

# Finally, how many efficiencies the supercomputers have?

And, what do they measure?

János Végh<sup>1</sup>

Published online: 27 February 2020 © The Author(s) 2020

# Abstract

Using extremely large number of processing elements in computing systems leads to unexpected phenomena, such as different efficiencies of the same system for different tasks, that cannot be explained in the frame of the classical computing paradigm. The introduced simple non-technical model enables to set up a frame and formalism needed to explain the unexpected experiences around supercomputing. The paper shows that the degradation of the efficiency of the parallelized sequential system is a natural consequence of the computing paradigm, rather than an engineering imperfectness. The workload is greatly responsible for wasting the energy as well as limiting the size and the type of tasks the supercomputers can run. Case studies provide insight how different contributions compete for dominating the resulting payload performance of the computing system and how enhancing the technology made the computing + communication the dominating contribution in defining the efficiency of supercomputers. The model also enables to derive predictions about the supercomputer performance limitations for the near future and provides hints for enhancing the supercomputer components. The phenomena show interesting parallels with the phenomena experienced in science more than a century ago, and through their studying, a modern science was developed.

**Keywords** Supercomputer performance · Parallelized sequential processing · Efficiency of supercomputers · Limitations of parallel processing · Behavior of extreme-scale systems

<sup>☑</sup> János Végh Vegh.Janos@gmail.com

<sup>&</sup>lt;sup>1</sup> Kalimános BT, Debrecen, Hungary

# 1 Introduction

After that the dynamic growing of the single-processor performance has stalled about 2 decades ago [1], the only way to achieve the required high computing performance remained parallelizing the work of a very large number of sequentially working single processors. However, as was very early predicted [2] and decades later experimentally confirmed [3], the scaling of the parallelized computing is not linear. Even, "there comes a point when using more processors ...actually increases the execution time rather than reducing it" [3]. The parallelized sequential processing has different rules of game [3, 4]: the performance gain ("the speedup") has its inherent bounds [5].

Akin to as the laws of science limit the performance of the single-thread processors [6], the commonly used computing paradigm (and its technical implementation) limits the performance of supercomputers [4]. On the one side, experts expected the performance<sup>1</sup> to achieve the magic 1 Eflops around year 2020, see Fig. 1 in [7].<sup>2</sup> The authors noticed that "the performance increase in the No. 1 systems slowed down around 2013, and it was the same for the sum performance," but they extrapolated linearly expecting that the development continues and the "zettascale computing" (i.e., 10<sup>4</sup> times more than the present performance) will be achieved in just more than a decade. On the other side, it has recently been admitted that linearity is "A trend that can't go on ad infinitum" as well as that it "can be seen in our current situation where the historical 10-year cadence between the attainment of megaflops, teraflops, and petaflops has not been the case for exaflops" [8].

The expectations against supercomputers are excessive. For example, as the name of the company PEZY<sup>3</sup> witnesses, a billion times increase in the payload performance is expected. It looks like that in the feasibility studies of supercomputing using parallelized sequential systems an analysis whether building computers of such size is feasible (and reasonable) remained out of sight either in the USA [9, 10] or in EU [11] or in Japan [12] or in China [7]. The "gold rush" is going on, even in the most prestigious journals [10, 13]. In addition to the previously existing "*two different efficiencies of supercomputers*" [14], further efficiency/performance values appeared<sup>4</sup> (of course with higher numeric figures) and several more can easily be derived.

Although severe counter-arguments are also published, mainly on the basis of the power consumption of both the single processors and the large computing centers [15], the moon shot of the limitless parallel processing performance is followed. The probable sources of the idea of having limitless parallelization performance are the

<sup>&</sup>lt;sup>1</sup> There are some doubts about the definition of exaflops, whether it means  $R_{\text{Peak}}$  or  $R_{\text{Max}}$ , in the former case whether it includes accelerator cores, and in the latter case measured by which benchmark and finally using what operand length. Here the term is used as  $R_{\text{Max}}^{\text{HPL}}$ , using 64-bit floating operands.

<sup>&</sup>lt;sup>2</sup> A special issue https://link.springer.com/journal/11714/19/10.

<sup>&</sup>lt;sup>3</sup> https://en.wikipedia.org/wiki/PEZY\_Computings: The name PEZY is an acronym derived from the Greek derived metric prefixes peta, eta, zetta, yotta.

<sup>&</sup>lt;sup>4</sup> https://blogs.nvidia.com/blog/2019/06/17/hpc-ai-performance-record-summit/ and https://www.olcf. ornl.gov/2018/06/08/genomics-code-exceeds-exaops-on-summit-supercomputer/.

ideas published in [3, 16]. The former<sup>5</sup> is based simply on a misinterpretation [17, 18] of the terms in Amdahl's law. In the latter, it might be misunderstood that "*the serial fraction ...is a diminishing function of the problem size*." It was valid at the time of writing, but because the weight of the housekeeping activity grows linearly with the number of the cores (and so does the idle time of the cores), at the today's large number of cores the function turns from diminishing to dominating. In reality, Amdahl's Law (in its original spirit) is valid for all parallelized sequential activities, including computing unrelated ones, and it is the governing law of supercomputing.

Demonstrative failures of some systems (such as the supercomputers Gyoukou and Aurora,<sup>6</sup> or the brain simulator SpiNNaker<sup>7</sup>) are already known, and many more (all they are expected to deliver 0.13-0.2 Eflops) are expected to follow: such as Aurora'21 [21], the China supercomputers<sup>8</sup> and the EU planned supercomputers.<sup>9</sup>

Similar is the case with the exascale applications, such as brain simulation. Exaggerate news about simulating brain of some animals or a large percentage of human brain appeared. The reality is that the many-thread version of the brain simulator can fill extremely large amount of memory with data of billions of artificial neurons [22], a purpose-built brain simulator can be designed to simulate one billion neurons [23], but in practice they both can simulate only about 80 thousand neurons [24], mainly because of "the quantal nature of the computing time" [25].

The confusion is growing. The paper attempts to clear up the terms through scrutinizing the basic terms, contributions, measurement methods. In Sect. 2 a, by intention strongly simplified non-technical model is presented. The notations for Amdahl's law that form the basis of the present paper are introduced in Sect. 3. It is shown that *the degradation of the efficiency of the parallelized sequential systems is a natural consequence of the computing paradigm, rather than an engineering imperfectness*. Furthermore, its consequence is that the parallelized sequential computing systems by their very nature have an upper performance bound. In Sect. 4, the form of the correction for adding performances, stemming out from Amdahl's law, enables to introduce an interesting parallel with the correction term introduced by the relativistic physics for adding speeds. Interestingly enough, under extreme conditions, the technical objects of computing show up a series of similar behavior (for more details see [4]) to that of natural objects.

Given that the race to produce computing systems having components and systems with higher performance numbers is going on, in Sect. 7 the expected results of the developments in the near future are predicted. The section introduces some further performance merits and through interpreting them concludes that increasing the

<sup>&</sup>lt;sup>5</sup> The related work and speedup deserved the Gordon Bell Prize.

<sup>&</sup>lt;sup>6</sup> It was also learned that *specific processor design is needed for exascale*: as part of the announcement the development line *Knights Hill* [19] was canceled and instead be replaced by a "new platform and new microarchitecture specifically designed for exascale."

<sup>&</sup>lt;sup>7</sup> Despite its failure, the SpiNNaker2 is also under construction [20].

<sup>&</sup>lt;sup>8</sup> https://www.scmp.com/tech/policy/article/3015997/china-has-decided-not-fan-flames-super-computing-rivalry-amid-us.

<sup>&</sup>lt;sup>9</sup> https://ec.europa.eu/newsroom/dae/document.cfm?doc\_id=60156.

size of supercomputers further and making expensive enhancements in their technologies only increase the *non-payload performance*.

# 2 A non-technical model of parallelized sequential operation

The performance measurements are simple time measurements<sup>10</sup> (although they need careful handling and proper interpretation, see good textbooks such as [26]): a standardized set of machine instructions is executed (a large number of times) and the known number of operations is divided by the measurement time, for both the single-processor and the distributed parallelized sequential systems. In the latter case, however, the joint work must also be organized, implemented with extra machine instructions and extra execution time, forming an overhead.<sup>11</sup> This is the origin of the efficiency: one of the processors orchestrates the joint operation; the others are waiting. At this point, the "*dark performance*" appears: the processing units are ready to operate, consume power, but do not make any payload work.

Amdahl listed [2] different reasons why losses in the "computational load" can occur. Amdahl's idea enables to *put everything that cannot be parallelized, i.e., distributed between the fellow processing units, into the sequential-only fraction*. For describing the parallel operation of sequentially working units, the model depicted in Fig. 1 was prepared. The technical implementations of different parallelization methods show up virtually infinite variety [27], so here a (by intention) strongly simplified model is presented. The model is general enough, however, to discuss qualitatively some examples of parallel working systems, neglecting different contributions as possible in different cases. The model can easily be converted to a technical (quantitative) one via interpreting the contributions in technical terms, although with some obvious limitations.

The non-parallelizable (i.e., apparently sequential) part comprises contributions from hardware (HW), operating system (OS), software (SW) and Propagation Delay (PD), and also some access time is needed for reaching the parallelized system. This separation is rather conceptual than strict, although dedicated measurements can reveal their role, at least approximately. Some features can be implemented in either SW or HW or shared between them, and also some apparently sequential activities may happen partly parallel with each other. The relative weights of different contributions are very different for different parallelized systems and even within those cases depend on many specific factors, so in every single parallelization case a careful analysis is required. The SW activity represents what was assumed by Amdahl as the total sequential fraction.<sup>12</sup> The non-determinism of the modern HW systems [28,

<sup>&</sup>lt;sup>10</sup> Sometimes also secondary merits, such as GFlops/Watt or GFlops/USD, are also derived.

<sup>&</sup>lt;sup>11</sup> This aspect is neglected in the weak and strong scaling approximations.

<sup>&</sup>lt;sup>12</sup> Although some OS activity was surely included, Amdahl assumed some 20% SW fraction, so at that time the other contributions could be neglected compared to SW contribution. As shown in Fig. 1 and discussed below, for today this contribution is by several orders of magnitude smaller. However, at the same time the number of the cores became several orders of magnitude larger.

29] also contributes to the non-parallelizable portion of the task: the resulting execution time of the parallel working processing elements is defined by the slowest unit.

Notice that our model assumes no interaction between the processes running on the parallelized systems in addition to the absolutely necessary minimum: starting and terminating the otherwise independent processes, which take parameters at the beginning and return result at the end. It can, however, be trivially extended to the more general case when processes must share some resource (like a database, which shall provide different records for different processes), either implicitly or explicitly. The concurrent objects have their inherent sequentiality [30], and the synchronization and communication between those objects considerably increase [31] the non-parallelizable portion (i.e., contribution to  $(1 - \alpha_{\text{eff}}^{\text{SW}})$  or  $(1 - \alpha_{\text{eff}}^{\text{OS}})$ ), so in the case of extremely large number of processors special attention must be devoted to their role on the efficiency of the application on the parallelized system.

The physical size of the computing system also matters: the processor, connected to the first one with a cable of length of dozens of meters, must spend several hundreds clock cycles with waiting, only because of the finite speed of propagation of light, topped by the latency time and hoppings of the interconnection (not mentioning geographically distributed computer systems, such as some clouds, connected through general-purpose networks). Detailed calculations are given in [32].

After reaching a certain number of processors, there is no more increase in the payload fraction when adding more processors: the first fellow processor already finished the task and is idle waiting, while the last one is still idle waiting for the start command. This limiting number can be increased by organizing the processors into clusters: then the first computer must speak *directly* only to the head of the cluster. Another way is to distribute the job near to the processing units, either inside the processor [33] or using processors to let do the job by the processing units of a GPGPU.

This looping contribution is not considerable (and so: not noticeable) at low number of processing units, but can be a dominating factor at high number of processing units. This "high number" was a few dozens at the time of writing the paper [3]; today it is a few millions. Considering the effect of the looping contribution is the border between the first- and second-order approximations in modeling the performance: the housekeeping keeps growing with the growing number of processors, while the resulting performance does not increase any more. The first-order approximation considers the contribution of the housekeeping as constant, while the second-order approximation considers also that as the number of processing units grows, the housekeeping grows with and gradually becomes the dominating factor of the performance limitation and leads to a decrease in the payload performance.

As Fig. 1 shows, in the parallel operating mode (in addition to the calculation, furthermore the communication of data between the processing units) *both the software and the hardware contribute to the execution time*, i.e., both must be considered in Amdahl's Law. This is not new, again: see [2].

Figure 1 also shows where is place to improve the efficiency. When combining properly the PD with the sequential scheduling, the non-payload time can be considerably reduced during fine-tuning the system (in the case of Sierra 0% increase in the number of cores resulted in 32% more performance, in the case of Summit a



**Fig. 1** A non-technical, simplified model of parallelized sequential computing operations. The contributions of the model component *XXX* to  $\alpha$  (sometimes used as  $\alpha_{\text{eff}}$  to emphasize that it is an effective, empirical value) will be denoted by  $\alpha_{\text{eff}}^{XXX}$  in the text. Notice the different nature of those contributions. They have only one common feature: *they all consume time*. The vertical scale displays the actual activity for processing units shown on the horizontal scale

5% increase in the number of cores resulted in 17% more performance). Also, mismatching the total time and the extended measurement time (or not making a proper correction) may lead to completely wrong conclusions [34] as discussed in [32].

#### 3 Amdahl's law in terms of the model

Usually, Amdahl's law is expressed as

$$S^{-1} = (1 - \alpha) + \alpha / N$$
 (1)

where *N* is the number of parallelized code fragments,  $\alpha$  is the ratio of the parallelizable fraction to the total, *S* is the measurable speedup. From this

$$\alpha = \frac{N}{N-1} \frac{S-1}{S} \tag{2}$$

When calculating speedup, one actually calculates

$$S = \frac{(1-\alpha) + \alpha}{(1-\alpha) + \alpha/N} = \frac{N}{N \cdot (1-\alpha) + \alpha}$$
(3)

hence the efficiency (see Fig. 2)

🖄 Springer

$$\underbrace{E(N,\alpha)}_{E(N,\alpha)} = \frac{S}{N} = \left[\frac{1}{N \cdot (1-\alpha) + \alpha}\right] = \frac{R_{Max}}{R_{Peak}}$$
(4)

The phenomenon itself is known since decades [3] and  $\alpha$  is theoretically established [35]. Presently, however, the theory was somewhat faded mainly due to the quick development of the parallelization technology and the increase in the single-processor performance.

During the past quarter of century, the proportion of the contributions changed considerably: today the number of processors is thousands of times higher than it was a quarter of century ago. The growing physical size and the higher processing speed increased the role of the propagation overhead, furthermore the large number of processing units strongly amplified the role of the looping overhead. As a result of the technical development, *the phenomenon on the performance limitation returned in a technically different form at much higher number of processors*.

Through using Eq. (4),  $E = \frac{S}{N} = \frac{R_{\text{Max}}}{R_{\text{Peak}}}$  can be equally good for describing the efficiency of the parallelization of a setup:

$$\alpha_{E,N} = \frac{E \cdot N - 1}{E \cdot (N - 1)} \tag{5}$$

As will be discussed below, with the exception of extremely high number of processors, it can be assumed that  $\alpha$  is independent from the number of processors. Equation (5) can be used to derive quickly the value of  $\alpha$  from the values of parameters  $R_{\text{Max}}/R_{\text{Peak}}$  and the number of cores N.

According to Eq. (4), the efficiency can be described with a 2-dimensional surface, as shown in Fig. 2. On the surface some measured efficiencies of the present top supercomputers are also depicted, just to illustrate some general rules. The HPL<sup>13</sup> efficiencies are sitting on the surface, while the corresponding HPCG<sup>14</sup> values are much below those values. As Fig. 2 witnesses, *Taihulight* and *K* computer stand out from the "millions core" middle group. Thanks to its 0.3M cores, *K* computer has the best efficiency for the *HPCG* benchmark, while *Taihulight* with its 10M cores the lowest one. The middle group follows the rules. For *HPL* benchmark: the more the cores, the lower the efficiency. For *HPCG* benchmark: the "roofline" [36] of that communication intensity reached; they have about the same efficiency.

The projections to the axes show that the top few supercomputers show up very similar parallelization efficiency and core number values: they are both required to receive one of the top slots, see also Fig. 3. The *Taihulight* is an exception on both axes: it has the highest number of cores and the best parallelization efficiency. Its secret is in the processor comprising cooperating cores [33], i.e., it uses a (slightly) different computing paradigm. See also Fig. 6: this trick enables to enhance the effective parallelization value below the limiting line (the other exception below the line is due to using shorter operands). Reducing the

<sup>13</sup> http://www.netlib.org/benchmark/hpl/.

<sup>&</sup>lt;sup>14</sup> https://www.epcc.ed.ac.uk/blog/2015/07/30/hpcg.

Dependence of  $E_{HPL}$  and  $E_{HPCG}$  on  $(1 - \alpha_{eff}^{HPL})$  and N

TOP5'2018.11

Taihulight

K computer

 $10^8 10^7 10^6 10^5$ No of cores

Tianhe-2

Summit .

п Sierra

~



loop count by internal clustering and exchanging data without using the global memory, however, work only for the HPL case, where the contribution of the SW is low. The poor value of  $(1 - \alpha_{\text{eff}}^{\text{HPCG}})$  is not necessarily a sign of architectural weakness [9]: it comprises about 4 times more cores than the new world recorder Summit.

According to Eq. (4), the efficiency can be interpreted in terms of  $\alpha$  and N and the efficiency of a parallelized sequential computing system can be calculated as

$$P(N,\alpha) = \frac{N \cdot P_{\text{single}}}{N \cdot (1-\alpha) + \alpha}$$
(6)

This simple formula explains why the payload performance is not a linear function of the nominal performance and why in the case of very good parallelization  $((1 - \alpha) \ll 1)$  and low N this non-linearity cannot be noticed.

The value of  $\alpha$ , however, can hardly be calculated for the present complex HW/ SW systems from their technical data. There are two ways that can be followed to estimate their value of  $\alpha$ . One way is to calculate  $\alpha$  for the existing supercomputing systems using Eq. (5) to the data in the TOP500 list [4]. This provides a lower bound for  $(1 - \alpha)$  that is already achieved. Another way round is to consider different contributions, see Sect. 5, and to calculate the high limit the value of  $(1 - \alpha)$  that those contributions alone do not enable to exceed (provided that that contribution is the dominant one). It gives us a good confidence on the reliability of the parameters that the values derived in these two ways differ only within a factor of two. At the same time, this also means that the technology is already very close to its theoretical limitations.

Notice that the "algorithmic effects"-like dealing with sparse data structures (which affects cache behavior, that will have growing importance in the age of "real-time everything") or communication between the parallel running threads, such as returning results repeatedly to the main thread in an iteration (which greatly increases the non-parallelizable fraction in the main thread)-manifest through the HW/SW architecture, and they can hardly be separated. Also notice



Fig. 3 Interrelation of ranking (by the benchmark HPL), the number of processors and the efficiency of parallelization. The data are taken from database TOP500 [37]. The correlation is drawn for both the TOP10 and TOP50 supercomputers, respectively

that there are one-time and fixed-size contributions, such as utilizing time measurement facilities or calling system services. Since  $\alpha_{eff}$  is a *relative* merit, the *absolute* measurement time shall be long. When utilizing efficiency data from measurements which were dedicated to some other goal, a proper caution must be exercised with the interpretation and accuracy of the data.

The "right efficiency metric" [38] has always been a question (for a summary see the cited references in [39]) when defining the efficient supercomputing. The goal of the present discussion is to find out the inherent limitations of the parallelized sequential computing and providing numerical values for it. For this goal, the "classic interpretation" [2, 3, 35] of the performance was used, in its original spirit. The contributions mentioned in those papers were scrutinized and their importance under the current technical conditions was revised.

The left subfigure of Fig. 3 shows that to get better ranking on the TOP500 list, higher number of processors is required. The regression line is different for the TOP10 and the TOP50 positions. The cut line between the "racing" and "commodity" supercomputers is around position 10. As the right subfigure underpins, *the high number of processors must be accompanied with good parallelization efficiency*; otherwise, the large number of cores cannot counterbalance the decreasing efficiency, see Eq. (6).

### 4 Analogies with the case of modern versus classic science

Equation (6) simply tells that (in the first-order approximation) the speedup in a parallelized computing system cannot exceed  $1/(1 - \alpha)$ , a wellknown consequence of Amdahl's statement. Due to this, the computing performance cannot be increased above the performance defined by the single-processor performance, the parallelization technology, and the number of processors (this is similar to affirming that the speed of an object cannot exceed the speed of light, independently from the technology used to accelerate it). Exceeding a certain computing performance (using the classic paradigm and its classic implementation) is prohibited by the laws of nature.

The above analogy discovers a parallel with the modern physics, namely with the relativistic speed addition, see Table 1 and Fig. 4. In the columns of Table 1, parallel terms are listed, from the modern science in the left column and from the modern (Amdahl-aware, or parallellized sequential processing aware) computing in the right column. The focus is on that the correction terms in the denominators in both cases force a "speed limit"; in the case of computing this factor is defined by Eq. (4). Figure 4 compares the limiting effect of the relativistic correction to adding speeds in "modern science" and that of the Amdahl-aware correction to adding performances in computing. The functional form of the limitation is of course different, but it is common in the two cases that *the correction term in not noticeable under "classic" conditions, but will dominate the effect when reaching extreme conditions.* 

One more analogy is introduced in Sect. 7; in that case with quantum theory, another is mentioned in Sect. 5.4 and several more are discussed in [4]. It seems to be an interesting parallel that both the nature and the extremely cutting-edge technical (computing) systems show up some extraordinary behavior, i.e., *the linear behavior experienced under normal conditions gets strongly nonlinear under extreme conditions*. That makes in computing *the validity of linear extrapolations as well as linear addition of performances at least questionable* at high-performance values.

The analogies do not want to imply direct correspondence between certain physical and computing phenomena. Rather, *the paper draws the attention to both that under extreme conditions qualitatively different behavior may be encountered and that scrutinizing certain, formerly unnoticed or neglected aspects enables to explain the new phenomena*. Unlike in the nature, in the computing the technical implementation of the critical points can be changed, and through this the behavior of the computing systems can also be changed.

# 5 The effect of different contributions to $\alpha$

The theory can display data from systems with any contributor with any parameter, but from the measured data only the sum of all contributions can be concluded, although dedicated measurements can reveal the value of the contributions experimentally. The publicly available data enable to draw only conclusions of limited validity.

# 5.1 Estimating different limiting factors of $\alpha$

The estimations below assume that the actual contribution is the dominating one and as such defines the achievable performance alone. This is usually not the case in practice, but this approach enables to find out the limiting  $(1 - \alpha)$  values for all contributions.

Table 1Analogy of addingspeeds in physics and addingperformances in computing,in the classic and modernparadigm, respectively

Physics	Computing
Adding of speeds	Adding of performance
Classic	Classic
$v(t) = t \cdot a$	$Perf_{total}(N) = N \cdot P_{single}$
t = time	N = number of cores
a = acceleration	$P_{\text{single}} = Single - performance$
n = optica density	Communication
c = light speed	$\alpha = \text{parallelism}$
Modern (relativistic)	Modern (Amdahl-aware) [4], see Eq. (4)
$v(t) = \underbrace{\frac{t \cdot a}{\sqrt{1 + \left(\frac{t \cdot a}{c/n}\right)^2}}}$	$P(N) = \frac{N \cdot P_{single}}{N \cdot (1 - \alpha) + \alpha}$

In both cases, a correction term is introduced that provides noticeable effect only at extremely large values

Relativistic speed of body accelerated by 'g'





**Fig. 4** Analogy between how the speed of light changes the linearity of speed dependence at extremely large speed values and that how the value of performance changes the linearity of performance dependence at extremely large performance values

In the systems implemented in the Single-Processor Approach (SPA) [2] as parallelized sequential systems, the life begins in one such sequential subsystem. In the large parallelized applications running on general-purpose supercomputers, initially and finally only one thread exists, i.e., the minimal absolutely necessary non-parallelizable activity is to fork the other threads and join them again. With the present technology, no such actions can be shorter than one processor clock period.<sup>15</sup> That is, the theoretical absolute minimum value of the non-parallelizable portion of the task will be given as the ratio of the time of the two clock periods to the total execution time. The latter time is a free parameter in describing the efficiency, i.e., the value of the effective parallelization  $\alpha_{\text{eff}}$  also depends on the total benchmarking time (and so does the achievable parallelization gain, too).

This dependence is of course well known for supercomputer scientists: for measuring the efficiency with better accuracy (and also for producing better  $\alpha_{eff}$  values) hours of execution times are used in practice. In the case of benchmarking, the supercomputer *Taihulight* [40] 13,298 seconds HPL benchmark runtime was used; on the 1.45-GHz processors it means  $2 \times 10^{13}$  clock periods. The inherent limit of  $(1 - \alpha_{eff})$  at such benchmarking time is  $10^{-13}$  (or equivalently the achievable performance gain is  $10^{13}$ ). In the followings for simplicity 1.00 GHz processors (i.e., 1 ns clock cycle time) will be assumed.

The supercomputers are also distributed systems. In a stadium-sized supercomputer, the distance between processors (cable length) up to about 100 m can be assumed. The net signal round trip time is ca.  $10^{-6}$  s, or  $10^3$  clock periods, i.e., in the case of a finite-sized supercomputer the performance gain cannot be above  $10^{10}$ , only because of the physical size of the supercomputer. The presently available network interfaces have 100...200 ns latency times, and sending a message between processors takes the time in the same order of magnitude. This also means that *making better interconnection is not really a bottleneck in enhancing performance*. This statement is underpinned also by discussion in Sect. 5.3.

These predictions enable to assume that the presently achieved value of  $(1 - \alpha_{\text{eff}})$  persists also for roughly hundred times more cores. However, another major issue arises from the computing principle SPA: only one processor at a time can be addressed by the first one. As a consequence, minimum as many clock cycles are to be used for organizing the parallel work as many addressing steps required. Basically, this number equals to the number of cores in the supercomputer, i.e., the addressing in the TOP10 positions typically needs clock cycles in the order of  $5 \times 10^5 \dots 10^7$ , degrading the value of  $(1 - \alpha_{eff})$  into the range  $10^{-6}...2 \times 10^{-5}$ . Two tricks may be used to mitigate the number of the addressing steps: either the cores are organized into *clusters* as many supercomputer builders do, or at the other end the processor itself can take over the responsibility of addressing its cores [33]. Depending on the actual construction, the reducing factor of clustering of those types can be in the range  $10^1...5 \times 10^3$ , i.e., the resulting value of  $(1 - \alpha_{eff})$  is expected to be around  $10^{-7}$ . Notice that deploying "cooperative computing" [33] enhances further the value of  $(1 - \alpha_{eff})$ , but it means already utilizing a (slightly) different computing paradigm: the cores have a kind of direct connection and can communicate with the exclusion of the main memory.

An operating system must also be used, for protection and convenience. If the fork/join is executed by the OS as usual, because of the needed context switchings

<sup>&</sup>lt;sup>15</sup> Taking these two clock periods as an ideal (but not realistic) case, the actual limitation will surely be (much) worse than the one calculated for this idealistic case. The actual number of clock periods depends on many factors, as discussed below.

 $2 \times 10^4$  [41, 42] clock cycles are needed rather than the 2 clock cycles considered in the idealistic case. *The derived values are correspondingly by 4 orders of magnitude different*, that is, the absolute limit is cca.  $5 \times 10^{-8}$ , on a zero-sized supercomputer. This value is somewhat better than the limiting value derived above, but it is close to that value and surely represents a considerable contribution. This is why *Taihulight* runs the actual computations in kernel mode [33].

However, this optimistic limit assumes that the instruction can be accessed in one clock cycle. It is usually not the case: even the cached instruction in the memory needs about 5 times more access time, and the time needed to access the "far" memory is roughly 100 times longer. Correspondingly, the most optimistic achievable performance gain values shall be scaled down by a factor 5...100. A considerable part of the difference between the efficiencies  $\alpha_{\text{eff}}^{\text{HPL}}$  and  $\alpha_{\text{eff}}^{\text{HPCG}}$  can be attributed to different cache behaviors because of the "sparse" matrix operations.

#### 5.2 The effect of workflow

Overly complex Fig. 5 attempts to explain the phenomenon, why and how the performance of a given supercomputer configuration depends on the application it runs.

The non-parallelizable fraction (denoted on the figure by  $\alpha_{\text{eff}}^X$ ) of the computing task comprises components *X* of different origins. As already discussed, and was noticed decades ago, "the inherent communication-to-computation ratio in a parallel application is one of the important determinants of its performance on any architecture" [3], suggesting that the communication can be a dominant contribution in system's performance. Figure 5a displays the case of the minimum communication and Fig. 5b the moderately increased one (corresponding to real-life supercomputer tasks). As the nominal performance increases linearly and the performance decreases exponentially with the number of cores, at some critical value where an inflection point occurs, the resulting performance starts to decrease. The resulting large non-parallelizable fraction strongly decreases the efficacy (or in other words: the performance gain or speedup) of the system [43, 44]. The effect was noticed early [3], under different technical conditions but somewhat faded due to the successes of the development of the parallelization technology.

Figure 5a illustrates the behavior measured with the HPL benchmark. The looping contribution becomes remarkable around 0.1 Eflops and breaks down the payload performance (see also Fig. 1 in [3]) when approaching 1 Eflops. In Fig. 5b, the behavior measured with the benchmark HPCG is displayed. In this case the contribution of the application (brown line) is much higher, the looping contribution (thin green line) is the same as above. Consequently, the achievable payload performance is lower and also the breakdown of the performance is softer.

Given that no dedicated measurements exist, it is hard to make direct comparison between the theoretical prediction and the measured data. However, the impressive and quick development of the interconnecting technologies provides a helping hand.

### 5.3 The contribution of the interconnection

As discussed above, in a somewhat simplified view, the resulting performance can be calculated using the contributions to  $\alpha$  as

$$P(N,\alpha) = \frac{N \cdot P_{\text{single}}}{N \cdot \left(1 - \alpha_{\text{Net}} - \alpha_{\text{Compute}} - \alpha_{\text{Others}}\right) + \approx 1}$$
(7)

That is, two of the contributions are handled with emphasis. The theory easily provides values for the contributions for the interconnection and calculation separately. Fortunately, the public database TOP500 [46] also provides data measured under conditions greatly similar to the "net" interconnection contribution. Of course, the measured data reflect the sum of the contribution of all components. However, as will be shown below, in the total contribution those mentioned contributions dominate, and all but the contribution from networking are (nearly) unchanged so the *difference* of the measured  $\alpha$  can be directly compared with the *difference* of the sum of the calculated  $\alpha$ , although here only qualitative agreement can be expected.

Both the quality of the interconnection and the nominal performance are a parametric function of the time, so one can assume on the theory side that (in a limited time span) the interconnection contribution was changing in function of the nominal performance as shown in Fig. 6a. The other major contribution is assumed to be the calculation<sup>16</sup> itself. The benchmark calculation contributions for HPL and HPCG are very different, so the sum of the respective component plus the interconnection component is also very different. Given that at the beginning of the considered time span the contribution from the HPCG calculation and that of the interconnection are in the same order of magnitude, the sum only changes marginally (see the upper diagram lines), i.e., the measured performance changes only marginally.

The case with the HPL calculation is drastically different (see the lower diagram lines). Since in this case at the beginning of the considered time span the contribution from the interconnection is very much larger than that from the calculation, the sum of these two contributions changes sensitively as the speed of the interconnection improves. As soon as the contribution from the interconnection decreases to a value comparable with that of the calculation, the decrease in the sum slows down considerably, and *the further improvement in the interconnection causes only marginal decrease* in the value of the resulting  $\alpha$  (and so only a marginal increase in the payload performance).

The measured data enable to draw the same conclusion, but one must consider that here multiple parameters may have been changed. The tendency, however, is surprisingly clear. Figure 6b shows actually a 2.5D diagram: the size of the marks is proportional with the time passed since the beginning of the considered time period. A decade ago, the speed of the interconnection gave the major contribution to  $\alpha_{total}$ ; enhancing it drastically in the past few years, increased the efficacy. At the same time, because of the stalled single-processor performance, the other

<sup>&</sup>lt;sup>16</sup> This time also accessing data ("communicating" is included).



**Fig. 5** How different communication/computation intensities of the applications lead to different payload performance values in the same supercomputer system. Left column: the models of the computing intensities for different benchmarks. Right column: the corresponding payload performances and  $\alpha$  contributions in function of the nominal performance of a fictive supercomputer (P = 1 Gflop/s @ 1 GHz) in function of the nominal performance. The blue diagram line refers to the right-hand scale ( $R_{\text{Max}}$  values), all others ( $(1 - \alpha_{\text{eff}}^X)$  contributions) to the left scale. The figure is purely illustrating the concepts; the displayed numbers are somewhat similar to the real ones. The performance breakdown shown in the figures was experimentally measured by [3], [45, Fig. 7] and [24, Fig. 8] (color figure online)

technology components only changed marginally. The calculation contribution to  $\alpha$  from the benchmark HPL remained constant in function of the time, so the quick improvement in the interconnection technology resulted in a quick decrease in  $\alpha_{total}$ , and the relative weights of  $\alpha_{Net}$  and  $\alpha_{Compute}$  reversed. The decrease in value of  $(1 - \alpha)$  can be considered as the result of the decreased contribution from the interconnection.

However, the total  $\alpha$  contribution decreased considerably *only* until  $\alpha_{\text{Net}}$  reached the order of magnitude of  $\alpha_{\text{Compute}}$ . This occurred in the first 4–5 years of the time span shown in Fig. 6b: the sloping line is due to the enhancement of the interconnection. Then, they changed their role, and the constant contribution due to the calculation started to dominate, i.e., the total  $\alpha$  contribution decreased only marginally.



**Fig. 6** Effect of changing the dominating contribution. The left subfigure shows the theoretical estimation, the right subfigure the corresponding measured data, as derived from the public database TOP500 [46] (only the values for the first four supercomputers are shown). When the contribution from the interconnection drops below that of the calculation, the value of  $(1 - \alpha)$  (and the performance gain) get saturated

As soon as the computing contribution took over the dominating role, the total  $\alpha_{\text{total}}$  did not decrease any more: all measured data remained above that value of  $\alpha$ . Correspondingly, the payload performance (due to the enhanced interconnection) improved only marginally (and due to factors other than the interconnection).

At this point, as a consequence of that the dominating contributor changed, it was noticed that the efficacy of the benchmark HPL and that of the real-life applications started to differ by up to two orders of magnitude. Because of this the new benchmark program HPCG [47] was introduced, since "HPCG is designed to exercise computational and data access patterns that more closely match a different and broad set of important applications" [48]. Since the major contributor is the computing itself, different benchmarks contribute differently and since that time "supercomputers have two different efficiencies" [14]. Yes, if the dominating  $\alpha$  contribution (from the benchmark calculation) is different, then the same computer shows different efficiencies in function of the calculation it runs. Since that time that the interconnection provides less contributes mainly to the dark performance, rather than to the payload performance.

Reducing the communication really makes sense, however. The so called *HPL-AI* benchmark used Mixed Precision<sup>17</sup> [49] rather than Double Precision calculations. This enabled to achieve apparently nearly 3 times better performance gain, that (as correctly stated in the announcement) "*Achieving a 445 petaflops mixed precision result on HPL (equivalent to our 148.6 petaflops DP result)*," i.e., the peak DP performance did not change. See also discussion in Sect. 7.

<sup>&</sup>lt;sup>17</sup> Both names are rather inconsequent. On one side, the test itself has not much to do with AI, just uses the operand length common in Artificial Intelligence (AI) tasks (HPL, similarly to AI, is a workload type). On the other side, the Mixed Precision is actually Half Precision: it is natural that for multiplication twice as long operands are used temporarily. A different question is that the operations are contracted.

Unfortunately, the achievement comes from accessing less data in memory and using quicker operations on the shorter operands rather than reducing the communication intensity. For AI applications, the limitations remain the same as described above; except that when using Mixed Precision, the efficiency will be better by a factor of 2–3. Similarly, exchanging data directly between the processing units [33] (without using the global memory) also enhances  $\alpha$  (and payload performance) [50], but it represents a (slightly) different computing paradigm. Only the two mentioned measured data fall below the limiting line of  $(1 - \alpha)$  in Fig. 6.

A warning sign is that two of the first ten supercomputers did not provide their HPCG performance and other two used only a small portion of their cores in the HPCG benchmarking. As predicted: "scaling thus put larger machines at an inherent disadvantage" [3]. The reason is Eq. (4): using all of their cores the achievable performance is not higher (or maybe even lower), only the power consumption is higher. The cloud-like supercomputers have definitely a disadvantage in the HPCG competition: the Ethernet-like operation results in relatively high  $(1 - \alpha)$  values.

### 5.4 The effect of the length of the clock period

The behavior of the time in the computing systems is in parallel with the quantal nature of energy, known from the modern science. As mentioned above, the time of computing is of quantal nature: the time passes in discrete steps rather than continuously. This difference is not noticeable under the usual conditions: both human perception and the macroscopic computing operations are millions times longer. Under the extreme conditions represented by the many–many core systems, however, the quantal nature is the source of the inherent limitation of the parallelized sequential systems. The basic issue is that the operations must be synchronized; the asynchronous operation provides performance advantages [51].

The need to synchronize the operations (including those of the many-many processors) using a central clock signal is especially disadvantageous when attempting to imitate the behavior of biological systems without such a central signal. Although the intention to provide asynchronous operating mode was a major design point [23], the hidden synchronization (mainly introduced by thinking in the conventional SW solutions) led to very poor efficiency [24] when the system was attempting to perform its flagship goal: simulating the functionality of (a large part of) human brain.

As discussed in Sect. 5.1, the performance depends also on the measurement time, because of the fixed-time contributions. When making only 10 s long measurements, compared to the HPL benchmarking time, the smaller denominator may result up to  $10^3$  times worse  $(1 - \alpha_{\text{eff}})$  and performance gain values. The dominant limiting factor, however, is a different one.

In brain simulation a 1 *ms* integration time (essentially sampling time) is commonly used [24], because the biological time (when the events happen) and the computing time (how much computing time is required to perform the computing operations to describe the same) are not only different, but also not directly related. To avoid working with "signals from the future," at the end of this period the calculated new state variables must be communicated to all of the (interested) fellow neurons. This action essentially introduces a "biology clock signal period," being million times longer than the electronic clock signal period. Correspondingly, the achievable performance is desperately low: less than 10<sup>5</sup> neurons can actually be simulated, out of the planned 10<sup>9</sup> [24].<sup>18</sup> For a detailed discussion see [25].

Figure 7 depicts the experimental equivalent of Fig. 5. In [24] the power consumption efficiency is also investigated. It is presumed that (in order to avoid obsolete energy consumption) they performed the measurement at the point where involving more cores increases the power consumption but does not increase the payload simulation performance. As using AI workload (for a discussion from this point of view see [52]) on supercomputers is of growing importance, the performance gain of an AI application can be estimated to be between those of HPCG and brain simulation, closer to that of the HPCG. As discussed experimentally in [53] and theoretically in [52], in the case of neural networks (especially in the case of selecting improper layering depth) the efficiency can be much lower.

Recall, that since the AI nodes perform simple calculations compared to the functionality of the supercomputer benchmarks, the communication/calculation ratio is much higher, making the efficacy even worse. The conclusions are underpinned by experimental research [53]:

- "Strong scaling is stalling after only a few dozen nodes"
- "The scalability stalls when the compute times drop below the communication times, leaving compute units idle, hence becoming an communication bound problem."
- "The network layout has a large impact on the crucial communication/compute ratio: shallow networks with many neurons per layer ...scale worse than deep networks with less neurons."

The massively "bursty" nature of the data (different nodes of the layer want to use the communication at the same moment) also makes the case harder. The commonly used global bus is overloaded with messages, that may lead to a "communicational collapse" (demonstrated in Fig 5a in [54]): at extremely large number of cores exceeding the critical threshold of communication intensity leads to unexpected and drastic change of network latency.

# 6 The accuracy and reliability of the model

As the parameters of the model are inferred from non-dedicated single-shot measurements, their reliability is limited. One can verify, however, how the model predicts values derived from later measurements. The supercomputers usually do not have a long lifespan and several documented stages. One of the rare exceptions is the

<sup>&</sup>lt;sup>18</sup> Despite its failure, the SpiNNaker2 is also under construction [20].



The rooflines of performance gain of supercomputers

**Fig. 7** Performance gains of supercomputers modeled as "roofline" [36], as measured with the benchmarks HPL and HPCG (taken from the database TOP500 [37]), and the one for brain simulation is concluded from [24]

supercomputer *Piz Daint*. The documented lifetime spans over 6 years and during that time different numbers of cores, without and with acceleration, using different accelerators, were used.

Figure 8 depicts the performance and efficiency values published during its lifetime, together with the diagram lines predicting (at the time of making the prediction) the values at the higher nominal performance values. The left subfigure shows how the changes made in the configuration affected the efficiency (the timeline starts in the top right corner and the adjacent stages are connected by a line).

In the right subfigure the bubbles represent data published in the adjacent editions of the TOP500 lists, the diagram lines crossing them are the predictions made from that snapshot. The predicted value shall be compared to the value published in the next list. It is especially remarkable that introducing GPGPU acceleration resulted only in a slight increase (in good agreement with [15, 55]) compared to the value expected based purely on the increase in the number of cores. Although between the samplings more than one parameter was changed, that is the net effect cannot be demonstrated clearly, the measured data sufficiently underpin our limited validity conclusions and show that the theory correctly describes the tendency of the development of the performance and the efficiency, and even the predicted performance values are reasonably accurate.

Introducing GPU accelerator is a one-time performance increase step [15], and cannot be taken into account by the theory. Notice that introducing the accelerator increased the payload performance, but decreased the efficiency (copying data from one address space to another increases latency). Changing the accelerator to another



**Fig. 8** History of supercomputer Piz Daint in terms of efficiency and payload performance [46]. The left subfigure shows how the efficiency changed as the developers proceeded toward higher performance. The right subfigure shows the reported performance data (the bubbles), together with the diagram line calculated from the value as described above. Compare the value of the diagram line to the measured performance data in the next reported stage

type with slightly higher performance (but higher latency due to the larger GPGPU memory) caused a slight *decrease* in the absolute performance because of the considerably dropped efficiency.

# 7 Toward zettaflops

As detailed above, the theoretical model enables to calculate the payload performance in the first-order approximation at any nominal performance value. Given that the parameter value is calculated from a snapshot, and that the calculation is essentially an extrapolation, the values shown in Fig. 9 are only approximations (on the expected accuracy see Sect. 6), but somewhat similar to the real values. In the light of all this, one can estimate the short time and the longer time development of the supercomputer performance, see Fig. 9a, b. The diagram lines are calculated using Eq. (4), with the  $\alpha$  parameter values calculated from the TOP500 data of Summit supercomputer; the bubbles show the measured values. The diagram lines from the bottom up show the double floating precision HPCG, HPL payload performances. After that the half precision [49] HPL diagram line follows.

In addition to the really measured and published performance data, two more diagram lines representing two more calculated  $\alpha$  values are also depicted. The "FPO" (orange) diagram line is calculated with the assumption that the supercomputer makes the stuff needed to perform the HPL benchmark, but the actual FP operations are not performed, or in other words the computer works with zero-bit length floating operands (FPO).<sup>19</sup>

<sup>&</sup>lt;sup>19</sup> The role of  $\alpha_{HPL}^{FPO}$  is akin to execution time of the "empty loop" in programming.



**Fig. 9** Tendency of the development of the payload performance, in the near and farther future of supercomputing. The parameters of the actual world leader are used for illustration. The diagram lines are calculated from the theory; the marks are the measured values from the database TOP500 [46, 49]

It is expected that when using half precision (FP16), four times less data are transferred and manipulated by the system (the measured power consumption data underpin the statement), so it is expected that

$$\alpha_{\rm HPL}^{\rm FP64} = 4 \times \alpha_{\rm HPL}^{\rm FP16} \tag{8}$$

However, the performance is only 3 times higher than in the case of using 64-bit (FP64) operands. Given that the measured payload performance directly reflects the sum of all contributions, one can assume that the contributions of the two calculations with differing operand length plus the rest of the contributions define the  $\alpha$  values we can conclude from the measurements:

$$1 - \alpha_{\rm HPL}^{\rm FP0} - \alpha_{\rm HPL}^{\rm FP64} = 1.465 \times 10^{-7}$$
  

$$1 - \alpha_{\rm HPL}^{\rm FP0} - \alpha_{\rm HPL}^{\rm FP16} = 0.488 \times 10^{-7}$$
(9)

where  $\alpha_{HPL}^{FP0}$  is the contribution of all parts independent from the floating manipulation. From this,

$$\begin{aligned} \alpha_{\rm HPL}^{\rm FP0} &= 0.19 \times 10^{-7} \\ \alpha_{\rm HPL}^{\rm FP16} &= 0.33 \times 10^{-7} \\ \alpha_{\rm HPL}^{\rm FP64} &= 1.3 \times 10^{-7} \\ \alpha_{\rm HPCG}^{\rm FP64} &= 208. \times 10^{-7} \end{aligned}$$
(10)

These data directly underpin that the technology is (almost) perfect: the contribution from the benchmark calculation HPCG-FP64 is orders of magnitude larger<sup>20</sup> than the contribution from all the rests. Recalling that the benchmark program imitates

<sup>&</sup>lt;sup>20</sup> Recall here that cache behavior may be included.

the behavior (as defined by the resulting  $\alpha$ ) of real-life programs, one can see that the contribution from the non-computational actors is about thousand times smaller than the contribution of the computation+communication itself.

The performance corresponding to  $\alpha_{HPL}^{FP0}$  is slightly above 1 EFlops (when actually making no floating operations, i.e., rather Eops). Another peak performance reported<sup>21</sup> when running genomics code on Summit (by using a mixture of numerical precisions and mostly non-floating point instructions) is 1.88 Eops, corresponding to  $\alpha_{Genom}^{FP0} = 1 \times 10^{-8}$ ; for the scaling of that type of calculations see Fig. 4. Given that those two values refer to different mixtures of instructions, the agreement is more than satisfactory.

The "Science" (red) diagram line is calculated with the assumption that nothing is calculated but the science (the finite propagation time due to the finite speed of light)<sup>22</sup> limits the payload performance. The "ideal interconnection" diagram line should come between the diagram lines "Science" and "FPO." The non-linearity of the payload performance around the Eflops nominal performance is obvious and depends both on the amount of computing+communication and the nominal performance (represented by the number of cores).

Figure 9b shows the farther future (in the first-order approximation): toward the Zflops [7]. No surprise that all payload performance diagram lines run into saturation, even the "FP0" and "Science" ones. As discussed in detail in [4], the behavior of the computing system under extreme conditions shows surprising parallels with the behavior of natural objects. The example shown in Table 1 and Fig. 4 parallels the non-linearity of speed addition in modern physics with the performance addition in modern computing.

The behavior of that supercomputer, as discussed in this section, is somewhat analogous with that of the quantum systems, where the measurement selects one of the possible states (and actually, kills the quantum state). Here the computer has the potential of high performance until the payload performance is measured (i.e., a calculation started). Different components have their really impressively high performance, but benchmarking the resulting system introduces one more component: the needed computations. Since the measurement method itself is a calculation (having the smallest possible  $\alpha^{\text{Calc}}$  contribution), the measurable payload performance value cannot be smaller than the value the measurement procedure itself represents. This means that the measuring process (running some calculation) itself destroys the potentially achievable high performance (i.e., the one when the supercomputer does not make any calculations) of the supercomputer. For the real-life programs (such as HPCG), the saturation value already set well below the Eflops nominal performance. Further enhancements in technology, like tensor processors and OpenCAPI connection bus, can slightly increase the saturation level, but cannot change the shape of the diagram line. The supercomputers reached their technical

<sup>&</sup>lt;sup>21</sup> https://www.olcf.ornl.gov/2018/06/08/genomics-code-exceeds-exaops-on-summit-supercomputer/. Accessed 21 Feb 2020

<sup>&</sup>lt;sup>22</sup> 100 m cable length was assumed, which means 10<sup>6</sup> processors pro cm and some GW dissipation.

limitations. Continuing enhancing the components of a supercomputer that wants to run any kind of calculation, without changing the computing paradigm, is not worth any more. To enter the "next level," really renewing the computing paradigm is needed [56, 57].

# 8 Conclusion

The ironic remark that ""Perhaps supercomputers should just be required to have written in small letters at the bottom on their shiny cabinets: 'Object manipulations in this supercomputer run slower than they appear." [14]" is becoming increasingly relevant. The impressive numbers about the performance of the individual components (including single-processor and/or GPU performance and speed of the interconnection) are becoming less relevant when going to the extremes. Given that the largest  $\alpha$  contribution today takes its origin in the calculation the supercomputer runs, even the best possible benchmark HPL dominates the performance measurement. Enhancing the other contributions, such as interconnection, results in marginal enhancement of the performance, i.e., the overwhelming majority of the expenses increases the "dark performance" only. Because of this, the answers to the questions in the title are: there are as many performance values as many measurement methods, and actually the benchmarks measure mainly how much mathematics/communication the benchmark program does, rather than the supercomputer architecture (provided that all components deliver the technically achievable best parameters).

Acknowledgements Open access funding provided by University of Debrecen (DE). Project No. K-125547 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the K funding scheme. Also the ERC-ECAS support of Project 886183 is acknowledged.

# **Compliance with ethical standards**

Conflict of interest The authors declare that they have no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

# References

1. Fuller SH, Millett LI (eds) (2011) The future of computing performance: game over or next level?. National Academies Press, Washington

- Amdahl GM (1967) Validity of the single processor approach to achieving large-scale computing capabilities. In: AFIPS Conference Proceedings, vol 30, pp 483–485
- Singh JP, Hennessy JL, Gupta A (1993) Scaling parallel programs for multiprocessors: methodology and examples. Computer 26(7):42–50
- Végh J, Tisan A (2019) The need for modern computing paradigm: science applied to computing. In: 2019 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE. http://arxiv.org/abs/1908.02651 (in print)
- Végh J (2019) The performance wall of parallelized sequential computing: the roofline of supercomputer performance gain. In: Parallel Computing. http://arxiv.org/abs/1908.02280 (in review)
- 6. Markov I (2014) Limits on fundamental limits to computation. Nature 512(7513):147-154
- Liao XK, Lu K, Yang CQ, Li JW, Yuan Y, Lai MC, Huang LB, Lu PJ, Fang JB, Ren J, Shen J (2018) Moving from exascale to zettascale computing: challenges and techniques. Front Inf Technol Electron Eng 19(10):1236–1244. https://doi.org/10.1631/FITEE.1800494
- Feldman M (2019) Exascale is not your grandfather's HPC. https://www.nextplatfo rm.com/2019/10/22/exascale-is-not-your-grandfathers-hpc/. Accessed 21 Feb 2020
- US Government NSA and DOE (December 2016) A report from the NSA-DOE technical meeting on high performance computing. https://www.nitrd.gov/nitrdgroups/images/b/b4/NSA\_ DOE\_HPC\_TechMeetingReport.pdf. Accessed 21 Feb 2020
- 10. Service RF (2018) Design for U.S. exascale computer takes shape. Science 359:617-618
- European Commission (2016) Implementation of the action plan for the European high-performance computing strategy. http://ec.europa.eu/newsroom/dae/document.cfm?doc\_id=15269. Accessed 21 Feb 2020
- Extremtech (2018) Japan tests silicon for Exascale computing in 2021. https://www.extremetec h.com/computing/272558-japan-tests-silicon-for-exascale-computing-in-2021. Accessed 21 Feb 2020
- 13. Bourzac K (2017) Stretching supercomputers to the limit. Nature 551:554-556
- 14. IEEE Spectrum (2017) Two different top500 supercomputing benchmarks show two different top supercomputers. https://spectrum.ieee.org/tech-talk/computing/hardware/two-different-top50 0-supercomputing-benchmarks-show-two-different-top-supercomputers. Accessed 21 Feb 2020
- 15. Simon H (2014) Why we need Exascale and why we won't get there by 2020. In: Exascale Radioastronomy Meeting, ser. AASCTS2. https://www.researchgate.net/publication/261879110\_ Why\_we\_need\_Exascale\_and\_why\_we\_wont\_get\_there\_by\_2020. Accessed 21 Feb 2020
- 16. Gustafson JL (1988) Reevaluating Amdahl's law. Commun ACM 31(5):532-533
- 17. Krishnaprasad S (2001) Uses and abuses of Amdahl's law. J Comput Sci Coll 17(2):288–293
- Shi Y (1996) Reevaluating Amdahl's law and Gustafson's law. https://www.researchgate.net/ publication/228367369\_Reevaluating\_Amdahl's\_law\_and\_Gustafson's\_law. Accessed 21 Feb 2020
- www.top500.org (2017) Intel Dumps Knights Hill, Future of Xeon Phi Product Line Uncertain. https://www.top500.org/news/intel-dumps-knights-hill-future-of-xeon-phi-product-line-uncer tain///. Accessed 21 Feb 2020
- Liu C, Bellec G, Vogginger B, Kappel D, Partzsch J, Neumärker F, Höppner S, Maass W, Furber SB, Legenstein R, Mayr CG (2018) Memory-efficient deep learning on a spinnaker 2 prototype. Front Neurosci 12:840. https://doi.org/10.3389/fnins.2018.00840
- Top500.org (2017) Retooled Aurora Supercomputer Will Be America's First Exascale System. https://www.top500.org/news/retooled-aurora-supercomputer-will-be-americas-first-exascalesystem/. Accessed 21 Feb 2020
- Kunkel S, Schmidt M, Eppler JM, Plesser HE, Masumoto G, Igarashi J, Ishii S, Fukai T, Morrison A, Diesmann M, Helias M (2014) Spiking network simulation code for petascale computers. Front Neuroinform 8:78
- 23. Furber SB, Lester DR, Plana LA, Garside JD, Painkras E, Temple S, Brown AD (2013) Overview of the SpiNNaker system architecture. IEEE Trans Comput 62(12):2454–2467
- 24. van Albada SJ, Rowley AG, Senk J, Hopkins M, Schmidt M, Stokes AB, Lester DR, Diesmann M, Furber SB (2018) Performance comparison of the digital neuromorphic hardware SpiNNaker and the neural network simulation software NEST for a full-scale cortical microcircuit model. Front Neurosci 12:291
- 25. Végh J (2019) How Amdahl's law limits the performance of large artificial neural networks: (why the functionality of full-scale brain simulation on processor-based simulators is limited). Brain Inform 6:1–11

- Patterson D, Hennessy J (eds) (2017) Computer organization and design. RISC-V edition. Morgan Kaufmann, Burlington
- 27. Hwang K, Jotwani N (2016) Advanced computer architecture: parallelism, scalability, programmability, 3rd edn. McGraw Hill, New York City
- Weaver V, Terpstra D, Moore S (2013) Non-determinism and overcount on modern hardware performance counter implementations. In: 2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), April 2013, pp 215–224
- Molnár P, Végh J (2017) Measuring performance of processor instructions and operating system services in soft processor based systems. In: 18th International Carpathian Control Conference (ICCC), pp 381–387
- Ellen F, Hendler D, Shavit N (2012) On the inherent sequentiality of concurrent objects. SIAM J Comput 43(3):519–536
- 31. Yavits L, Morad A, Ginosar R (2014) The effect of communication and synchronization on Amdahl's law in multicore systems. Parallel Comput 40(1):1–16
- Végh J, Molnár P (2017) How to measure perfectness of parallelization in hardware/software systems. In: 18th International Carpathian Control Conference (ICCC), pp 394–399
- Zheng F, Li H-L, Lv H, Guo F, Xu X-H, Xie X-H (2015) Cooperative computing techniques for a deeply fused and heterogeneous many-core processor architecture. J Comput Sci Technol 30(1):145–162
- Mohammadi M, Bazhirov T (2018) Comparative benchmarking of cloud computing vendors with high performance Linpack. In: Proceedings of the 2nd International Conference on High Performance Compilation, Computing and Communications, ser. HP3C. ACM, New York, NY, pp 1–5. https://doi.org/10.1145/3195612.3195613
- Karp AH, Flatt HP (1990) Measuring parallel processor performance. Commun ACM 33(5):539–543
- Williams S, Waterman A, Patterson D (2009) Roofline: an insightful visual performance model for multicore architectures. Commun ACM 52(4):65–76
- TOP500 (2017) November 2017 list of supercomputers. https://www.top500.org/lists/2017/11/. Accessed 21 Feb 2020
- Hsu C-H, Kuehn JA, Poole SW (2012) Towards efficient supercomputing: searching for the right efficiency metric. In: Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering, pp 1157–162. https://doi.org/10.1145/2188286.2188309
- 39. Martin DS (2012) Hardware and software techniques for scalable thousand-core systems. Ph.D. dissertation, Stanford University, Berkeley
- Dongarra J (2016) Report on the Sunway TaihuLight System. University of Tennessee Department of Electrical Engineering and Computer Science, Technical Report UT-EECS-16-742, June 2016. http://www.netlib.org/utk/people/JackDongarra/PAPERS/sunway-report-2016.pdf
- Tsafrir D (2007) The context-switch overhead inflicted by hardware interrupts (and the enigma of do-nothing loops). In: Proceedings of the 2007 Workshop on Experimental Computer Science, ser. ExpCS '07. ACM, New York, NY, pp 3–3
- David FM, Carlyle JC, Campbell RH (2007) Context switch overheads for Linux on ARM platforms. In: Proceedings of the 2007 Workshop on Experimental Computer Science, ser. ExpCS '07. ACM, New York, NY. https://doi.org/10.1145/1281700.1281703
- Végh J, Vásárhelyi J, Drótos D (2019) The performance wall of large parallel computing systems. Lecture Notes in Networks and Systems, vol 68. Springer, Berlin, pp 224–237
- 44. Végh J (2018) How Amdahl's law restricts supercomputer applications and building ever bigger supercomputers. CoRR. http://arxiv.org/abs/1708.01462
- 45. Ippen T, Eppler JM, Plesser HE, Diesmann M (2017) Constructing neuronal network models in massively parallel environments. Front Neuroinform 11:30
- 46. TOP500.org (2019) The top 500 supercomputers. https://www.top500.org/. Accessed 21 Feb 2020
- Dongarra J, Heroux MA, Luszczek P (2015) High-performance conjugate-gradient benchmark: a new metric for ranking high-performance computing systems. Int J High Perform Comput Appl. https://doi.org/10.1177/1094342015593158
- HPCG Benchmark (2016) HPCG Benchmark. http://www.hpcg-benchmark.org/. Accessed 21 Feb 2020
- 49. Haidar A, Tomov S, Dongarra J, Higham NJ (2018) Harnessing GPU tensor cores for fast FP16 arithmetic to speed up mixed-precision iterative refinement solvers. In: Proceedings of the

International Conference for High Performance Computing, Networking, Storage, and Analysis, ser. SC '18. IEEE Press, pp 47:1–47:11

- 50. Ao Y, Yang C, Liu F, Yin W, Jiang L, Sun Q (2018) Performance optimization of the HPCG benchmark on the Sunway TaihuLight supercomputer. ACM Trans Archit Code Optim 15(1):11:1–11:20
- Horn.J GP, He J, Papageorgiou A, Poole C (2017) IBM CICS Asynchronous API: concurrent processing made simple. http://www.redbooks.ibm.com/redbooks/pdfs/sg248411.pdf. Accessed 21 Feb 2020
- 52. Végh J (2020) How deep the machine learning can be, ser. A closer look at convolutional neural networks. Nova, Hauppauge, pp 141–169 (**in press**)
- Keuper J, Preundt FJ (2017) In: 2nd Workshop on Machine Learning in HPC Environments (MLHPC) (IEEE, 2016), pp 1469–1476. https://www.researchgate.net/publication/308457837. Accessed 21 Feb 2020
- 54. Moradi S, Manohar R (2018) The impact of on-chip communication on memory technologies for neuromorphic systems. J Phys D: Appl Phys 52(1):014003
- 55. Lee VW, Kim C, Chhugani J, Deisher M, Kim D, Nguyen AD, Satish N, Smelyanskiy M, Chennupaty S, Hammarlund P, Singhal R, Dubey P (2010) Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU. In: Proceedings of the 37th Annual International Symposium on Computer Architecture, ser. ISCA '10. ACM, New York, NY, pp 451–460. https://doi.org/10.1145/1815961.1816021
- Végh J (2018) Renewing computing paradigms for more efficient parallelization of single-threads, ser. Advances in parallel computing. Chapter 13, vol 29. IOS Press, Amsterdam, pp 305–330
- 57. Végh J (2018) Introducing the explicitly many-processor approach. Parallel Comput 75:28-40

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.