Measuring Flexibility in Single-ISA Heterogeneous Processors

Erik Tomusk Christophe Dubach Michael O'Boyle
University of Edinburgh, UK
{E.Tomusk, christophe.dubach, mob} @ed.ac.uk

ABSTRACT

Single-ISA heterogeneous processors are a promising method for enabling runtime power flexibility. Low-priority programs run on low-power cores, and high-priority programs run on high-power cores. In recent years, a number of methods for heterogeneous design space exploration have emerged. These methods search the design space for Pareto frontiers of cores that are optimal for power and speed. We demonstrate that a heterogeneous processor cannot be composed by simply selecting some cores from a Pareto-optimal set; the selection must give even coverage of the design space. We then define a metric—*clumpiness*—for measuring how well selected heterogeneous cores cover the design space.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Performance attributes, Design studies

Keywords

Clumpiness; heterogeneous design space exploration; single-ISA; Pareto-optimal

1. INTRODUCTION

Single-ISA heterogeneous processors are already available in the consumer space [4], and their design is the topic of ongoing research. A common method of design space exploration is to find a set of cores that is Pareto-optimal for power and speed [5, 8]. These methods do not offer guidance on which cores should be selected; they expect the designer to select cores from the Pareto-optimal set using external criteria. In section 2, we demonstrate that simply selecting some cores from a Pareto-optimal set will not guarantee that the selection makes effective use of heterogeneity. Then in section 3, we propose a metric for *clumpiness* to quantify the intuitive notion that a good selection of heterogeneous cores will provide uniform coverage of the design space.

2. MOTIVATING EXAMPLE

Given two different selections of heterogeneous cores, it should be possible to determine which is better. Consider figure 1. Two possible ways of selecting four Pareto-optimal cores are shown. Both selection $\mathcal{S}_1$ (top) and selection $\mathcal{S}_2$ (bottom) contain cores *C1*, *C3*, and *C4*. $\mathcal{S}_1$ also contains *C2a*, while $\mathcal{S}_2$ contains *C2b*. All five cores are Pareto-optimal for power and execution time, but $\mathcal{S}_2$ is intuitively better for two reasons: First, $\mathcal{S}_2$ provides the end user with ***graceful performance degradation*** (flexibility) in a power-constrained environment. If there is insufficient power to run on, e.g., *C3*, execution will be on *C2b* with a moderate speed reduction. In contrast, the gap between *C1* and *C2a* in $\mathcal{S}_1$ is so large that if execution cannot take place on *C2a*, the user will experience a significant slowdown when execution moves to *C1*. Second, $\mathcal{S}_1$ also has the opposite problem: the gap between *C2a* and *C3* is so small that the two cores are homogeneous for all practical purposes. There is no justification for the ***engineering effort*** required to design both, despite the fact that both are Pareto-optimal. In the following paragraphs, we first note the source of the data in figure 1. We then show that quantifying the intuitive difference between the two selections is non-trivial.

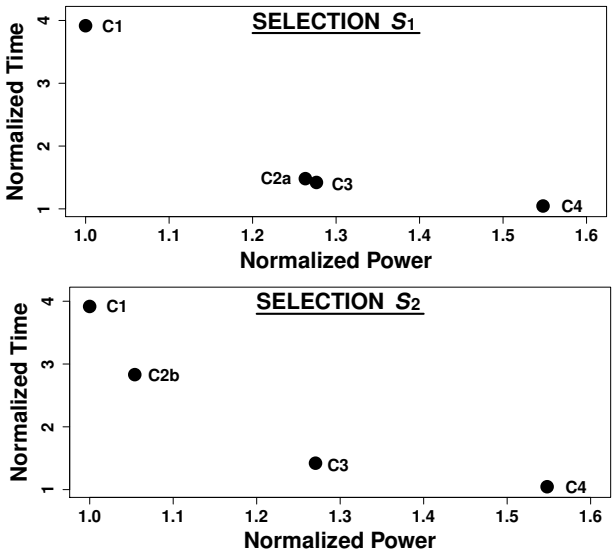

Figure 1: Two possibilities for selecting four Pareto-optimal cores for *AES*. The axes are normalized—1 on the time axis is fastest, 4 is four times slower, etc. Selection $\mathcal{S}_1$ shows two cores clumped together.

Data source We use gem5 [1] and McPAT [6] to simulate the *AES* encryption benchmark from EEMBC DENBench (digital entertainment benchmark) on 3000 out-of-order cores sampled from a large design space. The design space includes 19 microarchitectural parameters controlling L1

PACT'14, August 24–27, 2014, Edmonton, AB, Canada.
ACM 978-1-4503-2809-8/14/08.
http://dx.doi.org/10.1145/2628071.2628125.

cache, queues, the branch predictor, etc. We find the Pareto frontier of cores that are optimal for power and execution time. From the 76 cores on the frontier, we make two selections of four cores each (figure 1).

Average waste metric As noted above, time is wasted when running on a lower-power core. We can quantify average wasted time, TW, with equation 1. N is the number of intervals between cores, ΔP_r is the difference between the highest- and lowest-power core in the set, ΔP_i is the difference between the highest- and lowest-power core in interval i, $\max(T_i)$ is execution time on the slowest core in interval i, and $\overline{T_i}$ is an estimate for average time in interval i. $\overline{T_i}$ is derived from all cores in the Pareto-optimal set.

$$TW = \sum_{i=1}^{N} \frac{\Delta P_i}{\Delta P_r} \left(\frac{\max(T_i) - \overline{T_i}}{\overline{T_i}} \right) \quad (1)$$

Even though differences between selection $\mathcal{S}_1$ and $\mathcal{S}_2$ are intuitively significant, their average wasted time differs by only 5% (table 1). This is because the large gap between *C1* and *C2a* is averaged away by the minuscule gap between *C2a* and *C3*—the two problems with $\mathcal{S}_1$ cancel each other out. We can conclude that a metric based on an average fails to capture worst-case behavior and cannot effectively differentiate between a good and bad selection of cores.

Worst-case waste metric An alternative approach would be to measure worst-case behavior—the selection that minimizes the maximum gap between cores is considered best. This approach introduces a masking problem: two selections could have very similar worst-case gaps between cores, with the rest of the cores distributed completely differently (not pictured). Any metric that uses a local feature of a selection to evaluate the entire selection will inevitably make erroneous comparisons.

3. A METRIC FOR CLUMPINESS

We propose the *clumpiness* metric for differentiating between selections of heterogeneous cores. Clumpiness measures the same intuitive concept as entropy-based diversity [3], but while diversity is difficult to normalize and depends on a user-selected density function, clumpiness is normalized and does not require tuning. Clumpiness is related to ϵ-coverage and δ-uniformity [7], and Δ-nonuniformity [2].

Clumpiness is represented with ↄ (*kaph*) and is defined in equation 2 for a 1-dimensional distribution. ↄ is calculated over the range $[r_1, r_2]$ for N number of ordered points x. The first term in the numerator of equation 2 measures the distance from the beginning of the range to the first point. The second term measures the distance from the last point to the end of the range. The third term measures the distance from each intermediate point to the halfway mark between its two neighboring points. ↄ ranges from 0 (even distribution) to 1 (one tight cluster of points).

$$ↄ = \frac{(x_1 - r_1) + (r_2 - x_N) + \sum_{i=2}^{N-1} d_i}{r_2 - r_1} \quad (2)$$

$$d_i = \left| x_i - \frac{x_{i-1} + x_{i+1}}{2} \right|$$

Clumpiness takes all points into account, but measures point density locally. Local features cannot be averaged away, but neither can they dominate the metric. Equation 2 can be readily extended to more dimensions, but since we are considering points that lie on a curve, a 1-dimensional version of ↄ is sufficient.

	Selection $\mathcal{S}_1$	**Selection $\mathcal{S}_2$**
Avg. Waste (TW)	24%	19%
Clumpiness (ↄ)	47%	22%

Table 1: Clumpiness more accurately quantifies the intuition that selection $\mathcal{S}_2$ is significantly better

To measure the clumpiness of selections $\mathcal{S}_1$ and $\mathcal{S}_2$, we flatten the points in 2-dimensional, normalized power-time space to one dimension using the Euclidean distance between points. E.g., for selection $\mathcal{S}_1$, *C1* is at the origin. *C2a* is at 2.45, as this is the Euclidean distance to *C1*. *C3* is at 2.51, since the Euclidean distance to *C2a* is only 0.06. r_1 and r_2 are set to the coordinates of *C1* and *C4*, respectively.

The ↄ-values for the two flattened selections are in table 1. Clumpiness expresses what is intuitively obvious—that selection $\mathcal{S}_1$ is much more clustered (more than 2×), and that selection $\mathcal{S}_2$ gives more even coverage of the space.

4. CONCLUSION

Considerable effort has been invested into methods of design spaces exploration for power- and speed-optimal cores. The problem of selecting cores from Pareto-optimal sets has seen much less attention. We have defined clumpiness (ↄ) to quantify the intuition that some selections of cores are better than others. Selections with a high ↄ-value waste engineering effort on nearly identical cores while also leaving large performance gaps between cores. A small ↄ, on the other hand, indicates that a set of cores maximizes runtime flexibility. Unlike related metrics, ↄ evaluates both the spread and uniformity of points without the need to be tuned to the specific problem. Clumpiness provides a simple way for evaluating future work on heterogeneous core selection.

5. REFERENCES

[1] N. Binkert et al. The gem5 simulator. *SIGARCH Computer Architure News*, 39(2), Aug 2011.

[2] K. Deb et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. on Evolutionary Computation*, 6(2), Apr 2002.

[3] A. Farhang-Mehr and S. Azarm. Diversity assessment of Pareto optimal solution sets: An entropy approach. In *CEC*, May 2002.

[4] P. Greenhalgh. Big.little processing with ARM Cortex-A15 & Cortex-A7. White paper, ARM Ltd., 2011.

[5] B. C. Lee and D. M. Brooks. Illustrative design space studies with microarchitectural regression models. In *HPCA*, 2007.

[6] S. Li et al. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, 2009.

[7] S. Sayın. Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. *Mathematical Programming*, 87(3), 2000.

[8] Y. Turakhia et al. HaDeS: Architectural synthesis for heterogeneous dark silicon chip multi-processors. In *DAC*, 2013.