Voltage, throughput, power, reliability and multi-core scaling

Fei Xia[1], Ashur Rafiev[1], Ali Aalsaud[1], Mohammed Al-Hayanni[1], James Davis[2], Joshua Levine[2], Andrey Mokhov[1], Alexander Romanovsky[1], Rishad Shafik[1,3], Alex Yakovlev[1], Sheng Yang[3,4]

[1]Newcastle University, [2]Imperial College London, [3]University of Southampton, [4]ARM

1 Abstract

Parallelization has been used to maintain a reasonable balance between energy consumption and performance in computing platforms especially in modern multi- and many-core systems. This paper studies the interplay between performance and energy, and their relationships with parallelization scaling in the context of the *reliable operating region*, focusing on the effectiveness of parallelization scaling in throughput-power tradeoffs. Theoretical and experimental explorations show that a meaningful cross-platform analysis of this interplay can be achieved using the proposed method of bi-normalization of the ROR. The concept of this interplay is captured in an online tool for finding optimal operating points.

2 Introduction

In digital CMOS circuits, a higher supply voltage (called V henceforth) usually permits a higher operating (clock) frequency for capacitive load-balancing, and hence a higher throughput, given the same hardware platform. The scheme of dynamic voltage and frequency scaling (DVFS) scales V and clock frequency (henceforth called F) together in order to obtain the best throughput under a given power budget or to save power for a given throughput requirement [1].

It is possible to increase system throughput for a given power limit, or to reduce power whilst maintaining throughput, by combining DVFS with parallelization or scaling to multiple computation units if the computation can be parallelized [2]. A major challenge for the precise analysis of the effectiveness of using parallelization for these goals is to determine the parallelizability of any particular execution, which is related to complex issues such as software and hardware architecture details and must be modelled on a per-execution basis [3]. Another challenge is that quantitative studies of power and/or throughput improvements for any DVFS decision need complicated execution-dependent models [4].

This paper explores the interplay between DVFS and parallelization scalability with respect to performance and power. The interplay is captured using the concept of a reliable operating region (ROR), which can be established from the knowledge of system reliability through experiments or simulations. The ROR therefore provides containment for platform and application specifics, hence helping to make the further analysis steps generic.

The focus of this paper is the effectiveness of parallelization scaling, the latter denoted as η.

The ROR-based method can explore η across the entire voltage range of a platform, from sub-threshold to super-threshold regions. The explorations and models presented in this paper confirm and explain the general view that combined DVFS and parallelization scaling produces the best advantage when V is scaled down to near-threshold voltages. This is known as near-threshold

computing (NTC) [5]. Current commercial platforms tend to avoid this region, however, as shown later in the paper.

This paper highlights the following topics:

- The concept of the ROR.
- The study of the inter-relationship of voltage, throughput, power and reliability in the context of multi-core scaling.
- Bi-normalizing the ROR, which facilitates certain cross-platform and cross-application comparisons. The investigation is based on experimental data and mathematical models.
- Addressing both execution-independent analysis and complex system/workload combinations.
- A Web-based DVFS/parallelization scaling exploration tool as a technical solution for finding optimal operating points. The tool also has pedagogical applications in an academic teaching and research environment.

By addressing these topics, this paper sheds fundamental insight into the permissible points of operation under various system design/implementation constraints. It should be noted that the models derived in this paper are not intended for use in absolute value predictions, but aimed at exploring performance, energy, reliability (PER) and scalability relationships through η.

3 The reliable operating region

Operational reliability depends on the system platform including various hardware-related design-time and runtime decisions as well as applications and their requirements. Different metrics can be used to describe the degree of reliability. The proposed method is agnostic to the exact type of reliability metric as long as it facilitates a fair comparison. A popular reliability metric for cross-comparing different systems and applications is mean time between failures (MTBF), which assumes that 'failure' can be fairly defined in each comparison. For instance, failure can be defined as losing accuracy, and accuracy metrics such as Signal-to-noise ratio (SNR), widely used in information engineering fields, can be much easier to measure in experiments but are application-specific. This paper does not attempt to study the relationships between different reliability metrics, but assumes that for any problem being studied, a metric or set of metrics can be agreed on. SNR is used in this paper as an example to demonstrate the execution-independent comparison of η applied to execution-dependent data (like SNR), as described in Section 5. Furthermore, this paper focuses on the effects of voltage, frequency and parallelization scaling, and does not address the dependency of reliability on other (e.g. microarchitecture and application) design-time and run-time decisions. For any given application design and microarchitecture choice, the proposed models and techniques apply on voltage, frequency and parallelization scaling decisions.

To achieve any particular value of any reliability metric, a system must operate within voltage and frequency constraints. For instance, reducing V may cause an increase in soft error rate (SER) [6]. Conversely, increasing V causes an increase of temperature, accelerates aging and the probability of breakdowns [7]. This leads to

$$V_{\min} \leq V \leq V_{\max} \quad (1)$$

An execution may require more than a certain level of throughput θ to be meaningful [8]. This leads to

$$\theta \geq \theta_{\min} \quad (2)$$

The problem of fault tolerance can be addressed by requiring computational redundancy. This is reflected in an increased aggregated $\theta_{\min}$. The tradeoff between spatial redundancy and time redundancy can then be captured in the parallelization scaling described in Section 4.

The amount of available power $P_{\max}$ limits the behaviour of the system [2] [9].

If hardware is run with a clock too fast for its V, computations may not complete in time, leading to reductions of any reasonable reliability metric. With aging, to maintain the same θ, V needs to be increased [10].

We explore the concept of the ROR in the context of a throughput-voltage (θ-V) space. The ROR for a platform within the θ-V space is bounded by constraints on power, timing reliability and $\theta_{\min}$, $V_{\min}$ and $V_{\max}$ boundaries, as illustrated in Figure 1.

The ROR boundaries are therefore directly related to physical causes of all types of reliability attributable to computation. It should be possible to express any reasonable reliability metric with these boundaries.

This method caters for both execution-dependent RORs, which are more precise but require higher effort to obtain, and conservative RORs as shown in Figure 1. Commercial systems typically provide RORs in the form of pre-defined sets of conservative DVFS points. This allows for execution-independent studies.

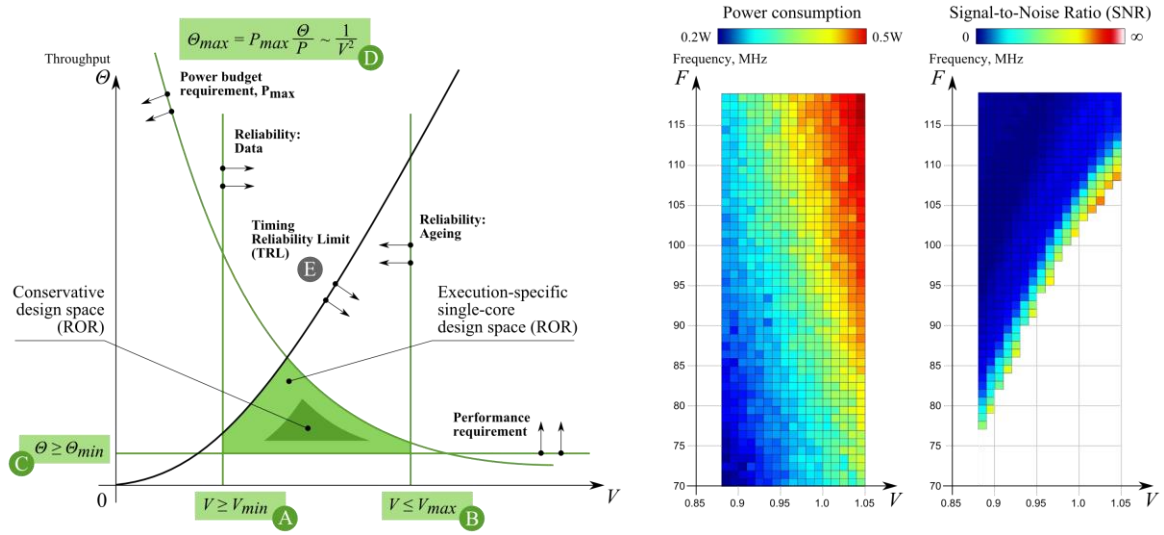


Figure 1 On the left, the ROR is bounded by the high and low voltage limits (A and B), the throughput requirement line (C), the timing (clock) reliability limit or TRL (E) and the power limit (D). The TRL is usually obtained through experiments or specified by the vendor of a platform. The other boundaries have known formulae. An exact ROR is application-dependent and provides for the most efficient operation. Shrinking the ROR in the directions of the arrows will eventually provide conservative and application-independent operating points. On the right is power and reliability data collected from a Xilinx Zynq ZC702 FPGA device running an image processing application. It shows the ROR of a real platform and application with SNR as the accuracy metric, assuming a reliability metric related to accuracy. Various reliability factors affect these boundaries. A more stringent SER requirement pushes A to the right (V needs to be increased to reduce SER). Aging causes the TRL (E) to drop as a higher V is required to maintain the same θ after aging. However, a higher V accelerates aging, hence a more stringent aging speed requirement pushes B to the left, reducing how much V can be raised. In general, whichever reliability metric is used, a more stringent requirement shrinks the RoR and a more relaxed requirement expands it. For instance, if the SNR requirement is relaxed in the right-hand picture, the ROR is enlarged upwards.

4 Exploring parallelization scaling

In the previous section it was shown that among the boundaries, only TRL and $P_{\max}$ boundaries are dependent on both θ and V. Hence only these boundaries affect parallelization scaling, as explained in this section.

4.1 Switching power considerations

Switching (dynamic) power is related to frequency F and voltage V in the following manner [2]:

$$P = AFV^2 \quad (3)$$

where A is a coefficient influenced by hardware area and switching activity , P is the power and F is the switching frequency.

Computational throughput θ is usually expressed in instructions per second (IPS), which is related to F through instructions per clock cycle (IPC or u), i.e. $\theta = uF$. Assuming that a certain computation execution has a constant average IPC, constant $P_{\max}$ curves can then be plotted in the θ-V space, following equation D in Figure 1, as shown on the left of Figure 2.

Next, we explore the issue of parallelization scaling, initially with an assumption of ideal scaling with $\theta_k = k\theta$. Non-ideal parallelization scaling will be discussed later. Under ideal scaling with a scaling factor of k (scaling a computation to k computation units, henceforth called k–scaling), A is scaled in the same way, i.e.

$$A_k = kA \quad (4)$$

where A_k is the A coefficient of the hardware after k-scaling.

If the power budget does not change after k-scaling, for each computation unit in a scaled set-up, equation (3) becomes

$$P = \frac{AFV^2}{k} \quad (5)$$

as k cores share the power budget.

The per-core power reduction by a factor of k usually reduces a unit's maximum throughput by less than k. This reduced maximum throughput multiplied across k units provides a net increase of usable throughput (Figure 2), which motivates combining DVFS with parallelization.

In this section we use P, F and V data collected from an asynchronous SRAM controller [11] as an example of determining the ROR from experimental data. Considering just switching power is valid only in the V range where the switching power dominates. This is found to be $0.6\text{V} \leq V \leq 1.2\text{V}$, where the experimental data shows A to be near-constant.

The SRAM controller is self-timed and hence always runs at the highest speed which maintains 100% reliability when operating within the aforementioned voltage range. It also has two computational actions, read and write, and each has a constant IPC. This results in the timing reliability limit (TRL) curves on the right of Figure 2. Although only a memory controller and not a full processor core, it is a CMOS combinatorial logic block and we can explore core scaling with its curves without losing generality. Similar TRLs have been observed from experimental data on a large number of combinatorial logic computation units including full cores running standard benchmarks [12]. The ROR (here without considering $V_{\max}$, $V_{\min}$ and $\theta_{\min}$) is reduced when the power limit is lowered.

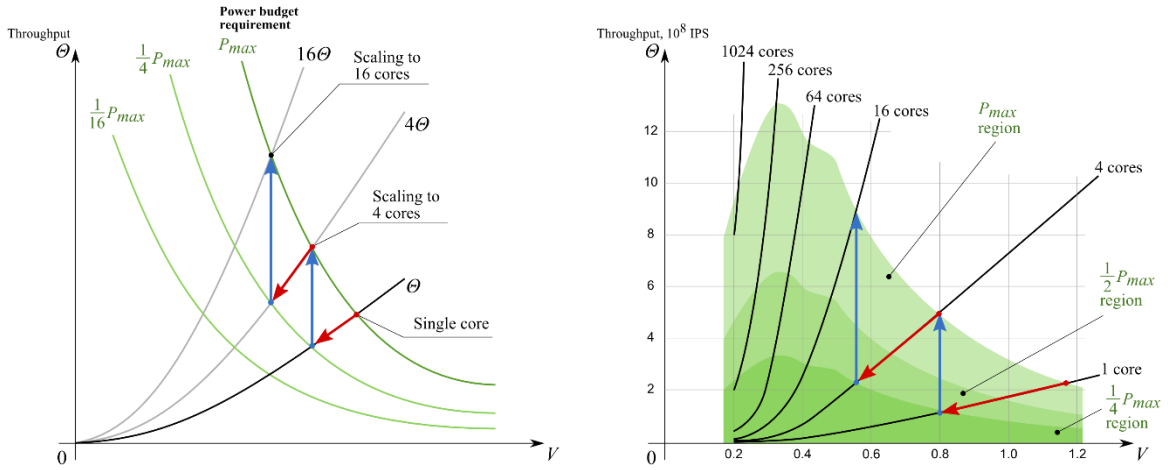


Figure 2 On the left, perfect scaling to theoretical switching power limit. On the right, real measured data from an asynchronous SRAM controller illustrating parallelization scaling and power limits. Under the same power consumed by a single core at nominal V (1.2V) and max θ (234MIPS – millions of instructions per second), the system is explored with more cores. With four cores, each core shares ¼ of the power budget, corresponding to 0.8V and $\theta = 129\text{MIPS}$. All four cores at 0.8V gives $\theta = 516\text{MIPS}$. With 16 cores, the system works at around 0.6V and achieves $\theta = 1\text{GIPS}$. Other factors being equal, k-scaling enlarges the ROR upwards in when considering only switching power.

4.2 Additional power consumption considerations

Below about 0.6V in the above example, equation (3) no longer approximates the total power as leakage power becomes significant.

Instead of using the complex power equations taking leakage power into account, observed power from experiments can be used to draw the constant power curves.

To determine the shape of the power boundary for any $P_{\max}$, given (5), for each point i where experimental power data exists, we calculate the maximum scaling factor k_i based on

$$k_i = \frac{P_{\max}}{P_i} \qquad (6)$$

where P_i is the experimental single-core power observed at data point i.

Plotting $\theta = k_i\theta_i$ produces the constant power curve $P = P_{\max}$ in the θ-V space. The $P = P_{\max}$ curves for the asynchronous SRAM controller are also shown in Figure 2. Similar constant power curve shapes have also been observed from other platforms [12].

The benefit of scaling is reduced when leakage power becomes important. Scaling with a factor of four from 0.6V leads to just above 0.4V with a throughput increase of roughly ⅓. Scaling further may reduce $\theta_{\max}$.

When $P_{\max}$ is increased, scopes for scaling further are enhanced. The $k = 16$ TRL intersects the $¼P_{\max}$ boundary at a lower V (worse scaling) than where it intersects the $P_{\max}$ boundary.

In general, a system design may be limited by power limit $P_{\max}$ and hardware availability limit $k_{\max}$. k-scaling characteristics based on the ROR in the form of Figure 2 help the designer find the best matching P for a given $k_{\max}$ and the best matching k for a given $P_{\max}$.

In the next section we first concentrate only on switching power.

5 The influence of the TRL on throughput and power

5.1 The bi-normalized ROR

Whilst the qualitative shape of constant power curves is dictated by CMOS fundamentals, the TRLs are the results of platform design decisions and their shapes may influence the tradeoffs between throughput, power and reliability in the context of parallelization scaling.

Assuming k-scaling, we consider the general case where k is a real number.

Let

$$V_k = \alpha_V V,\ F_k = \alpha_F F \tag{7}$$

where V_k and F_k are the k-scaled voltage and frequency.

α_V and α_F are the voltage and frequency scaling ratios. The unscaled switching power and throughput are respectively

$$P = AFV^2,\ \theta = uF \tag{8}$$

Assuming ideal scaling, the k-scaled power and throughput are

$$P_k = kA\alpha_F F\alpha_V^2 V^2 = \alpha_P P,\ \theta_k = k\alpha_F uF = \alpha_\theta uF \tag{9}$$

The scaling point $\langle V_k, F_k \rangle$ must fall within the ROR. Also, the scaling factor k must not exceed the platform's limit (the number of computation units) $k_{\max}$. α_P is the power scaling ratio and α_θ is the throughput scaling ratio.

Ratios allow working in a bi-normalized α_F-α_V space (Figure 3) instead of the specific θ-V space of any platform. This leads to platform independence and better comparisons between multiple platforms and between different scaling regions of the same platform.

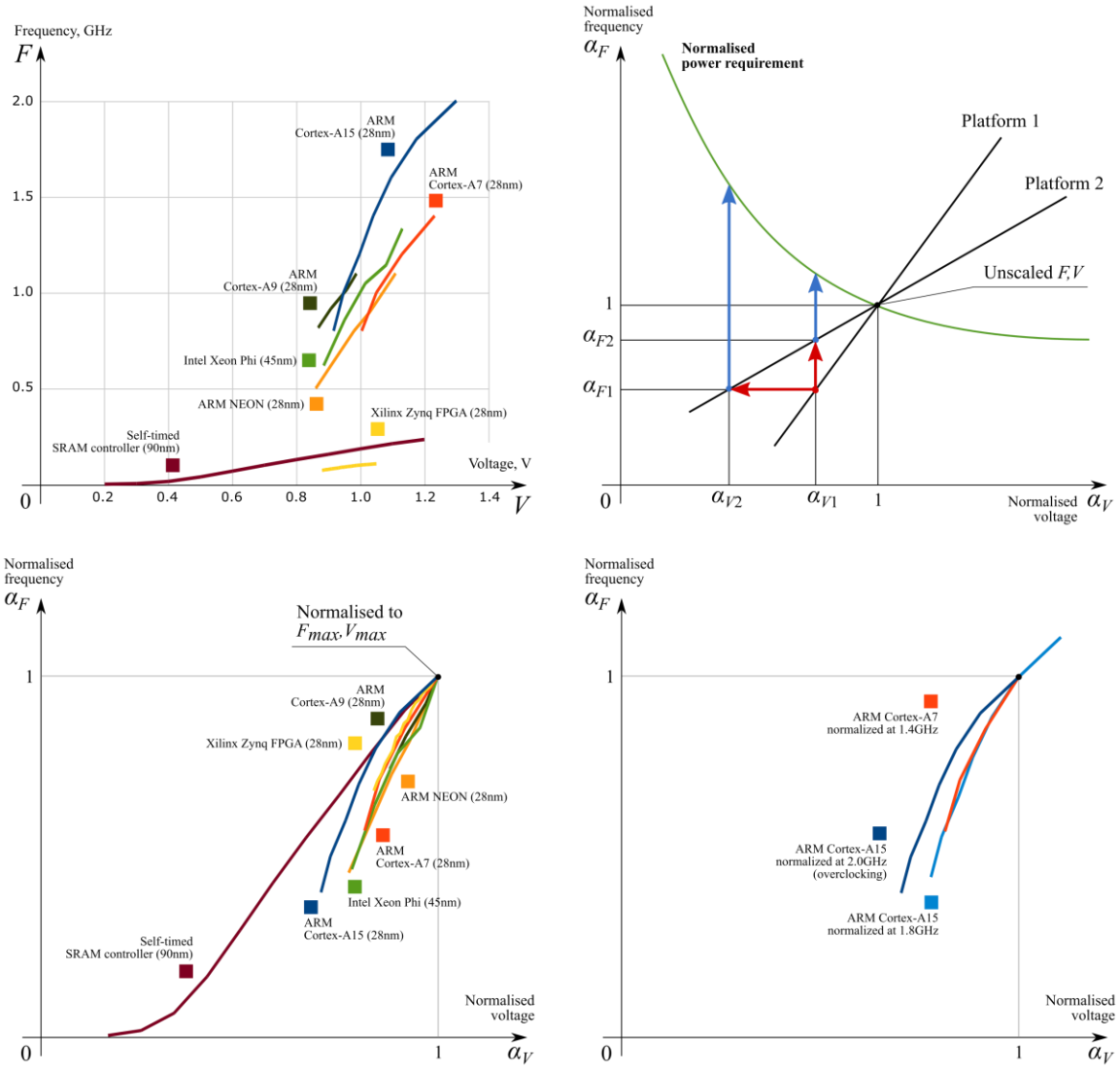


Figure 3 Cross-platform comparison using bi-normalized frequency/voltage space. The top-left figure shows TRLs in the **absolute value F-V space** from experimental data collected from a variety of platforms. For the FPGA, this is the boundary between infinite SNR and non-infinite SNR from Figure 1. This type of cross-platform comparison is not meaningful as the frequencies do not correspond to comparable throughput as the IPC values can differ and different platforms may have different V ranges. For instance, the FPGA, NEON and A9 data, collected from a Xilinx Zynq ZC702 platform running an image-processing application, show that the FPGA has the highest throughput and the NEON also has higher throughput than the A9 [13]. However, the cross-platform comparison of the effectiveness of k-scaling η (see (10)) is possible in a **bi-normalized F-V space**. This concept is shown in the top-right figure, and the data is shown in the bottom-left. The bottom-right figure shows the effect of overclocking on scalability, using ARM big.LITTLE as an example. The non-overclocked $F_{\max}$ for A15 is 1.8GHz, and the corresponding bi-normalized curve matches that of the A7. When overclocked to 2.0GHz, the A15 produces a shallower curve indicating less power-efficiency at $F_{\max} = 2.0$GHz. This means better scalability when scaling away from this operating point than from the nominal $F_{\max}$ of 1.8GHz.

5.2 Frequency vs voltage in practical systems

On the top-left of Figure 3 are F-V scaling curves obtained from the following experiments: ARM Cortex-A7 and A15 executing the Parsec *bodytrack* benchmark and square root computation, Intel Xeon Phi executing Splash-2's *raytrace* and Xilinx Zynq FPGA performing hardware acceleration for

image processing. These commercial systems display higher $V_{\min}$ (much higher than their threshold voltages) and steeper TRLs, resulting in smaller RORs after bi-normalization, as shown in the lower-left of Figure 3. In this range of voltages, switching power dominates.

For any platform, DVFS scaling along different parts of their TRLs may result in different α_V and α_F values. The next section focuses on the implications of this.

5.3 General k-scaling to reduce power and/or improve throughput

A popular measure for system efficiency is power-normalized throughput, which is the amount of computation per unit of energy. The unit for such a measure is instructions per second per watt (IPS/W). Here, this measure is given by θ/P before scaling and θ_k/P_k after scaling.

The effectiveness of DVFS/k-scaling can be measured by comparing the IPS/W figures before and after scaling, i.e., the larger $\frac{\theta_k/P_k}{\theta/P}$ is, the better. From (7)–(9), we can derive

$$\eta = \frac{\theta_k/P_k}{\theta/P} = \frac{(k\alpha_F uF)/(kA\alpha_F F\alpha_V^2 V^2)}{(uF)/(AFV^2)} = \frac{1}{\alpha_V^2} \quad (10)$$

In other words, within the ROR, a smaller α_V provides better IPS/W improvements when considering only switching power and not worrying about specific throughput or power requirements. The tendency would therefore be to scale voltage down as far as possible ($\alpha_V \to \min$). In Figure 3, scaling to α_{V2} is better than scaling to α_{V1} in terms of improving IPS/W. Note that IPC (u) is eliminated from the equation, allowing for further comparison across platforms.

Different platforms may have different TRLs. It is important to investigate the influence of this boundary and other limits of the ROR on the effectiveness of k-scaling.

5.4 Scaling along different TRLs to the same α_V

The following discussion compares two systems scaled to the same α_V – a situation shown in Figure 3 with both platforms scaled to α_{V1}. This happens when reducing α_V is constrained by $V_{\min}$ limits. The platforms' different TRLs lead to different α_F values, α_{F1} and α_{F2}. Both systems achieve the same IPS/W improvements given their α_Vs are the same.

From (8) and (9), we can find P_1 and P_{k1} for system 1 and P_2 and P_{k2} for system 2. These k-scaling operations result in the following power scaling ratios

$$\alpha_{P1} = \frac{P_{k1}}{P_1} = k_1\alpha_{F1}\alpha_V^2,\ \alpha_{P2} = \frac{P_{k2}}{P_2} = k_2\alpha_{F2}\alpha_V^2 \quad (11)$$

If both systems are scaled to the same power scaling ratio – the same eventual power relative to their original power (i.e. a power limit as a percentage of their original power; $\alpha_P = 1$ for no power change) – then $\alpha_{P1} = \alpha_{P2}$, leading to

$$\frac{k_1}{k_2} = \frac{\alpha_{F2}}{\alpha_{F1}} \text{ and } \alpha_{\theta 1} = \alpha_{\theta 2} \quad (12)$$

For both systems, because α_V and α_P are the same, the achievable α_θ is also the same. However, the platform with the greater α_F has a smaller k. With k_1 and k_2 below $k_{\max}$, a smaller k implies using fewer hardware resources. If a platform has a k factor above $k_{\max}$ it cannot consume its entire power budget before exhausting its resources. The conclusion is that, when α_V is the same, the greater α_F is the better.

This result is confirmed by studying the A15 and A7 cores in an ARM big.LITTLE system. They both allow scaling the voltage down from their maximum Vs with a ratio of $\alpha_V \approx 0.8$. At this point, the A7 cores have $\alpha_F \approx 0.57$ and the A15 cores have $\alpha_F \approx 0.7$. With $k_{\max} = 4$ for both core blocks, neither is able to use its entire power budget but with A15 scaled to $k = 4$, to get $\alpha_{P,\mathrm{A15}} = \alpha_{P,\mathrm{A7}}$ we need to scale A7 to $k = 3.6$. The interpolated experimental data shows that $\frac{\theta_{4,\mathrm{A15}}}{\theta_{3.6,\mathrm{A7}}} = \frac{0.7}{0.57} \approx 1.25$.

5.5 Scaling along different TRLs to the same α_{F}

The following discussion compares two systems with different TRLs being scaled to the same frequency scaling ratio α_{F}, as shown in Figure 3 with both systems scaled to α_{F1}. In practice this is related to systems unable to scale below certain frequency values. In this case,

$$\alpha_{P1} = \frac{P_{k1}}{P_1} = k_1 \alpha_F \alpha_{V1}^2,\ \alpha_{P2} = \frac{P_{k2}}{P_2} = k_2 \alpha_F \alpha_{V2}^2 \quad (13)$$

When scaling to the same power scaling ratio, we have

$$k_1 \alpha_{V1}^2 = k_2 \alpha_{V2}^2 \quad (14)$$

This means that the system with the greater α_V will have a smaller k. Since both systems have the same α_F, this leads to a smaller α_θ and smaller throughput gain (i.e. the smaller α_V is, the better).

When scaling to the same throughput scaling ratio α_θ, we have $\alpha_{\theta 1} = \alpha_{\theta 2}$. This leads to

$$k_1 \alpha_F = k_2 \alpha_F,\ \frac{\alpha_{P1}}{\alpha_{P2}} = \frac{\alpha_{V1}^2}{\alpha_{V2}^2} \quad (15)$$

This means that the system with the greater α_V will have a greater power scaling ratio α_P, i.e. after k-scaling it will consume a greater proportion of power compared to before scaling, for a smaller power saving. The conclusion here is therefore the smaller α_V is, the better.

This observation was verified through studying the A15 vs A7 experimental data. Both core blocks allow scaling from 1400MHz down to 800MHz, but A7 gives a smaller α_V (0.815 vs 0.881). The power advantage of this predicted by (15) is confirmed approximately from the experimental data with an error of 6%.

Combining the findings from these investigations, considering only switching power and ideal scaling, from the point of view of extracting either power or performance benefits from k-scaling, scaling should be done to a point with as small as possible an α_V and as large as possible an α_F. The latter requirement means we should always scale along the TRL for any given system.

5.6 Non-ideal scaling and heterogeneity

In an ideal scaling scenario, $\theta_k = k\theta$, but in real-world scenarios this is almost never the case. A software execution may not be entirely parallelizable and many-core hardware may suffer from a number of bottlenecks, most notably shared memory and communication overheads.

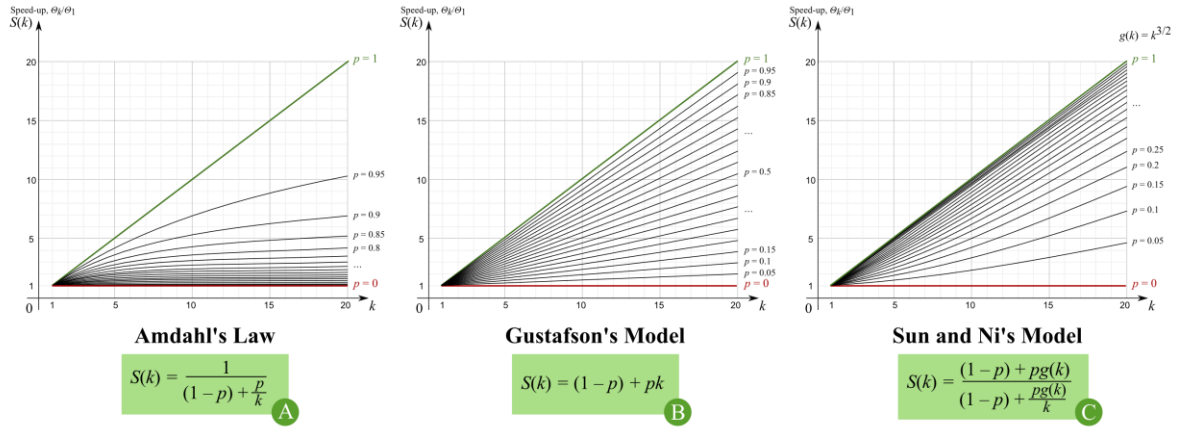
The actual throughput can be found using a speed-up function as shown below:

$$\theta_k = S(k)\theta. \quad (16)$$

Substituting (16) for θ into the ideal scaling equations in the previous sections will expand them to cover general cases of execution.

There are a number of known models for $S(k)$ [3], shown in Figure 4. Amdahl's Law computes the speed-up with k cores assuming a fixed-size workload. Parallelization factor p is the fraction of the

workload executed in parallel; $p = 1$ is the ideal scaling case. The law is famous for predicting that even a small drop in p causes the throughput to quickly saturate [3].



Benchmarks	*blackscholes*	*freqmine*	*bodytrack*	*streamcluster*	*raytrace*
p	0.7817	0.9202	0.9407	0.9857	0.9863
R-Squared	0.9986	0.9344	0.9407	0.9923	0.9457

Figure 4 At the top, non-ideal scaling models. At the bottom, the results of finding p for a number of benchmarks from the Parsec and Splash-2 suites running on an Intel Xeon Phi E5-2650 v1 platform. The values are obtained from experimental data using curve-fitting in Matlab showing reasonable R-Squared metrics.

Gustafson's model argues, however, that it is possible to scale the speedup linearly if the workload size can be increased with the number of cores, increasing the parallelizable portion while keeping the sequential portion the same [3]. Sun and Ni expand this idea towards a general metric $g(k)$, showing how the memory requirement of an algorithm scales relative to the computation requirement, and confirm that for $g(k) \geq k$ it is possible to achieve linear or better-than-linear many-core scaling [3].

Ideally, the parallelization factor p should be a property of the algorithm. However, real-life devices also affect p due to hardware-specific critical sections. Performance profiling can be used to characterize non-ideal scaling (Figure 4).

For systems with heterogeneous computation units (e.g. ARM big.LITTLE with different core types), k-scaling becomes a multi-dimensional optimisation with vector $K = \langle k_1, \cdots, k_X \rangle$ for X types of cores [3].

6 Interplay Exploration Tool

The interplay models presented in the previous sections led to the development of an analysis tool useful for reasoning in the ROR.

6.1 Idle power consideration

In real systems, the power budget is usually required to cover not only switching power but also leakage power. To complicate the issue, not all switching in a system is attributable to any particular computation we want to study (e.g. the power used by an operating system that stays relatively constant whichever application computation is executed). It is sometimes more convenient to group leakage power and any extra switching power not directly related to the computation into the notion

of ‘idle power’, which affects the power budget as shown in the top-left of Figure 5. In this view, we have

$$P = P_c k + P_i \quad (17)$$

where P_c is the power used for computation by a single core and P_i is the system idle power. P_c and P_i can be obtained through the curve-fitting of experimental data.

6.2 Tool description

Scaling and bi-normalized analytical models that include P_c and P_i can be derived in a similar manner to Section 5, however they tend to be very large and impractical. A practical solution is to solve the problem of optimal operating point using discrete numerical solutions, e.g. searching through a limited number of fixed DVFS points and integer k values. This method is implemented in a software tool.

This tool is equipped with experimentally measured data from the platforms mentioned in Section 5.2. In addition, there is an option for the user to provide their own data in CSV format.

The tool can plot the data in θ-V space and find the solution to one of the following problems:

- For the user-specified θ_k, find k, V_k and F_k such that P_k is minimal while the total throughput still satisfies θ_k.
- For the user-specified $P_{\max}$, find k, V_k and F_k such that θ_k is maximized while the total power stays under $P_{\max}$.

The tool supports the three models of non-ideal scaling (Section 5.6) and also allows the tweaking of all relevant parameters such as the parallelization factor.

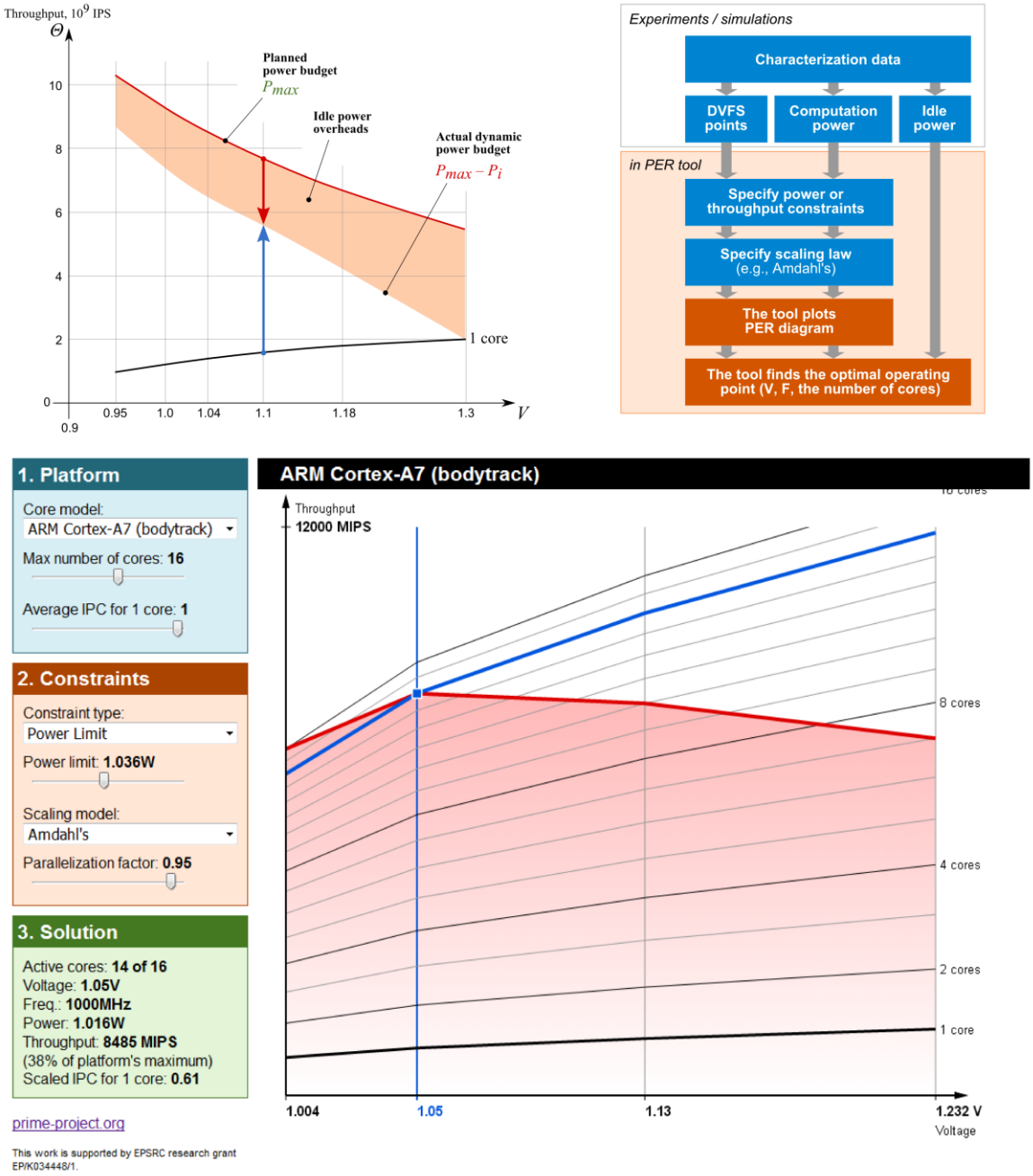


Figure 5 On the top-left, the influence of idle power on the total power budget with data collected from the ARM A15 block of a big.LITTLE platform. The top-right is the work flow for using the presented web-based PER tool for k-scaling and DVFS exploration and a screenshot of the tool can be found at the bottom. The tool is available at http://async.org.uk/prime/PER/per.html.

7 Conclusions

A method of exploring the effectiveness of DVFS and parallelization of systems in the bi-normalized ROR is proposed. The derived metric η can be used to compare proportional improvements of power, throughput and power-normalized throughput from a knowledge of scaling ratios. This bypasses the challenges posed by the very difficult task of modeling power and performance behaviors of systems in the general, execution-independent sense. However, execution-dependent studies are also possible with this method. The analytical metric is suitable in the range of voltages dominated by switching

power where most commercial systems operate and provide quantitative insights into important design metrics such as IPS/W. The leakage power is then considered in a Web-based DVFS/parallelization scaling tool (called PER) that implements a numeric solution to the method and allows for the exploration of throughput, power and reliability interplay as well as parallelization scaling. Wide-ranging experiments with real systems are used to demonstrate the method.

8 Acknowledgements

This work is supported by the EPSRC through the PRiME project (EP/K034448/1). Mokhov's research is additionally supported by the RS through the grant Computation Alive. Aalsaud and Al-Hayanni thank the Government of Iraq for supporting their work with research studentships.

9 Keywords

Energy efficient design; many-core systems; parallel computing; reliability; scalability

10 References

1. S. Mittal. "A Survey of Techniques for Improving Energy Efficiency in Embedded Computing Systems," IJCAET, 6(4), 440–459, 2014
2. J. M. Rabaey. Low Power Design Essentials (Integrated Circuits and Systems). Springer, 2009
3. M. A. N. Al-hayanni, A. Rafiev, R. Shafik, F. Xia. "Power and Energy Normalized Speedup Models for Heterogeneous Many Core Computing," ACSD 2016, Toruń, Poland, June 2016
4. S. Kerrison, K. Eder. 'Energy Modeling of Software for a Hardware Multithreaded Embedded Microprocessor.' ACM TECS, 14. 2015
5. S. Mittal. "A Survey Of Architectural Techniques for Near-Threshold Computing," ACM JETC, 12(4), 2015
6. N. Seifert, M. Kirsch. "Real-Time Soft-Error Testing Results of 45-nm, High-K Metal Gate, Bulk CMOS SRAMs," IEEE Transactions on Nuclear Science, vol. 59, no. 6, pp. 2818-2823, Dec. 2012
7. A. Das, B. M. Al-Hashimi, G. V. Merrett. "Adaptive and Hierarchical Runtime Manager for Energy-Aware Thermal Management of Embedded Systems". ACM TECS 15, 2, Article 24, January 2016
8. K.G. Shin, P. Ramanathan. "Real-time Computing: a New Discipline of Computer Science and Engineering." Proceedings of the IEEE 82 (1): 6–24. 1994
9. A. Fehske, G. Fettweis, J. Malmodin, G. Biczok. "The Global Footprint of Mobile Communications: The Ecological and Economic Perspective," IEEE Communications Magazine, 49(8), 55-62, 2011
10. E. Stott, J. S. J. Wong, P. Y. K. Cheung. "Degradation Analysis and Mitigation in FPGAs," 2010 International Conference on Field Programmable Logic and Applications, Milano, 2010, pp. 428-433
11. A. Baz, D. Shang, F. Xia, A. Yakovlev. "Self-Timed SRAM for Energy Harvesting Systems," Lecture Notes in Computer Science, vol. 6448, 105-115, Springer, 2011
12. J.N. Mistry. Leakage Power Minimisation Techniques for Embedded Processors. PhD Thesis. Univ of Southampton, 2013
13. S. Yang, R. Shafik, G.V. Merrett, E. Stott, J. Levine, J. Davis, B. Al-Hashimi. "Adaptive Energy Minimization of Embedded Heterogeneous Systems using Regression-based Learning," PATMOS 2015, Salvador, Brazil, Sept. 2015

11 Author bios

Fei Xia is a Senior Research Associate with the School of Electrical and Electronic Engineering, Newcastle University. His research interests are in asynchronous and concurrent systems with an

emphasis on power consumption. Fei holds a PhD from King's College, London, an MSc from the University of Alberta, Edmonton, and a BEng from Tsinghua University, Beijing.

Ashur Rafiev received his PhD in 2011 from the School of Electrical and Electronic Engineering, Newcastle University. He currently works in the School of Computing Science, Newcastle University, as a Research Associate. His research interest is focused on power modelling and hardware-software co-simulation of many-core systems.

Ali Aalsaud (Student Member, IEEE and IET) is a Lecturer in Computer Engineering at Al-Mustansiriyah University in Baghdad-Iraq, from which He received both the BSc and MSc degrees in electronic engineering. He is currently on leave, studying for a PhD with the School of Electrical and Electronic Engineering, Newcastle University. His research interests are in power/performance of many-cores heterogeneous systems using novel architectures, efficient algorithms and emerging technologies.

Mohammed Al-hayanni is an experienced computer and software engineer. He is currently studying for his PhD with the School of Electrical and Electronic Engineering, Newcastle University. His research interests include developing practically validated robust performance adaptation models for energy-efficient many-core computing systems.

James Davis (Member, IEEE) is a Research Associate with the Department of Electrical and Electronic Engineering at Imperial College London. His current research concerns the runtime monitoring and adaptation of digital electronic hardware for energy efficiency and reliability. James received the MEng and PhD degrees in Electrical and Electronic Engineering from the aforementioned department in 2011 and 2016, respectively.

Joshua Levine is currently a Research Associate in the Department of Electrical and Electronic Engineering, Imperial College London. He completed his PhD and MEng degrees in the same department in 2014 and 2009. His research interests include field-programmable gate arrays, reliability, resilience and ageing, and the development and application of circuit-level knobs and monitors for runtime adaptation.

Andrey Mokhov is a Lecturer in Computer Engineering at Newcastle University. His PhD dissertation (Newcastle University, 2009) introduced Conditional Partial Order Graphs as a compact model for concurrent and multi-mode systems. His current research interests are in applying formal methods and functional programming to software and hardware design.

Alexander Romanovsky is a Professor at the School of Computing Science, Newcastle University, where he leads the Secure and Resilient Systems Group, and a Visiting Professor at NII, Japan. He received an MSc. degree in Applied Mathematics from Moscow State University and a PhD degree in Computer Science from St. Petersburg State Technical University. His main research interests are system dependability, fault tolerance, software architectures, safety verification/assurance, system structuring and verification of fault tolerance.

Rishad Shafik (Member, IEEE) is a Lecturer in Electronic Systems at Newcastle University. His research interests include the design of intelligent and energy-efficient embedded systems. He holds a PhD and an MSc from The University of Southampton., and a BEng from IUT, Bangladesh. He has authored 80+ research articles published by IEEE/ACM, and is the co-editor of “Energy-efficient Fault-Tolerant Systems”. He is chairing DFT’17 (http://www.dfts.org), to be held in Cambridge, UK.

Alex Yakovelv (Senior Member, IEEE and Fellow, IET) is a Professor in the School of Electrical and Electronic Engineering, Newcastle University. His research interests include asynchronous circuits and systems, concurrency models, and energy-modulated computing. Yakovlev received a DSc in Engineering at Newcastle University. In 2011-2013 he was a Dream Fellow of the UK Engineering and Physical Sciences Research Council (EPSRC).

Sheng Yang (Member, IEEE) received his BEng degree in Electronic Engineering in 2008 and his PhD degree in Electronics Engineering in 2013, both from the University of Southampton, UK. Currently he is a Research Engineer with the Applied Silicon Group, ARM R&D Cambridge. His research interests include low power embedded system design, signal processing and machine learning.

12 Author emails

fei.xia@newcastle.ac.uk

ashur.rafiev@newcastle.ac.uk

a.m.m.aalsaud@newcastle.ac.uk

m.a.n.al-hayanni@newcastle.ac.uk

james.davis06@imperial.ac.uk

josh.levine05@imperial.ac.uk

andrey.mokhov@newcastle.ac.uk

alexander.romanovsky@newcastle.ac.uk

rishad.shafik@newcastle.ac.uk

alex.yakovlev@newcastle.ac.uk

sheng.yang@arm.com