2026 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) | 979-8-3315-5103-2/26/$31.00 ©2026 IEEE | DOI: 10.1109/ISPASS69572.2026.00054

ASI: A Unifying Metric to Assess and Improve Processor Microarchitecture Sustainability

Jaime Roelandts
Ghent University
Ghent, Belgium
Jaime.Roelandts@UGent.be

Ajeya Naithani
TU Eindhoven
Eindhoven, The Netherlands
a.naithani@tue.nl

Lieven Eeckhout
Ghent University
Ghent, Belgium
Lieven.Eeckhout@UGent.be

***Abstract*—Computing systems contribute significantly to global carbon emissions. Designing sustainable processor microarchitectures is particularly challenging due to the huge design space, the complex objective function, and the inherent data uncertainty.**

We introduce the Architectural Sustainability Indicator (ASI), a novel metric for evaluating the sustainability of processor microarchitectures. ASI enables classifying a design in different sustainability regions: strongly, weakly or unsustainable. ASI's key strength lies in its ability to offer insights for how to transform unsustainable and weakly sustainable designs into strongly sustainable ones. Moreover, ASI can be used to guide automated design space exploration towards microarchitecture configurations that optimally trade off sustainability for performance. We analyze the effect of design decisions, such as the dominance of embodied versus operation footprint on ASI, and demonstrate that optimizing ASI differs from optimizing other metrics such as energy or energy-delay-area-product (EDAP).

I. Introduction

Sustainability is a major challenge for our society and generation due to its impact on global warming and the continuous quest for raw materials. Information and Communication Technology (ICT) is responsible for 2.1% to 3.9% of global warming — on par with the aviation industry — and this contribution is expected to continue to increase substantially in the near future [18]. Without a significant and continuous effort from our community to reduce computing devices' carbon footprint, the sustainability gap between ICT's actual contribution to global warming versus the Paris agreement will continue to grow [16].

Even though energy- and power-efficient computing has been a 'hot' topic for several decades now [36], only recently have computer architects realized that improving the environmental footprint of a computing device involves addressing both its embodied and operational footprint [22], [23], [25]. The former refers to the upfront environmental footprint due to manufacturing, while the latter refers to the environmental footprint due to device use during its lifetime. While the embodied footprint dominates for mobile devices, the operational footprint tends to dominate for connected always-on devices [22], [23]. Carbon models have been proposed to assess a device's carbon footprint at design time, including the bottom-up data-driven GreenChip [25] and ACT [20], [21] models, as well as the top-down first-order FOCAL model [14], [15].

Designing sustainable processor microarchitectures is a major challenge because of the vast and complex design space, i.e., there are many design parameters that all impact performance, power, energy, chip area, and thus ultimately a design's environmental footprint. To make things worse, the rebound effect [3] may turn a presumably sustainable design into an unsustainable one as efficient designs tend to be used more intensively and thus perform more work, which in turn leads to an increased environmental footprint. How to guide computer architects towards sustainable design points is an open challenge.

This paper presents the *Architectural Sustainability Indicator (ASI)* [35] to help computer architects assess whether a microarchitectural design is strongly, weakly or unsustainable compared to a reference design. Weakly sustainable means that a design is subject to a rebound effect, i.e., while a more efficient design may reduce the footprint under a fixed-work scenario, it may lead to an increased footprint under a fixed-time scenario when more work is done. Strongly sustainable means that a design is robust to rebound effects, i.e., a design reduces the footprint under both the fixed-work and fixed-time scenarios. While FOCAL [14], [15] introduced the notion of strongly, weakly and unsustainable designs, it did not provide a metric that captures a design's sustainability profile in a single number. Visualizing ASI versus speedup divides the design space in sustainability regions: strongly, weakly, and unsustainable. A concrete design point's position in this space provides insight into how to change the design to make it strongly sustainable.

The real power of ASI is that it enables automatically exploring the processor microarchitectural design space while trading off sustainability versus performance. To illustrate this, we perform ASI-guided design space exploration while considering four representative microarchitectural techniques: Out-of-Order (OoO) core, In-Order (InO) core, Indirect Memory Prefetcher (IMP), and Scalar Vector Runahead (SVR). For each technique, the designer provides a set of values for different microarchitectural parameters such as L1-D cache size, last-level cache (LLC) size, processor width, etc. An iterative algorithm starts with a reference design, and alters one parameter at a time, eventually delivering a set of Pareto-optimal points in the ASI versus speedup space. Such Pareto-optimal points offer immediate insight into how to adjust the

979-8-3315-5103-2/26/$31.00 ©2026 IEEE
DOI 10.1109/ISPASS69572.2026.00054

microarchitectural structures for a significant improvement in sustainability while incurring minimal performance degradation. For example, we notice that the sustainability of SVR can be significantly improved from 1.2× to 2.7×, while minimally degrading its speedup from 2.1× to 2× relative to a reference design. To the best of our knowledge, this is the first work to perform and demonstrate automated exploration to identify strongly sustainable processor designs. We also compare ASI against energy and energy-delay-area-product (EDAP), and show that these metrics do not yield the full spread of Pareto-optimal sustainable designs, specifically missing out on some of the strongly sustainable designs.

II. The FOCAL Model

When assessing the sustainability of a device, the typical solution is to perform a Life Cycle Assessment (LCA) [5]. This method calculates all the emissions of a device, including the emissions from manufacturing, use, and recycling of the device. However, LCAs and derived methods require data that is either uncertain or may not be available. To overcome this problem, FOCAL [14], [15] introduces *Normalized Carbon Footprint (NCF)* to quantify the carbon impact of a system *X* relative to a reference system *Y* using proxies. NCF is calculated as a weighted sum of the normalized embodied and operational footprints of the system. The weighting factor α ($0 \leq \alpha \leq 1$) reflects the relative importance of these two components: for example, mobile devices typically have a high embodied footprint ($\alpha \simeq 0.8$), while always-on systems are more influenced by the operational footprint ($\alpha \simeq 0.2$) [22], [23]. Other factors, such as carbon intensity, also affect α. For example, if the chip is *powered on* using green energy, α will be higher, reflecting higher emphasis on the embodied footprint. On the other hand, if the chip is *manufactured* using green energy, α will be lower, as the operational footprint will have a higher impact on the overall footprint.

To address data uncertainty, FOCAL relies on proxies when comparing processors manufactured in the same technology node. Specifically, it uses *chip area (A)* to represent the embodied footprint, and either *energy (E)* or *power (P)* to represent the operational footprint — depending on the evaluation scenario. In a *fixed-work* scenario, where the total work performed is assumed constant, energy (E) is used. In contrast, a *fixed-time* scenario assumes constant execution time, meaning that a more efficient system performs more work, and power (P) is used as the operational proxy. This distinction anticipates the rebound effect, also known as Jevons' paradox [3].

Equations (1) and (2) report NCF for the fixed-work and fixed-time scenarios, respectively, with $A_n = \frac{A_X}{A_Y}$ normalized chip area, and P_n and E_n normalized power and energy, respectively.

$$NCF_{fw} = \alpha \times A_n + (1-\alpha) \times E_n \tag{1}$$

$$NCF_{ft} = \alpha \times A_n + (1-\alpha) \times P_n \tag{2}$$

Based on the results of the different scenarios, FOCAL classifies a system into one of three categories [14], [15]. A system *X* is called *strongly sustainable* relative to system *Y* when it reduces the carbon footprint under both the fixed-work and fixed-time scenarios, thus $NCF_{fw} < 1$ and $NCF_{ft} < 1$. A system is classified as *weakly sustainable* if it reduces the carbon footprint under only one of the two scenarios, i.e., $NCF_{fw} < 1 < NCF_{ft}$ or $NCF_{fw} > 1 > NCF_{ft}$. A system is *unsustainable* if the carbon footprint increases under both the fixed-work and fixed-time scenarios, i.e., $NCF_{fw} > 1$ and $NCF_{ft} > 1$.

III. The Architectural Sustainability Indicator

The Architectural Sustainability Indicator (ASI) enables computer architects to assess whether a design is strongly, weakly or unsustainable using a single metric, and more importantly it provides insight and hints to improve the sustainability of a design. Recall from Section II that a design is strongly, weakly or unsustainable depending on whether the value of NCF_{fw} and NCF_{ft} is larger or smaller than one as denoted below in Equation (3). Using the symbols α, A_n and P_n as defined in Section II, and by defining T_n as normalized execution time or $T_n = T_X/T_Y$, Equations (4) to (6) rewrite the conditions as follows:

$$\begin{cases} NCF_{fw} \lesseqgtr 1 \\ NCF_{ft} \lesseqgtr 1 \end{cases} \tag{3}$$

$$\iff \begin{cases} \alpha A_n + (1-\alpha) P_n T_n \lesseqgtr 1 \\ \alpha A_n + (1-\alpha) P_n \lesseqgtr 1 \end{cases} \tag{4}$$

$$\iff \begin{cases} (1-\alpha) P_n T_n \lesseqgtr 1 - \alpha A_n \\ (1-\alpha) P_n \lesseqgtr 1 - \alpha A_n \end{cases} \tag{5}$$

$$\iff \begin{cases} T_n \lesseqgtr \frac{1-\alpha A_n}{(1-\alpha)P_n} \\ 1 \lesseqgtr \frac{1-\alpha A_n}{(1-\alpha)P_n} \end{cases} \tag{6}$$

$$\iff \begin{cases} T_n \lesseqgtr ASI \\ 1 \lesseqgtr ASI \end{cases} \tag{7}$$

We now define ASI as

$$ASI = \frac{1 - \overbrace{\alpha A_n}^{\text{weighted normalized embodied footprint}}}{\underbrace{(1-\alpha)P_n}_{\text{weighted normalized operational footprint}}} \tag{8}$$

which is a *higher-is-better* metric, measuring sustainability relative to a reference design with ASI = 1. Whether a design is strongly, weakly or unsustainable can now be derived from analyzing the ASI value:

Strongly sustainable: A system is strongly sustainable if its normalized footprint under both the fixed-work and fixed-time scenarios is smaller than the reference system. In other words, $NCF_{ft} < 1$ and $NCF_{fw} < 1$. This implies that $1 < ASI$ and $T_n < ASI$, or $ASI > \max(1, T_n)$.

Unsustainable: A system is unsustainable if $NCF_{ft} > 1$ and $NCF_{fw} > 1$. This implies that $1 > ASI$ and $T_n > ASI$, or $ASI < \min(1, T_n)$.

Sustainable under fixed work (S-FW): This is a system that improves sustainability under a fixed-work scenario but not

TABLE I: Classifying sustainability of a system using ASI.

Strongly sustainable	$ASI > \max(1, T_n)$
Unsustainable	$ASI < \min(1, T_n)$
Weakly sustainable	$\min(1, T_n) < ASI < \max(1, T_n)$

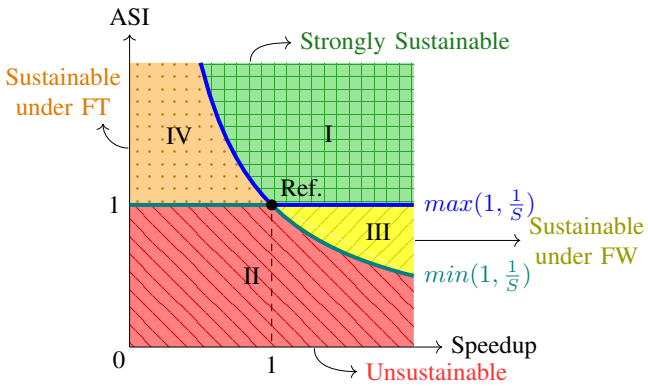

Fig. 1: This graph depicts the relationship between ASI and speedup S defined as $S = 1/T_n$. The four regions denote whether a system is strongly, weakly, or unsustainable with respect to the reference system.

under a fixed-time scenario. This happens when performance improves while reducing energy, but at the cost of increased power consumption. The inequalities become $NCF_{ft} > 1$ and $NCF_{fw} < 1$. This implies that $1 > ASI$ and $T_n < ASI$, or $T_n < ASI < 1$.

Sustainable under fixed time (S-FT): This is a system that improves sustainability under a fixed-time scenario but not under a fixed-work scenario. This can happen when the reduction in power consumption is less than the increase in speedup, which results in higher energy consumption. The inequalities become $NCF_{ft} < 1$ and $NCF_{fw} > 1$, which implies that $1 < ASI$ and $T_n > ASI$, or $T_n > ASI > 1$.

Note that the latter two categories denote *weakly sustainable* systems because they improve sustainability in either the fixed-work or fixed-time scenario, but not both. Combining both categories of weakly sustainable systems with respective conditions, $T_n < ASI < 1$ and $T_n > ASI > 1$, we obtain $\min(1, T_n) < ASI < \max(1, T_n)$. Interestingly, these boundary conditions are identical to the ones for the strongly and unsustainable designs, which leads to the overall summary as presented in Table I.

Figure 1 illustrates the relationship between ASI (vertical axis) versus speedup ($S = 1/T_n$ along the horizontal axis). It depicts the different sustainability regions of a new system *X* versus a reference system *Y*, which is represented by the reference point $(1, 1)$. The blue line denotes the $\max(1, T_n)$ boundary condition while the green line denotes the $\min(1, T_n)$ boundary condition. We identify four regions: (I) the top right region (green) is the strongly sustainable region, (II) the bottom left region (red) is the unsustainable region, (III) the right-most region (yellow) represents weakly sustainable designs under the fixed-work scenario (S-FW), and (IV) the top-left region (orange) denotes weakly sustainable designs under the fixed-time scenario (S-FT).

The strength of this representation lies in its ability to guide computer architects where to focus on for improving a design's sustainability. To illustrate this, we analyze four sample microarchitecture configurations presented in Figure 2. The reference design is the baseline OoO core from Table II; the four design points highlighted in Figure 2 are the designs listed in Table III. We further assume $\alpha = 0.8$ to reflect that the embodied footprint dominates, as is common for low-power mobile processors [22], [23].

Region I (green) represents strongly sustainable designs. The performance impact is possibly small compared to the reference design. A design in Region II (red) is unsustainable. To turn such a design into a strongly sustainable one, ASI needs to be improved at the very least, i.e., a performance boost alone cannot achieve this. In other words, chip area and/or power consumption need to be reduced. A design in Region III (S-FW) is weakly sustainable, i.e., while consuming less energy under a fixed-work scenario compared to the baseline, the design is subject to a rebound effect, and improving ASI is essential to make the design strongly sustainable. Finally, Region IV (S-FT, orange) includes designs that reduce area and power but suffer from a severe performance loss, leading to higher energy consumption. Here, ASI is already favorable, but performance must improve to reach strong sustainability.

Although ASI is derived from FOCAL, it expresses the overall sustainability of an architecture as a single number. This makes it intuitive to draw insight in the sustainability versus performance trade-off space, allowing computer architects to reason about the impact of trading performance for sustainability, and vice versa. For example, reduced cache size decreases chip area and power consumption, and thereby improving sustainability. However, performance will possibly degrade, placing the design in the second quadrant (sustainability regions `I` or `IV`) of Figure 1. Thanks to its unified sustainability metric, ASI facilitates evaluation in automated design-space exploration (see Section VI).

We recognize that sustainability is a multi-dimensional challenge, and ASI's purpose is to evaluate sustainability at the earliest stages of the design cycle, where it is inherently difficult, if not impossible, to precisely predict factors such as the exact amount of ultra-pure water required or the specific energy sources and raw materials used or chemicals emitted during manufacturing. Moreover, recent research on semiconductor sustainability [7], [19], [32] reports that several manufacturing resources (e.g., carbon, ultra-pure water, chemicals and gases, raw materials) scale with chip area, and thus correlate with carbon. On the operational side, there is inherent uncertainty due to variations in carbon intensity upon device use, which is why ASI includes the α parameter to weigh the embodied versus operational footprint. Having said that, ASI does not capture the sustainability impact a device may have on water stress nor spatial shifting of workloads upon deployment in concrete use cases [1], [24]. Overall, ASI is a measure for the

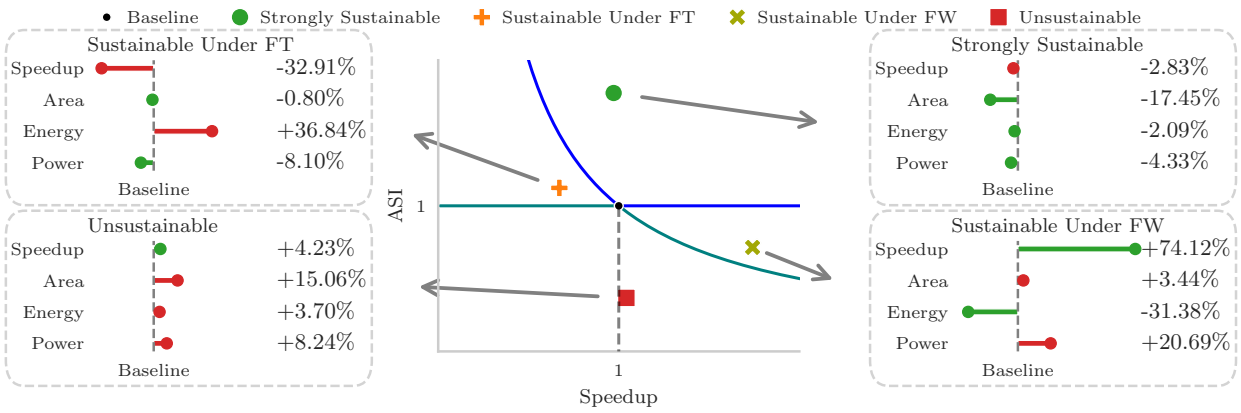

Fig. 2: Four processor design points, each belonging to a different sustainability region, correspond to the configurations listed in Table III. For each point, we show the impact on performance, chip area, energy, and power relative to the baseline.

TABLE II: The baseline OoO and InO cores configured after the Arm Cortex A510 [41].

Core	**Out-of-Order**	**In-Order**
Frequency Dispatch/commit width	2.0 GHz 3 Instr/cycle	
ROB/Scoreboard Load/store queue Reservation station	32 16 16	32 — —
Branch Predictor	hybrid local/global predictor, 10-cycle misprediction penalty	
Address translation	4 page table walkers 16-entry fully assoc. D-TLB 16-entry fully assoc. I-TLB 2048-entry 8-way S-TLB	
L1-I cache	64 KiB, 64 B cacheline, 4-way	
L1-D cache	64 KiB, 64 B cacheline, 4-way, 16 MSHRs, stride prefetcher	
L2 cache	512 KiB, 64 B cacheline, 8-way	
DRAM	50 GiB/s bandwidth, 45 ns latency	

TABLE III: Different core configurations derived from the baseline OoO core to illustrate the four regions in Figure 2. Differences with the baseline are highlighted in bold.

	Baseline	Unsustainable	Sustainable Under FT	Sustainable Under FW	Strongly Sustainable
L1-D size (KiB)	64	**128**	64	**32**	**32**
L1-I size (KiB)	64	**128**	64	**32**	**32**
LLC size (KiB)	512	**1024**	**256**	**1024**	**128**
Processor width	3	**4**	**4**	3	**2**
ROB entries	32	32	**16**	**64**	32
Reservation station	16	16	**8**	**32**	16
ASI ($\alpha = 0.8$)	1	**0.37**	**1.12**	**0.71**	**1.77**

sustainability impact that correlates with carbon.

IV. Experimental Setup

1) Baseline Processor Core: Table II summarizes our baseline processor configuration modeled after the Arm Cortex A510 [41] which is a representative high-efficiency, low-power core in the mobile and embedded space. We evaluate the performance of different architectures using Sniper v8.1 [12], and estimate chip area and power consumption using McPAT v1.0 [26] assuming a 22 nm technology node.

2) Benchmarks: We consider two sets of benchmarks: (1) Five benchmarks from the GAP suite [8]: Between Centrality, Breadth First Search, Connected Components, Page Rank, and Single-Source Shortest Path. We use a real-world input from the LiveJournal website, and two synthetic inputs: a Kronecker graph, and a uniform random distributed graph. (2) Eight benchmarks from the database and high-performance computing (HPC) domains, including camel [2], integer sort and conjugated gradients from the NAS parallel benchmarks [6], kangaroo [2], randacc from the HPC Challenge [27], hash-join [9] with a bucket size of 2 and 8, and seq-CSR from the Graph500 benchmark suite [4]. Overall, we simulate 23 workloads, and each workload is simulated in detail for 200 M instructions in the representative region of interest.

3) Microarchitecture Techniques: Next to the InO and OoO baseline cores, we evaluate two techniques running on top of the InO baseline: Indirect Memory Prefetcher (IMP) and Scalar Vector Runahead (SVR). IMP [42] captures the memory access patterns at the L1 cache, and attempts to derive the indirect memory addresses from the base address of the initially accessed array with simple addition or bit-shift operations. While IMP captures simple indirect patterns, it fails for complex address calculations. SVR [33], [34] is a lightweight in-core prefetching technique that follows the dynamic instruction stream to prefetch chains of indirect memory accesses. It achieves significant speedups on low-power InO cores by issuing speculative prefetches. (SVR builds upon Vector Runahead (VR) [28], [29] and Decoupled Vector Runahead (DVR) [30], [31], adapted to energy-constrained cores.)

TABLE IV: Configuration parameters during design space exploration towards the ASI-speedup Pareto front. The baseline values are indicated in bold.

L1-D cache (KiB)	32, **64**, 128
LLC cache (KiB)	64, 128, 256, **512**, 1024
Processor width	2, **3**, 4
ROB size	16, **32**, 64
IMP prefetch count	8, **16**, 32
SVR vector length (#scalars)	8, **16**, 32, 64, 128

V. ASI-Guided Design Space Exploration

As discussed in Section III, the overall sustainability of a processor design depends on both ASI and speedup; ASI, in turn, is affected by a design's power consumption and incurred chip area. The goal of this section is to perform design space exploration and offer insight into automatically (fine-)tuning a design for the best ASI versus speedup trade-off.

We begin with a reference baseline and allow our search algorithm to suggest a design point that provides either better ASI or better performance, relative to *any* other design point, i.e., deliver a set of Pareto-optimal points in the ASI versus speedup design space. Such an analysis — and including sustainability as one of the design goals — leads to chips that lower carbon footprint while having a *minimal* impact on other design goals such as performance, power, energy, etc.

A. Background: Pareto Front

We determine the Pareto front (or skyline) in the ASI versus speedup landscape in a manner similar to Spea2 [44]. While optimizing multiple objective functions in Spea2, a point is located on the Pareto front if there is no other point that is better for *all* the objective functions. In the ASI versus speedup landscape, a point therefore is on the Pareto front if it offers either higher ASI than any other point with a higher speedup, or vice versa, higher speedup than any other point with a higher ASI. Figure 3 shows that points A and B are both on the Pareto front, since point A has a higher ASI but point B has a higher speedup. Point C, on the other hand, has a lower ASI and speedup than point B. Therefore, point C is not on the Pareto front.

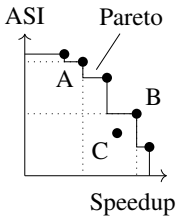

Fig. 3: Points A and B are Pareto-optimal while C is not because both ASI *and* speedup are inferior compared to B.

B. Automated Exploration of the Pareto Front

We construct the Pareto front using an algorithm that iteratively finds points on the front from a given set of configurations. We vary the architecture configuration parameters listed in Table IV, and the algorithm automatically converges to the configurations on the Pareto front.

At the start of the search process, we create a *search-set* to hold the configurations to be evaluated; a *Pareto-set* to track the current Pareto front; and a *discarded-set*, which helps determine whether a configuration has already been evaluated in previous iterations. To bootstrap the process, we evaluate the baseline configuration to compute both its performance and ASI. Once this is done, the baseline is added to the *Pareto-set*.

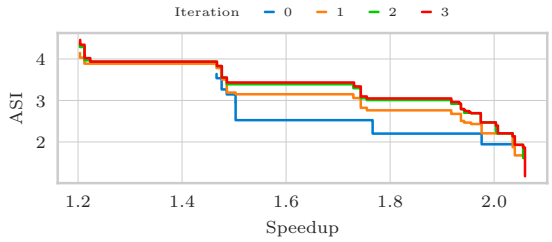

Fig. 4: The Pareto front for SVR assuming $\alpha = 0.8$, converges with every iteration, sometimes already settling on the front before the last iteration. This graph is zoomed in for better clarity.

At the beginning of each iteration, we generate new experiments from the configuration that were newly inserted into the *Pareto-set* during the previous iteration. For each of these configurations, we vary one parameter at a time — only selecting from those that have not yet been modified — and check whether the resulting configuration is already present in either the *Pareto-set* or the *discarded-set*. If it is not, the configuration is added to the *search-set*. For example, we might vary only the L1-D cache size to 32 or 128, keeping all other parameters unchanged, or we might vary only the LLC size to 512 while keeping the default L1-D size. However, if the parent configuration from the *Pareto-set* has already modified the L1-D cache size, we do not vary it again; this parameter remains fixed for all of its successors.

The new core configurations in the *search-set* are then evaluated for all benchmarks; both ASI and speedup are calculated with respect to the baseline. The Pareto front is constructed using the points from the current *search-set* and any existing points in the *Pareto-set*. Configurations on the resulting Pareto front are added (or returned) to the *Pareto-set*, while all others are placed in the *discarded-set* for future reference.

This completes the iteration with the most recent Pareto front stored in the *Pareto-set*. The process is repeated for the remaining parameters and terminates when the *search-set* is empty or when a threshold number (= 10 in our setup) of iterations is reached. By construction, the Pareto front continuously improves. Indeed, in Figure 4 we show the Pareto front advancing at the end of every iteration for SVR, demonstrating the effectiveness of the search algorithm.

VI. Identifying Strongly Sustainable Designs

We construct Pareto fronts for four processor architectures: (1) OoO, (2) InO, (3) IMP [42], and (4) SVR [33], [34]. We analyze ASI and speedup for all the architectures relative to the baseline OoO core, see Figure 5. The key takeaway from this analysis is that the area-efficient designs such as

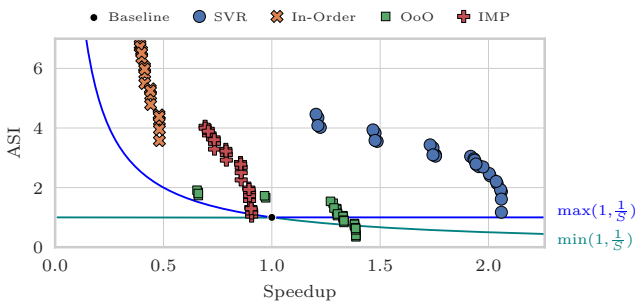

Fig. 5: The Pareto front for InO, OoO, IMP, and SVR, assuming $\alpha = 0.8$.

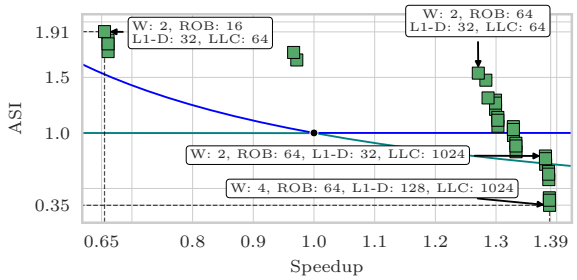

Fig. 6: The Pareto front for the OoO core (zoomed in from Figure 5).

the InO core, IMP, and SVR are strongly sustainable; SVR additionally delivers a high performance boost on top of the underlying InO core. The Pareto-optimal points for the OoO core span three sustainability regions. There is potential for a large improvement in ASI with a relatively small impact on performance for all the architectures.

Out-of-Order (OoO) Core. The Pareto front for the OoO core lies in three sustainability regions, see Figure 6 for a zoomed-in view. Relative to the baseline, ASI varies between 0.3× to 1.9× while performance varies between 0.7× to 1.4×. The highest speedup is achieved when the LLC equals 1024 KiB, L1-D is 128 KiB, the re-order buffer (ROB) has 64 entries, and the processor width is increased to four. All these factors are obviously suited for achieving high degrees of instruction-level parallelism (ILP) and memory-level parallelism (MLP). However, on the downside, this configuration yields the lowest ASI, thereby rendering the design unsustainable.

The Pareto front, however, offers significantly better ways to design a more sustainable OoO core while incurring minimal performance degradation. By reducing the L1-D cache size to 32 KiB and processor width to two, the core already transitions to a weakly sustainable design with a performance degradation of *only* 0.5% relative to the best performing design. By further reducing these structures, the core becomes *strongly* sustainable — delivering an ASI 1.5× higher than the baseline — with a degradation in performance, from 1.4× to 1.3×,

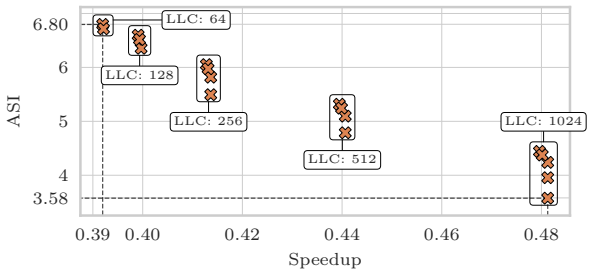

Fig. 7: The Pareto front for the InO core (zoomed in from Figure 5).

which is relatively small. The other points on the Pareto front provide opportunities to trade off performance for a more sustainable design. These examples highlight the potential for improving the sustainability of an architecture with a negligible degradation in performance.

In-Order (InO) Core. As expected, the InO core is inferior to the OoO baseline in terms of performance. However, on the flip side, the simpler design of the InO core incurs smaller power consumption and chip area leading to all the points on the Pareto front being situated in the strongly sustainable region, see Figure 7. The points with a lower ASI incur a larger chip area and yield higher performance. This is intuitive, as increased area, for example larger caches, leads to better performance. There are five groups of points, one per LLC size. Within each group, ASI and performance is affected by the size of the L1-D cache. Processor width negligibly impacts performance as the core is not able to exploit additional ILP with increasing width.

Across all the InO design points, the increase in ASI from the worst to the best design is significantly higher than the increase in performance: the range for ASI amounts to 1.9× (from 3.58 to 6.80), while the performance range is limited to 1.2× (from 0.39 to 0.48). Therefore, as InO cores are typically suited for edge devices where energy efficiency is the key optimization metric, deploying a core with reduced caches can bring a large reduction in the overall carbon footprint with minimal impact on performance. Obviously, the reduced area also reduces power consumption, further reducing the environmental footprint of the core.

Indirect Memory Prefetcher (IMP). The Pareto front for IMP shows that the optimal points belong to two sustainability regions, see Figure 8. The obvious distinction with the InO core is that IMP delivers substantially higher performance. Between the top-left and the bottom-right points on the Pareto front, performance increases by 1.3× while ASI decreases by 3.9×. The performance gain increases with increasing LLC size and prefetch count. However, the increase in power consumption due to prefetching and larger area of caches also lowers ASI. In fact, the bottom-right point on the Pareto front for IMP lies in the weakly sustainable region due to the increased power consumption from its large LLC and

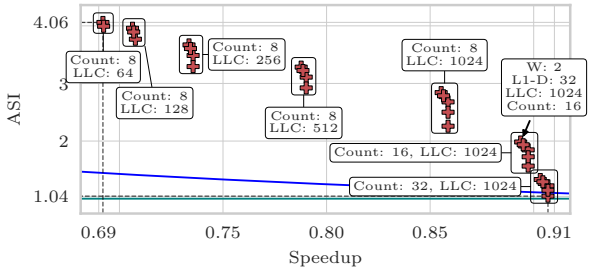

Fig. 8: The Pareto front for the IMP core (zoomed in from Figure 5). With two groups of prefetch count 16 and 32 on the right. The other groups all have a prefetch count of 8, but the LLC has the biggest impact on speedup.

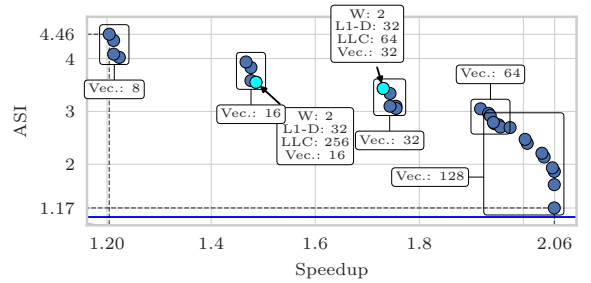

Fig. 9: The Pareto front for the SVR core (zoomed in from Figure 5).

the increased activity of the prefetcher to generate 32 future accesses. The straightforward way to increase the sustainability of this design is to lower both the size of the LLC and the number of IMP prefetches. However, if the goal is to achieve high performance while being strongly sustainable, one direction is to maintain the 512 KiB LLC and generate 8 prefetches.

All points on the Pareto front for IMP offer a more sustainable design than the baseline: ASI improves from 1.15× to 4.06× while performance varies between 0.69× and 0.91×. The large change in ASI for a small degradation in performance means a designer can select a far more sustainable architecture without much degradation in performance. For example, it is possible to achieve 2× higher ASI than the baseline while staying within 0.99× from the best performing IMP design; this is the architecture that generates 16 prefetches and has a 1024 KiB LLC.

Scalar Vector Runahead (SVR). For SVR, the vector length determines the number of speculative memory accesses generated for each load instruction belonging to the chain of indirect memory accesses. We separate out the Pareto front for only SVR in Figure 9. Foremost, we note that SVR is always strongly sustainable, and the ASI is 1.2× to 4.5× higher than the baseline. All the points on the front deliver better performance, delivering a speedup in the range of 1.2× to 2.1× over the baseline. The points on the Pareto front are primarily separated into five groups. Each group belongs to one of the five vector lengths, and the performance of SVR increases with vector length. There is a large variation in ASI within the group of points with a vector length of 128. This again presents an opportunity to substantially improve ASI without (significant) performance loss. For the same vector length of 128, ASI improves from 1.2× to 2.7× while degrading performance from 2.1× to 2×, relative to the baseline. ASI improves primarily due to a reduction in the size of the caches and the width of the processor. As SVR is implemented on top of an InO core, there is not a large performance degradation due to the downsizing of these structures. The other points on the Pareto front provide further opportunity to select an SVR design with a different performance versus ASI trade-off.

Figure 9 highlights two design points to illustrate how to improve performance while minimally degrading ASI: the design on the left (vector length of 16, 32 KiB L1-D, 256 KiB LLC) achieves a speedup of 1.49× and an ASI of 3.55, whereas the design on the right (vector length of 32, 32 KiB L1-D, 64 KiB LLC) achieves a speedup of 1.73× while minimally degrading ASI to 3.43. Note that, when comparing the design point on the left versus the design point on the right, to compensate for the impact on sustainability by increasing vector size from 16 to 32, LLC has been reduced from 256 KiB to 64 KiB to achieve a minimal impact on ASI.

Takeaways. As Figures 5 to 9 show, there is always a trade-off between performance and sustainability. Increasing microarchitectural structure sizes improves performance but also reduces ASI. However, ASI can be used to focus on improving performance while keeping the reduction in sustainability to a minimum. Alternatively, it allows sacrificing minimal performance for a significantly higher improvement in sustainability. The key contribution of ASI is to provide designers with a way to identify the optimal point in the performance versus sustainability trade-off space, while holistically considering both fixed-work and fixed-time scenarios to mitigate the rebound effect. Furthermore, ASI allows for discarding design points that do not meet certain minimal requirements. For example, maintaining a minimal speedup against a reference design while maximizing ASI, or, vice versa, maintaining a certain ASI while maximizing speedup.

VII. Sensitivity Analyses

ASI depends on two factors: (1) the α parameter, which balances the embodied versus operational footprints, and (2) the baseline processor design used for comparison. We examine their impact on ASI through an SVR case study and discuss broader implications.

A. Embodied versus Operational Footprint

So far, we assumed $\alpha = 0.8$. A high α favors reducing chip area when optimizing ASI, typical for mobile and embedded devices. In contrast, always-connected devices usually have low α since operational emissions dominate [22], [23]. To explore this effect, we also consider $\alpha = 0.5$ (equal emphasis)

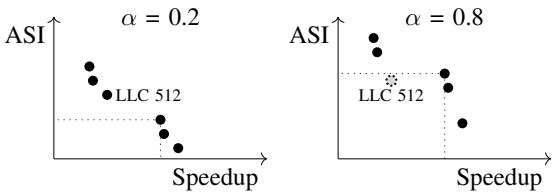

Fig. 10: Illustration of a point previously on the Pareto front, which is not anymore, when α changes value.

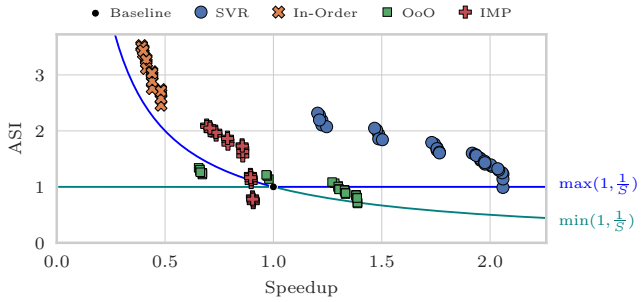

Fig. 11: The Pareto front for the different architectures assuming $\alpha = 0.5$.

and $\alpha = 0.2$ (operational-focused). Using the four architectures from Section VI, we generate Pareto fronts for these values (Figures 11 and 12) and compare them with $\alpha = 0.8$ (Figure 5).

We observe opposing trends: InO, IMP, SVR, and some OoO designs shift downward, while other OoO designs shift upward as α decreases. Specifically, InO, IMP, SVR, and certain OoO cores move to lower ASI values because they cannot reduce area or power enough to offset the reduced emphasis on chip area. This even pushes some high-performing IMP configurations from the strongly sustainable region to the unsustainable region. Conversely, other OoO cores become more sustainable: while some were unsustainable at $\alpha = 0.8$, none remain so at $\alpha = 0.2$. Intuitively, when chip area of a design exceeds the baseline's chip area, the embodied footprint penalizes ASI. Lowering α reduces this penalty, improving ASI. Mathematically, the derivative of ASI with respect to α (see Equation (9)) is negative if $A_n > 1$, meaning ASI decreases as α increases. Since we decrease α, ASI rises, confirming this intuition.

$$\frac{\partial ASI}{\partial \alpha} = \frac{1 - A_n}{P_n(1-\alpha)^2} \tag{9}$$

Figure 13 illustrates how the architectural parameters vary on the Pareto front for SVR as a function of α. The total bar length indicates the number of Pareto-optimal points: 45, 35, and 29 for $\alpha = 0.2$, 0.5, and 0.8, respectively. Colored segments represent parameter choices; e.g., for $\alpha = 0.2$, nearly 50%, 25%, and 25% of the design points feature a L1-D size of 32, 64, and 128 KiB. As seen in Figure 9, vector length

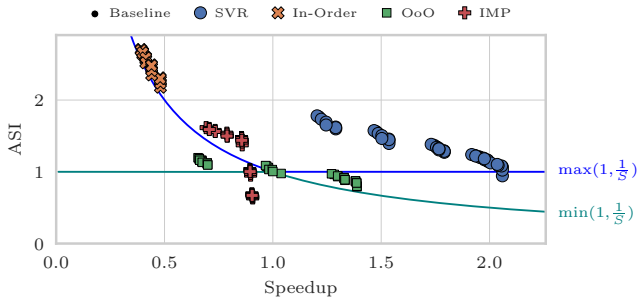

Fig. 12: The Pareto front for the different architectures assuming $\alpha = 0.2$.

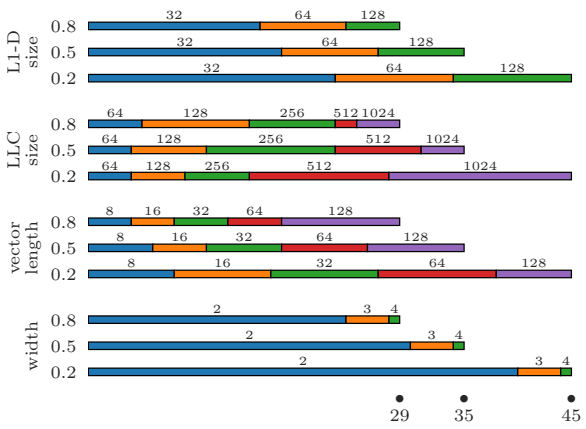

Fig. 13: The values for different architectural parameters on the Pareto front for SVR with changing α. The overall length of each bar shows the total number of points on the Pareto front for a given α.

dominates, followed by LLC size. Increasing α favors smaller LLC sizes, pushing larger ones off the Pareto front (Figure 10). This explains the drop in the number of Pareto points for higher α, as designs with large LLC, smaller vector lengths, and narrower pipelines turn out to be sub-optimal.

The key takeaway is that optimizing for a specific α value does not yield an optimal design for other α values. Therefore, we recommend evaluating a range of α values representative of the target device (e.g., 0.8 ± 0.1) to build confidence that the selected design remains on the Pareto front across that range.

B. Sensitivity to Baseline

So far, our analysis assumed the OoO baseline (Table II), making sustainability comparisons relative to that design. Since ASI measures sustainability against a reference, an architecture's status can change with the baseline. A larger-area baseline offers more scope to improve embodied footprint, while a higher-power baseline favors operational improvements. To illustrate this, we switch to an InO baseline and

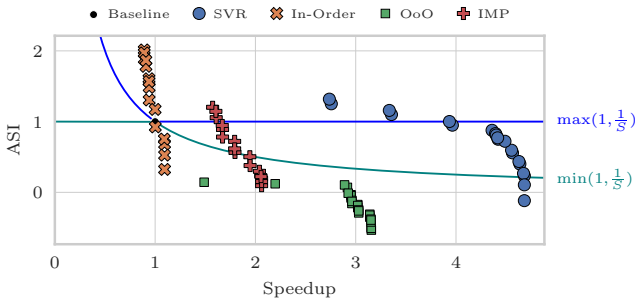

Fig. 14: Pareto fronts for the four architectures relative to the InO baseline rather than the OoO baseline reported in Figure 5.

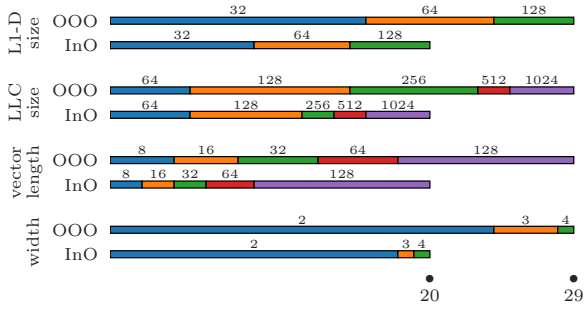

Fig. 15: The values for different architectural parameters on the Pareto front for SVR assuming different (InO vs OoO) baselines.

show Pareto fronts for the four architectures from Section VI in Figure 14 for $\alpha = 0.8$.

Compared to Figure 5 (which assumed an OoO baseline), all fronts shift downward to lower ASI. This is expected: the InO baseline incurs a smaller area, so other designs now appear less sustainable. All OoO points become unsustainable since they exceed both area and power of the InO baseline, making even weak sustainability infeasible. InO variants achieve similar performance to the baseline but can double ASI by reducing cache size. Likewise, some IMP points can move to strongly sustainable regions with minor performance trade-offs.

For SVR, all points were strongly sustainable relative to OoO (Figure 5), but with the InO baseline, Pareto points span three sustainability regions (Figure 14). The highest speedup (4.7×) features an LLC of 1024 KiB and vector length of 128, but these configurations are now unsustainable due to their higher area and power cost. Achieving strong sustainability requires reducing speedup to 3.4×. Figure 15 compares parameter choices across baselines: more points favor larger vector lengths and LLC sizes relative to InO. Thus, changing the baseline alters the optimal ASI-speedup trade-off.

VIII. Optimizing for ASI versus Other Metrics

The goal of ASI is to provide a unifying metric that can be optimized to improve the overall sustainability of an architecture. However, it is essential to assess how optimizing for ASI differs from other metrics. Therefore, in this section, we optimize for other metrics such as energy and energy-delay-area-product (EDAP) [20], [21], and demonstrate that optimizing for ASI is notably different than other metrics.

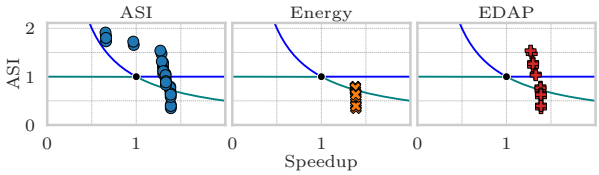

Fig. 16: The Pareto front optimized for ASI (left) vs energy (middle) vs EDAP (right). Optimizing for energy does not mean we are optimizing for sustainability, while EDAP misses out on the most sustainable solutions.

We calculate the energy versus speedup Pareto front using the algorithm in Section V-B (while replacing ASI with energy) relative to the OoO baseline, and then follow the same procedure for EDAP. For all the points on this energy-speedup Pareto front, we calculate their ASI assuming $\alpha = 0.8$, and place those points in the ASI versus speedup Pareto front shown in Figure 16. The design points for optimal energy and EDAP overlap with some of the design points for optimal ASI. Nonetheless, all the points for optimal energy are on the bottom right side of the graph, with most of them being unsustainable. While initially unexpected, it can be understood intuitively in hindsight, as these are the points in the unsustainable regions with the highest speedup. There are a couple design points in Figure 16 similar to the points in the 'Sustainable Under FW' region in Figure 2 where speedup increases more than power resulting in overall lowest energy requirement. Even if the value of α was set to 0.2, the energy-optimal points would still be located in the bottom right side of the Pareto front; this is obvious from the analysis in Section VII-A. Extending this analysis to the energy-delay-product (EDP) or energy-delay-square-product (ED^2P) metrics, which put even higher emphasis on performance, further exacerbates the problem and moves the optimal design points towards even lower ASI.

EDAP is more closely related to ASI, as it accounts for chip area, energy consumption and performance. It performs better than the energy metric in terms of sustainability, as it manages to capture design points in the strongly sustainable region as well, thanks to the inclusion of chip area in the overall EDAP metric. However, EDAP is agnostic to power consumption and there is hence a tendency towards high-performance cores, thereby missing out on some of the strongly sustainable design points. As a result, EDAP does not identify the most sustainable designs with the highest ASI at the top left in Figure 16 (leftmost graph), as well as some designs with large chip area that are part of the ASI Pareto-optimal set. Another issue related to EDAP (and energy) as a metric is that it is difficult to tell when a design would be prone to the rebound effect because it does not explicitly account for

power consumption.

From this analysis, we conclude that optimizing for ASI differs vastly from other metrics like energy and EDAP, necessitating the need for optimizing a metric such as ASI to better capture and assess the overall sustainability of processor microarchitecture designs.

IX. Related Work

Recent work has focused on modeling, estimating, and reducing the carbon footprint of ICT. This includes design space exploration and sustainability evaluation in datacenters [1], [13]; distributed carbon-aware scheduling of edge applications [37]; the impact of logic node technologies on power, area, performance, and greenhouse gas emissions [19]; estimating the embodied footprint of SSDs [40]; assessing the sustainability of ICs including the impact on dark silicon [10]; estimating the sustainability of chiplets using ECO-Chip [38]; trading off chiplet size versus count for GPUs to minimize environmental footprint [43]; investigating the embodied footprint of 3D digital compute-in-memory macros of 3D compute-in-memory architectures [11]. ASI is most closely related to two prior works, namely ACT and CCI, which we discuss below.

ACT [20], [21] is an analytical bottom-up architectural carbon model for estimating the total carbon footprint of a device while accounting for both the embodied and operational footprint. However, ACT's predictions are based on uncertain data. For example, it requires yield numbers from fab manufacturers, which are closely kept industry secrets, or it needs to estimate the material greenhouse gas emissions per unit area of a chip. The ACT paper introduces five optimization metrics for sustainability-aware design. In Section VIII, we evaluate and discuss the EDAP metric. The remaining four — carbon-delay product, carbon-energy product, carbon^2-energy-product, and carbon-energy^2-product — as well as the total-carbon-delay-product used in CORDOBA [17], all rely on carbon estimates derived using ACT. As such, they inherit the data uncertainty challenges discussed previously.

Another related sustainability metric is Computation Carbon Intensity (CCI) [39], which quantifies the total CO_2 emissions associated with a device, including embodied, operational, and network emissions, divided by the total number of operations it performs over its lifetime. CCI is particularly suitable for evaluating devices in data center contexts, where workloads can be distributed at scale. It facilitates comparisons between using new devices and reusing older ones — even unconventional candidates like smartphones — by assuming that the embodied emissions of reused devices have already been amortized. However, CCI does not account for performance differences between devices, and therefore overlooks quality-of-service (QoS) requirements. Where CCI guides long-term deployment and reuse decisions of hardware in a datacenter context, ASI enables sustainability-aware architectural exploration at the design-stage of a microarchitecture, serving complementary purposes.

X. Conclusion

The Architectural Sustainability Indicator (ASI) is a novel metric for assessing whether a microarchitectural design is strongly sustainable, weakly sustainable, or unsustainable. ASI provides intuitive understanding of the sustainability of a design relative to a reference design, and offers insight for transforming an unsustainable or weakly sustainable design into a strongly sustainable one. We illustrate ASI by resizing microarchitectural structures of a reference design, and show how the design can transition to different (sustainability) regions. The ASI-guided automatic design space exploration provides computer architects with a set of optimal points in the sustainability versus performance landscape. Using four representative processor microarchitectures, we perform such an automatic design space exploration, and highlight the high potential for improving sustainability while minimally degrading performance. We compare ASI against other metrics such as energy or energy-delay-area-product (EDAP), and illustrate the impact of embodied versus operational footprint on the overall sustainability profile of a design.

Acknowledgements

We thank the reviewers for their thoughtful feedback. This work is supported in part by the Research Foundation Flanders (FWO) grants No. G096225N and G031826N.

References

[1] B. Acun, B. Lee, F. Kazhamiaka, K. Maeng, U. Gupta, M. Chakkaravarthy, D. Brooks, and C.-J. Wu, "Carbon Explorer: A holistic framework for designing carbon aware datacenters," in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ser. ASPLOS 2023. New York, NY, USA: Association for Computing Machinery, 2023, p. 118–132. [Online]. Available: https://doi.org/10.1145/3575693.3575754

[2] S. Ainsworth and T. M. Jones, "Software prefetching for indirect memory accesses: A microarchitectural perspective," *ACM Transactions on Computer Systems*, vol. 36, no. 3, jun 2019. [Online]. Available: https://doi.org/10.1145/3319393

[3] B. Alcott, "Jevons' paradox," *Ecological Economics*, vol. 54, no. 1, pp. 9–21, 2005. [Online]. Available: https://doi.org/10.1016/j.ecolecon.2005.03.020

[4] J. A. Ang, B. W. Barrett, K. B. Wheeler, and R. C. Murphy, "Introducing the graph 500." *Cray User's Group (CUG)*, vol. 19, pp. 45–74, 5 2010. [Online]. Available: https://www.osti.gov/biblio/1014641

[5] R. U. Ayres, "Life cycle analysis: A critique," *Resources, Conservation and Recycling*, vol. 14, no. 3, pp. 199–223, 1995, life Cycle Management. [Online]. Available: https://doi.org/10.1016/0921-3449(95)00017-D

[6] D. H. Bailey, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, L. Dagum, R. A. Fatoohi, P. O. Frederickson, T. A. Lasinski, R. S. Schreiber, H. D. Simon, V. Venkatakrishnan, and S. K. Weeratunga, "The NAS parallel benchmarks—summary and preliminary results," in *Proceedings of the 1991 ACM/IEEE Conference on Supercomputing*, ser. Supercomputing '91. New York, NY, USA: Association for Computing Machinery, 1991, p. 158–165. [Online]. Available: https://doi.org/10.1145/125826.125925

[7] R. Basu Roy, R. Kanakagiri, Y. Jiang, and D. Tiwari, "ForgetMeNot: Understanding and modeling the impact of forever chemicals toward sustainable large-scale computing," in *Abstracts of the 2025 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '25. New York, NY, USA: Association for Computing Machinery, 2025, p. 183–185. [Online]. Available: https://doi.org/10.1145/3726854.3727288

[8] S. Beamer, K. Asanovic, and D. A. Patterson, "The GAP benchmark suite," *CoRR*, vol. abs/1508.03619, 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1508.03619

[9] S. Blanas, Y. Li, and J. M. Patel, "Design and evaluation of main memory hash join algorithms for multi-core CPUs," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 37–48. [Online]. Available: https://doi.org/10.1145/1989323.1989328

[10] E. Brunvand, D. Kline, and A. K. Jones, "Dark silicon considered harmful: A case for truly green computing," in *2018 Ninth International Green and Sustainable Computing Conference (IGSC)*, Oct 2018, pp. 1–8. [Online]. Available: https://doi.org/10.1109/IGCC.2018.8752110

[11] H. J. Byun, U. Gupta, and J.-S. Seo, "Energy-/carbon-aware evaluation and optimization of 3-D IC architecture with digital compute-in-memory designs," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 10, pp. 98–106, 2024. [Online]. Available: https://doi.org/10.1109/JXCDC.2024.3479100

[12] T. E. Carlson, W. Heirman, S. Eyerman, I. Hur, and L. Eeckhout, "An evaluation of high-level mechanistic core models," *ACM Trans. Archit. Code Optim.*, vol. 11, no. 3, aug 2014. [Online]. Available: https://doi.org/10.1145/2629677

[13] J. Chang, J. Meza, P. Ranganathan, A. Shah, R. Shih, and C. Bash, "Totally green: evaluating and designing servers for lifecycle environmental impact," *SIGPLAN Not.*, vol. 47, no. 4, p. 25–36, Mar. 2012. [Online]. Available: https://doi.org/10.1145/2248487.2150980

[14] L. Eeckhout, "A first-order model to assess computer architecture sustainability," *IEEE Computer Architecture Letters*, vol. 21, no. 2, pp. 137–140, July 2022. [Online]. Available: https://doi.org/10.1109/LCA.2022.3217366

[15] ——, "FOCAL: A first-order carbon model to assess processor sustainability," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ser. ASPLOS '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 401–415. [Online]. Available: https://doi.org/10.1145/3620665.3640415

[16] ——, "The sustainability gap for computing: Quo vadis?" *Commun. ACM*, vol. 68, no. 3, p. 70–79, Feb. 2025. [Online]. Available: https://doi.org/10.1145/3699595

[17] M. Elgamal, D. Carmean, E. Ansari, O. Zed, R. Peri, S. Manne, U. Gupta, G.-Y. Wei, D. Brooks, G. Hills, and C.-J. Wu, "CORDOBA: Carbon-efficient optimization framework for computing systems," in *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, March 2025, pp. 1289–1303. [Online]. Available: https://doi.org/10.1109/HPCA61900.2025.00098

[18] C. Freitag, M. Berners-Lee, K. Widdicks, B. Knowles, G. S. Blair, and A. Friday, "The real climate and transformative impact of ICT: A critique of estimates, trends, and regulations," *Patterns*, 2021. [Online]. Available: https://doi.org/10.1016/j.patter.2021.100340

[19] M. Garcia Bardon, P. Wuytens, L.-Å. Ragnarsson, G. Mirabelli, D. Jang, G. Willems, A. Mallik, A. Spessot, J. Ryckaert, and B. Parvais, "DTCO including sustainability: Power-performance-area-cost-environmental score (PPACE) analysis for logic technologies," in *2020 IEEE International Electron Devices Meeting (IEDM)*, Dec 2020, pp. 41.4.1–41.4.4. [Online]. Available: https://doi.org/10.1109/IEDM13553.2020.9372004

[20] U. Gupta, M. Elgamal, G. Hills, G.-Y. Wei, H.-H. S. Lee, D. Brooks, and C.-J. Wu, "ACT: Designing sustainable computer systems with an architectural carbon modeling tool," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*, ser. ISCA '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 784–799. [Online]. Available: https://doi.org/10.1145/3470496.3527408

[21] ——, "Architectural CO2 footprint tool: Designing sustainable computer systems with an architectural carbon modeling tool," *IEEE Micro*, vol. 43, no. 4, pp. 107–117, 2023. [Online]. Available: https://doi.org/10.1109/MM.2023.3275139

[22] U. Gupta, Y. G. Kim, S. Lee, J. Tse, H.-H. S. Lee, G.-Y. Wei, D. Brooks, and C.-J. Wu, "Chasing carbon: The elusive environmental footprint of computing," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, Feb 2021, pp. 854–867. [Online]. Available: https://doi.org/10.1109/HPCA51647.2021.00076

[23] ——, "Chasing carbon: The elusive environmental footprint of computing," *IEEE Micro*, vol. 42, no. 4, pp. 37–47, July 2022. [Online]. Available: https://doi.org/10.1109/MM.2022.3163226

[24] Y. Jiang, R. B. Roy, R. Kanakagiri, and D. Tiwari, "WaterWise: Co-optimizing carbon- and water-footprint toward environmentally sustainable cloud computing," in *Proceedings of the 30th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP '25. New York, NY, USA: Association for Computing Machinery, 2025, p. 297–311. [Online]. Available: https://doi.org/10.1145/3710848.3710891

[25] D. Kline, N. Parshook, X. Ge, E. Brunvand, R. Melhem, P. K. Chrysanthis, and A. K. Jones, "GreenChip: A tool for evaluating holistic sustainability of modern computing systems," *Sustainable Computing: Informatics and Systems*, vol. 22, pp. 322–332, 2019. [Online]. Available: https://doi.org/10.1016/j.suscom.2017.10.001

[26] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO 42. New York, NY, USA: Association for Computing Machinery, 2009, p. 469–480. [Online]. Available: https://doi.org/10.1145/1669112.1669172

[27] P. R. Luszczek, D. H. Bailey, J. J. Dongarra, J. Kepner, R. F. Lucas, R. Rabenseifner, and D. Takahashi, "The HPC challenge (HPCC) benchmark suite," in *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, ser. SC '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 213–es. [Online]. Available: https://doi.org/10.1145/1188455.1188677

[28] A. Naithani, S. Ainsworth, T. M. Jones, and L. Eeckhout, " Vector Runahead ," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*. Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2021, pp. 195–208. [Online]. Available: https://doi.org/10.1109/ISCA52012.2021.00024

[29] ——, " Vector Runahead for Indirect Memory Accesses ," *IEEE Micro*, vol. 42, no. 04, pp. 116–123, Jul. 2022. [Online]. Available: https://doi.org/10.1109/MM.2022.3163132

[30] A. Naithani, J. Roelandts, S. Ainsworth, T. M. Jones, and L. Eeckhout, "Decoupled vector runahead," in *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 17–31. [Online]. Available: https://doi.org/10.1145/3613424.3614255

[31] ——, " Decoupled Vector Runahead for Prefetching Nested Memory-Access Chains ," *IEEE Micro*, vol. 44, no. 04, pp. 20–26, Jul. 2024. [Online]. Available: https://doi.org/10.1109/MM.2024.3406891

[32] L.-A. Ragnarsson, M. G. Bardon, P. Wuytens, G. Mirabelli, D. Jang, G. Willems, A. Mallik, A. Spessot, J. Ryckaert, and B. Parvais, "Environmental impact of CMOS logic technologies," in *2022 6th IEEE Electron Devices Technology & Manufacturing Conference (EDTM)*, March 2022, pp. 82–84. [Online]. Available: https://doi.org/10.1109/EDTM53872.2022.9798208

[33] J. Roelandts, A. Naithani, S. Ainsworth, T. M. Jones, and L. Eeckhout, " Scalar Vector Runahead ," in *2024 57th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. Los Alamitos, CA, USA: IEEE Computer Society, Nov. 2024, pp. 1367–1381. [Online]. Available: https://doi.org/10.1109/MICRO61859.2024.00101

[34] ——, "Scalar vector runahead: Removing the shackles of indirect memory chains on in-order cores," *IEEE Micro*, pp. 1–7, 2025. [Online]. Available: https://doi.org/10.1109/MM.2025.3577524

[35] J. Roelandts, A. Naithani, and L. Eeckhout, "The architectural sustainability indicator," *IEEE Computer Architecture Letters*, vol. 24, no. 2, pp. 205–208, July 2025. [Online]. Available: https://doi.org/10.1109/LCA.2025.3576891

[36] M. Själander, M. Martonosi, and S. Kaxiras, *Power-efficient computer architectures: Recent advances*, ser. Synthesis Lectures on Computer Architecture. Springer International Publishing, 2022. [Online]. Available: https://doi.org/10.1007/978-3-031-01745-2

[37] Y. Son, U. Gupta, A. McCrabb, Y. G. Kim, V. Bertacco, D. Brooks, and C.-J. Wu, "GreenScale: Carbon optimization for edge computing," *IEEE Internet of Things Journal*, pp. 1–1, 2025. [Online]. Available: https://doi.org/10.1109/JIOT.2025.3555153

[38] C. C. Sudarshan, N. Matkar, S. Vrudhula, S. S. Sapatnekar, and V. A. Chhabria, "ECO-CHIP: Estimation of carbon footprint of chiplet-based architectures for sustainable VLSI," in *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, March 2024, pp. 671–685. [Online]. Available: https://doi.org/10.1109/HPCA57654.2024.00058

[39] J. Switzer, G. Marcano, R. Kastner, and P. Pannuto, "Junkyard computing: Repurposing discarded smartphones to minimize carbon," in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ser. ASPLOS 2023. New York, NY, USA: Association for Computing Machinery, 2023, p. 400–412. [Online]. Available: https://doi.org/10.1145/3575693.3575710

[40] S. Tannu and P. J. Nair, "The dirty secret of SSDs: Embodied carbon," *SIGENERGY Energy Inform. Rev.*, vol. 3, no. 3, p. 4–9, Oct. 2023. [Online]. Available: https://doi.org/10.1145/3630614.3630616

[41] WikiChip, "Cortex-A510 - Microarchitectures - ARM." [Online]. Available: https://en.wikichip.org/wiki/arm_holdings/microarchitectures/cortex-a510

[42] X. Yu, C. J. Hughes, N. Satish, and S. Devadas, "IMP: Indirect memory prefetcher," in *Proceedings of the 48th International Symposium on Microarchitecture*, ser. MICRO-48. New York, NY, USA: Association for Computing Machinery, 2015, p. 178–190. [Online]. Available: https://doi.org/10.1145/2830772.2830807

[43] S. Zhang, M. Naderan-Tahan, M. Jahre, and L. Eeckhout, "Balancing performance against cost and sustainability in multi-chip-module GPUs," *IEEE Computer Architecture Letters*, vol. 22, no. 2, pp. 145–148, July 2023. [Online]. Available: https://doi.org/10.1109/LCA.2023.3313203

[44] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," ETH Zurich, Tech. Rep., 2001. [Online]. Available: https://doi.org/10.3929/ethz-a-004284029