, Rubio EH (2015) Energy consumption model over parallel programs implemented on multicore architectures. Int J Adv Comput Sci Appl 6(6):21
35. Pei S, Zhang J, Xiong N, Kim MS, Gaudiot JL (2018) Energy efficiency of heterogeneous multicore system based on the enhanced Amdahl's law. IJHPCN 12(3):261–269
36. Hsu CH, Poole SW (2013) In: 2013 42nd International Conference on Parallel Processing, IEEE, pp 834–840
37. Hsu CH, Poole SW (2015) In: Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering, pp 235–240
38. Ruiu P, Fiandrino C, Giaccone P, Bianco A, Kliazovich D, Bouvry P (2017) On the energy-proportionality of data center networks. IEEE Trans Sustain Comput 2(2):197–210
39. Jiang C, Wang Y, Ou D, Luo B, Shi W (2017) In: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), IEEE, pp 1649–1660
40. Malla S, Christensen K (2020) The effect of server energy proportionality on data center power oversubscription. Future Gener Comput Syst 104:119–130
41. Martin AJ (2001) Towards an energy complexity of computation. Inf Process Lett 77(2–4):181–187
42. Tran VNN, Ha PH (2016) In: 2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS), IEEE, pp 1041–1048
43. Roy S, Rudra A, Verma A (2013) In: Proceedings of the 4th conference on Innovations in Theoretical Computer Science, ACM, pp 283–304
44. Swapnoneel R, Rudra A, Verma A (2013) In: 4th Conference on Innovations in Theoretical Computer Science ITCS '13, pp 283–304
45. Basmadjian R, de Meer H (2012) In: Future Energy Systems: Where Energy, Computing and Communications Meet (e-energy), pp 1–10
46. Wu F, Chen J, Dong Y, Zheng W, Pan X, Sun Y (2018) In: 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), IEEE, pp 960–967
47. Amdahl GM (1967) In: AFIPS Conference, vol 30, pp 483–485
48. Kim SH, Kim D, Lee C, Jeong WS, Ro WW, Gaudiot JL (2014) A performance-energy model to evaluate single thread execution acceleration. IEEE Comput Archit Lett 14(2):99–102
49. Acun B, Miller P, Kale LV (2016) In: Proceedings of the 2016 International Conference on Supercomputing, pp 1–12
50. Marathe A, Zhang Y, Blanks G, Kumbhare N, Abdulla G, Rountree B (2017) In: Proceedings of the 5th International Workshop on Energy Efficient Supercomputing, pp 1–8
51. Hackenberg D, Schöne R, Ilsche T, Molka D, Schuchart J, Geyer R (2015) In: 2015 IEEE International Parallel and Distributed Processing Symposium Workshop, IEEE, pp 896–904

52. Wang B, Schmidl D (2015) In International Workshop on OpenMP. Springer, Switzerland, pp 233–246
53. Marques SMV, Medeiros TS, Rossi FD, Luizelli MC, Girardi AG, Beck ACS, Lorenzon AF (2019) In: 2019 IFIP/IEEE 27th International Conference on Very Large Scale Integration (VLSI-SoC), IEEE, pp 149–154
54. Jarus M, Varrette S, Oleksiak A, Bouvry P (2013) In: Revised Selected Papers of the COST IC0804 European Conference on Energy Efficiency in Large Scale Distributed Systems - Volume 8046, Springer, Berlin, EE-LSDS 2013, pp 182–200. https://doi.org/10.1007/978-3-642-40517-4$_$16
55. Kambadur M, Kim MA (2014) In: Proceedings of the 2014 ACM International Conference on Object Oriented Programming Systems Languages and Applications, pp 329–344
56. Porterfield AK, Olivier SL, Bhalachandra S, Prins JF (2013) In: 2013 IEEE International Symposium on Parallel and Distributed Processing, Workshops and PhD Forum, IEEE, pp 884–891
57. Hackenberg D, Oldenburg R, Molka D, Schöne R (2013) In: 2013 International Green Computing Conference Proceedings, IEEE, pp 1–9