ArchExplorer: Microarchitecture Exploration Via Bottleneck Analysis

Chen Bai[1,4] Jiayi Huang[2] Xuechao Wei[4] Yuzhe Ma[2] Sicheng Li[4] Hongzhong Zheng[4] Bei Yu[1] Yuan Xie[3,4]

[1]The Chinese University of Hong Kong
[2]The Hong Kong University of Science and Technology (Guangzhou)
[3]Hong Kong University of Science and Technology
[4]DAMO Academy, Alibaba Group

Introduction

Problem formulation:

Microprocessor Microarchitecture Design Space Exploration (DSE)

Given benchmark suites and microprocessor microarchitecture design space, find optimal microarchitecture parameters that can achieve good trade-offs between performance, power, and area (PPA).

Previous Methodologies & Limitations

- Industry:
  - Expertise of computer architects. → Architects' bias.
- Academia:
  - Analytical methodologies: based on mechanistic models with intepretable equations. → Require immense domain knowledge.
  - Black-box methodologies: based on machine-learning techniques. → Require high computing resources.

Goal & Approach:

- Goal: solve the problem by removing limitations of previous methodologies: remove massive domain knowledge requirement & mitigate the high computing demands.
- Approach: DSE via automated bottleneck analysis.

Rationales:

- Perfect machine: unlimited hardware resources.
  - Performance is constrained only by program's true data dependencies.
- Real machine: limited hardware resources.
  - Performance is constrained by program's true data dependencies and resource constraints.
  - Two distinct types of resources: deficient and exhausted & abundant and idle.

Balanced Microarchitecture

A balanced microarchitecture can simultaneously maximize the utilization of each hardware resource. We refer to a bottleneck as insufficient hardware resource that is exhausted by instructions and results in high program runtime.

Findings:

We find that the relations between resource constraints and machine parallelism are similar to the cask effect.

How to identify the type of resources?

- The utilization status of each resource in the microexecution should be captured.
- Whether the overlapping events matter for the execution time should be considered. → Call for a global view of the entire microexecution, which the critical path analysis can help.

Background & Motivation

Challenges in Microarchitecture DSE:

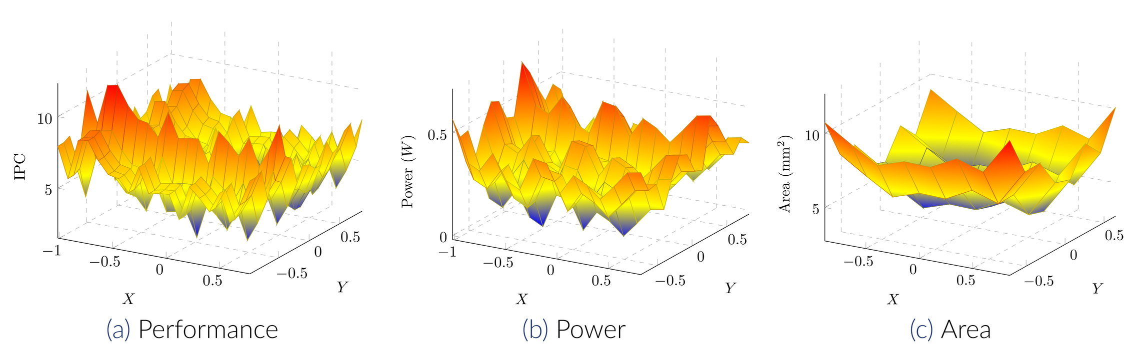

Figure 1. A visualization of the design space for 458.sjeng. Each microarchitecture is reduced to two-dimension through t-SNE to facilitate the visualization of PPA distributions.

- Complicated design space.
- High simulation runtime.

Bottleneck Analysis Matters in DSE:

Removing microarchitecture bottlenecks can significantly enhance the PPA trade-off.

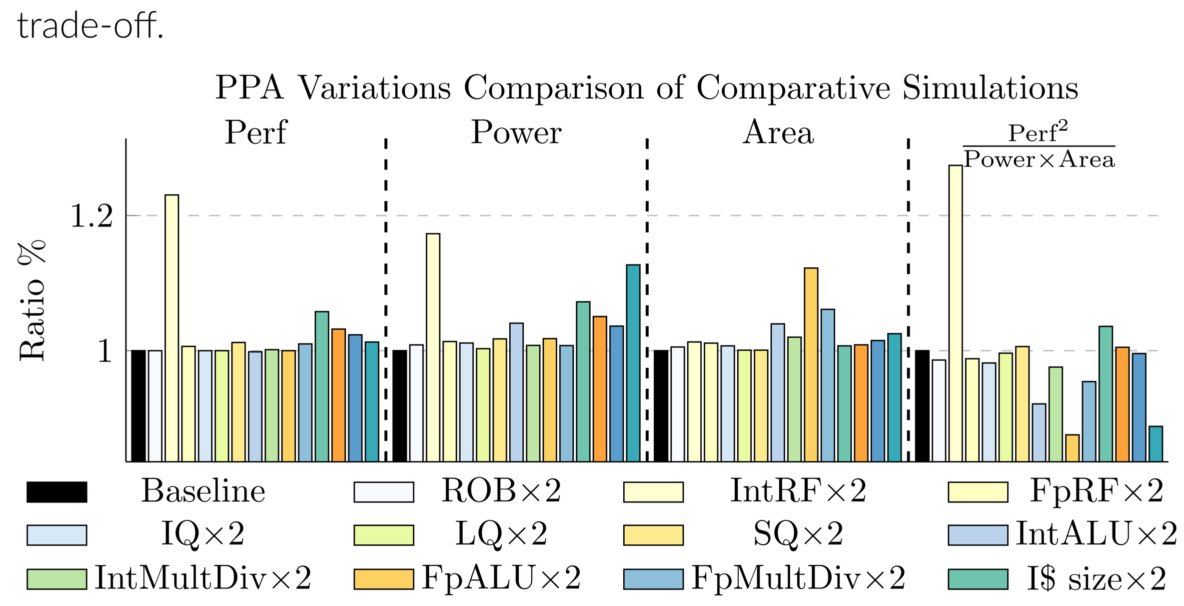

Figure 2. Each bar represents the microarchitecture's metric in %. The bar, *e.g.*, "ROB ×2", indicates the microarchitecture is the same as the baseline except that it doubles ROB. $\text{Perf}^2/(\text{Power} \times \text{Area})$ denotes the PPA trade-off.

A straightforward heuristic: in the DSE, assigning necessary hardware resources and reducing redundant ones.

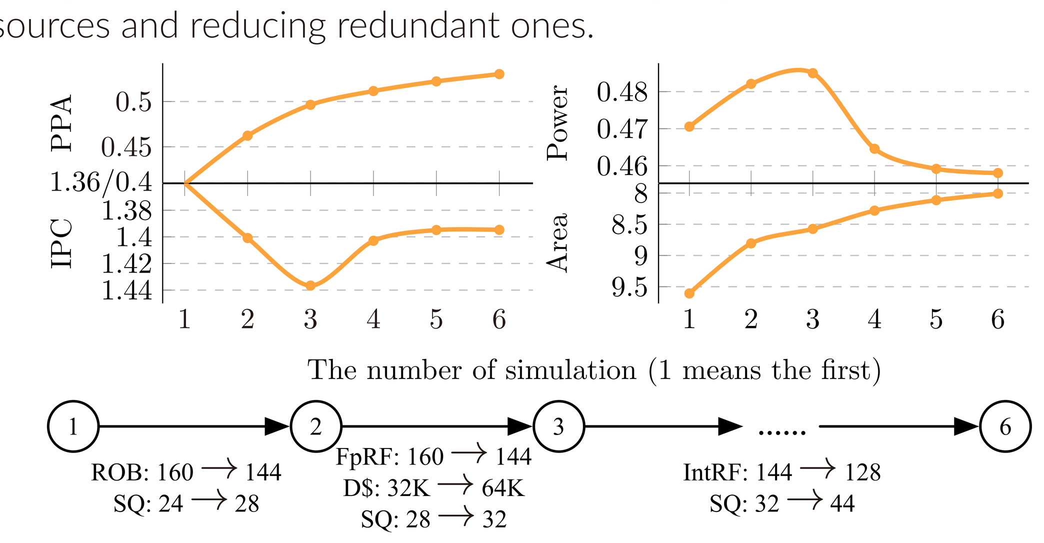

Figure 3. Search following series of small changes stepwise. PPA denotes $\text{Perf}^2/(\text{Power} \times \text{Area})$.

Critical Path Analysis:

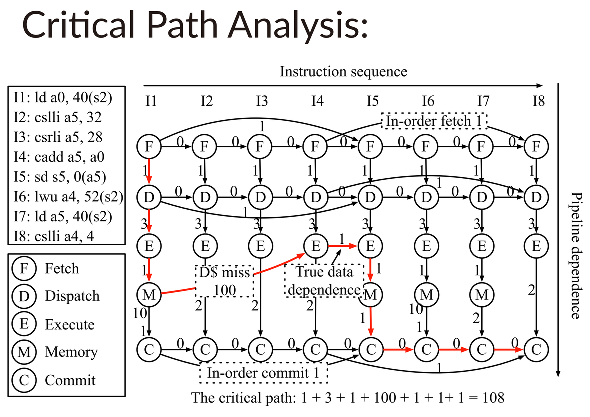

Figure 4. An overview of the dynamic event-dependence graph (DEG).

The former dynamic event dependence graph is inaccurate:

- The dependence and weights assignment are static without adhering to actual microexecution.
- The critical path cannot accurately characterize the bottlenecks' contributions to the overall runtime.

Lessons Learned & Design Principles

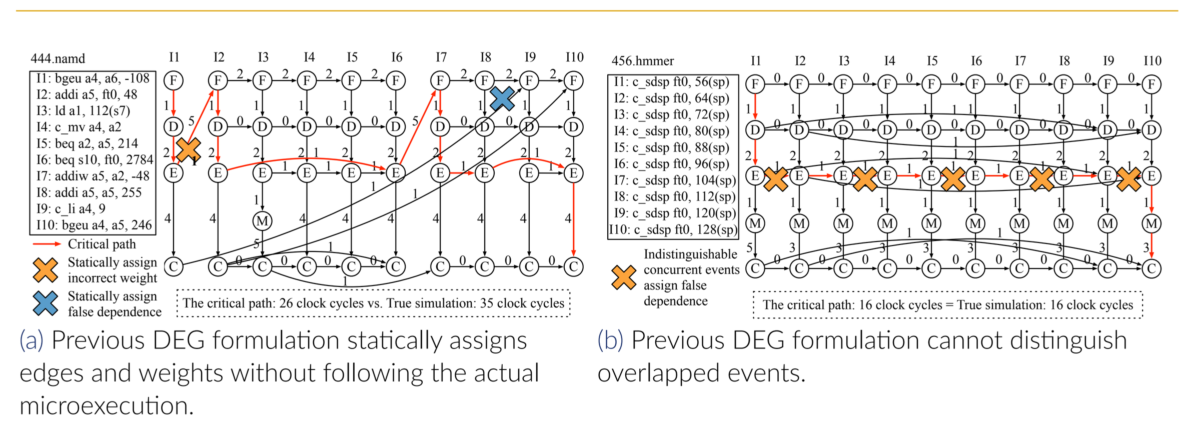

Figure 5. (a) and (b) uses Calipers to demonstrate three kinds of error sources.

Design Principles

- The dependencies contributing to execution time should be captured as much as possible. → Capturing more resource usages improves the utilization approximation.
- Concurrent events should be distinguishable. → The distinguishability unveils whether we matter a concurrent event for bottleneck contributions to the overall execution time.

The ArchExplorer Approach

Overview:

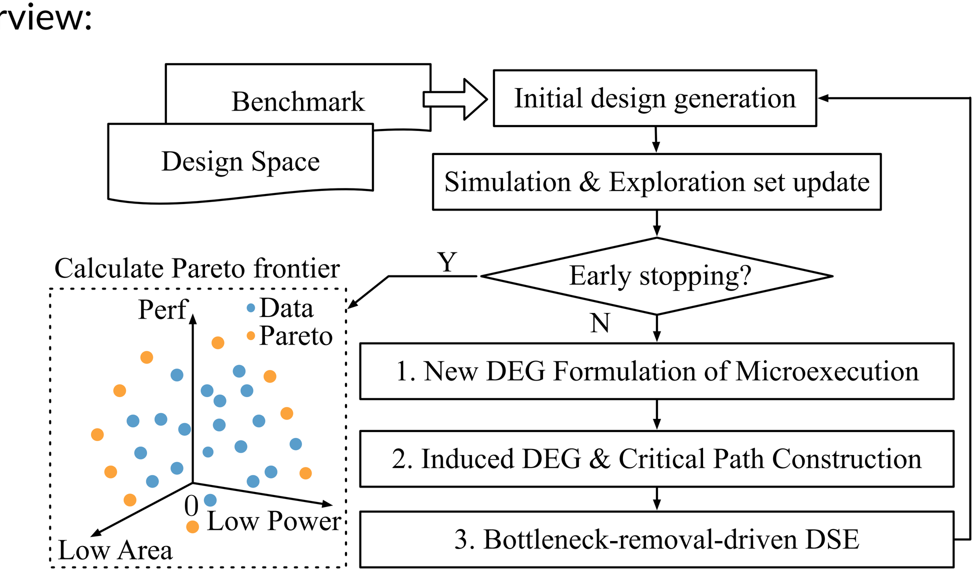

Figure 6. An overview of the ArchExplorer approach.

New DEG Formulation of Microexecution:

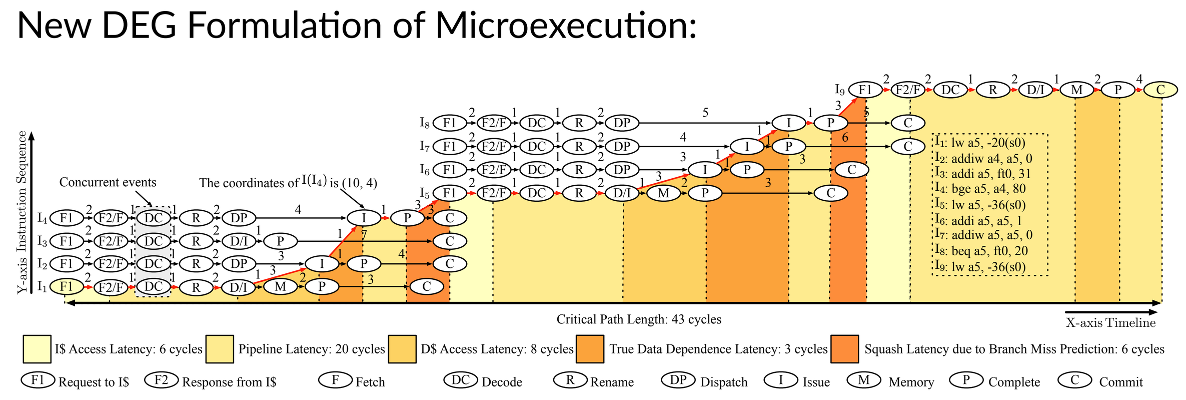

Figure 7. An overview of the new DEG formulation of microexecution. The critical path is highlighted in red.

Highlights of new DEG formulation:

- Nodes represent pipeline stages, and edges represent dependencies.
- Align instructions *w.r.t.* the time instead of pipeline stages.
- Dynamic DEG construction.
- Ascertain the overlapped events.

Induced DEG & Critical Path Construction:

Two "skewed" edges are annotated with $s_i \rightarrow e_j$ and $s_k \rightarrow e_l$:

- Rule 1 (Connect via time): s_i is connected to s_k if the time of s_k is the closest to s_i.
- Rule 2 (Connect via instruction sequence): s_i is connected to s_k if the instruction sequence k is the closest to i.

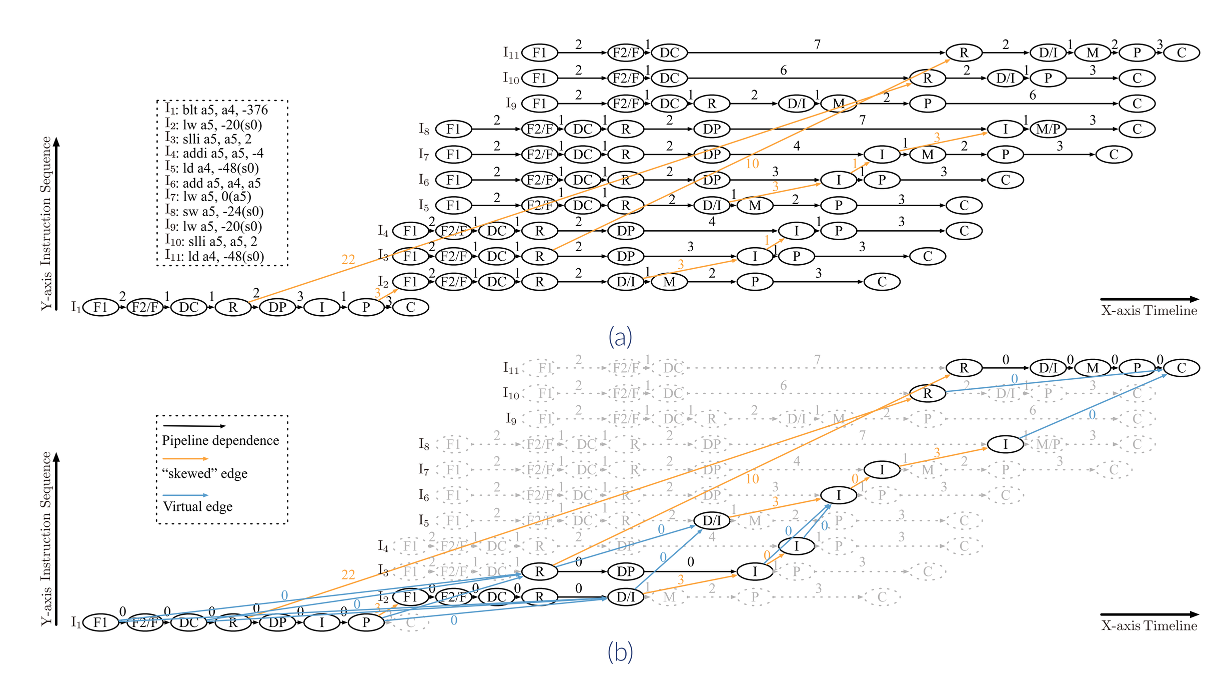

Figure 8. (a) An example code snippet and its corresponding new DEG formulation. (b) The overview of induced DEG with edge cost extracted from DEG.

Bottleneck-removal-driven DSE:

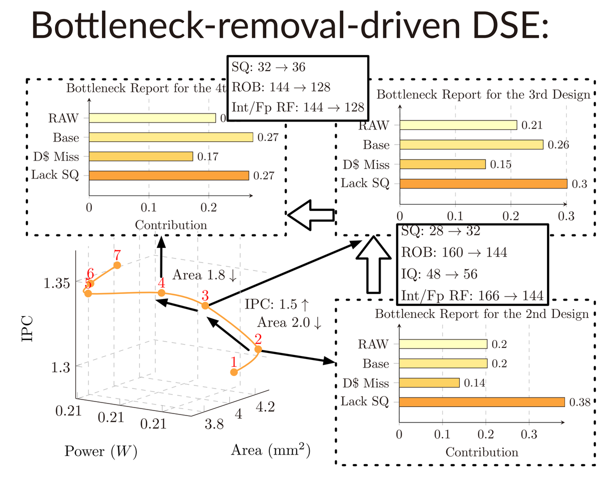

Figure 9. An overview of the dynamic event-dependence graph.

Resource reassignment:

- We select the next larger candidate value from the specification if we need to increase it.
- We decrease them to the next smaller candidate value if they do not have a contribution.

Results

Due to the limited poster space, we only showcase the main results. For experiment setup and detailed results, please refer to our paper.

Comparison w. DSE Methodologies:

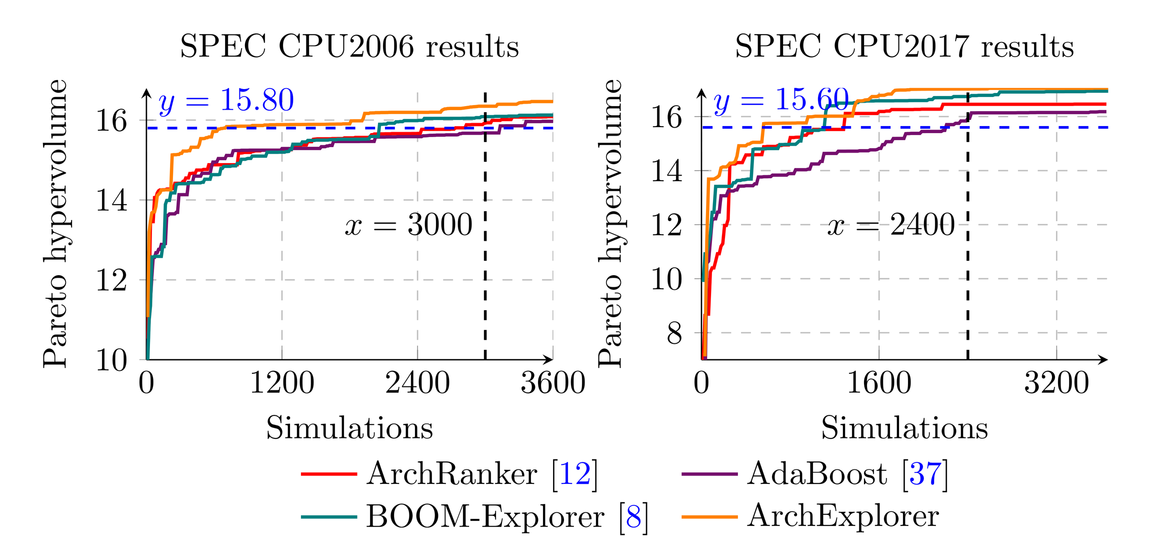

Figure 10. The visualization of Pareto hypervolume curves in terms of the number of simulations.

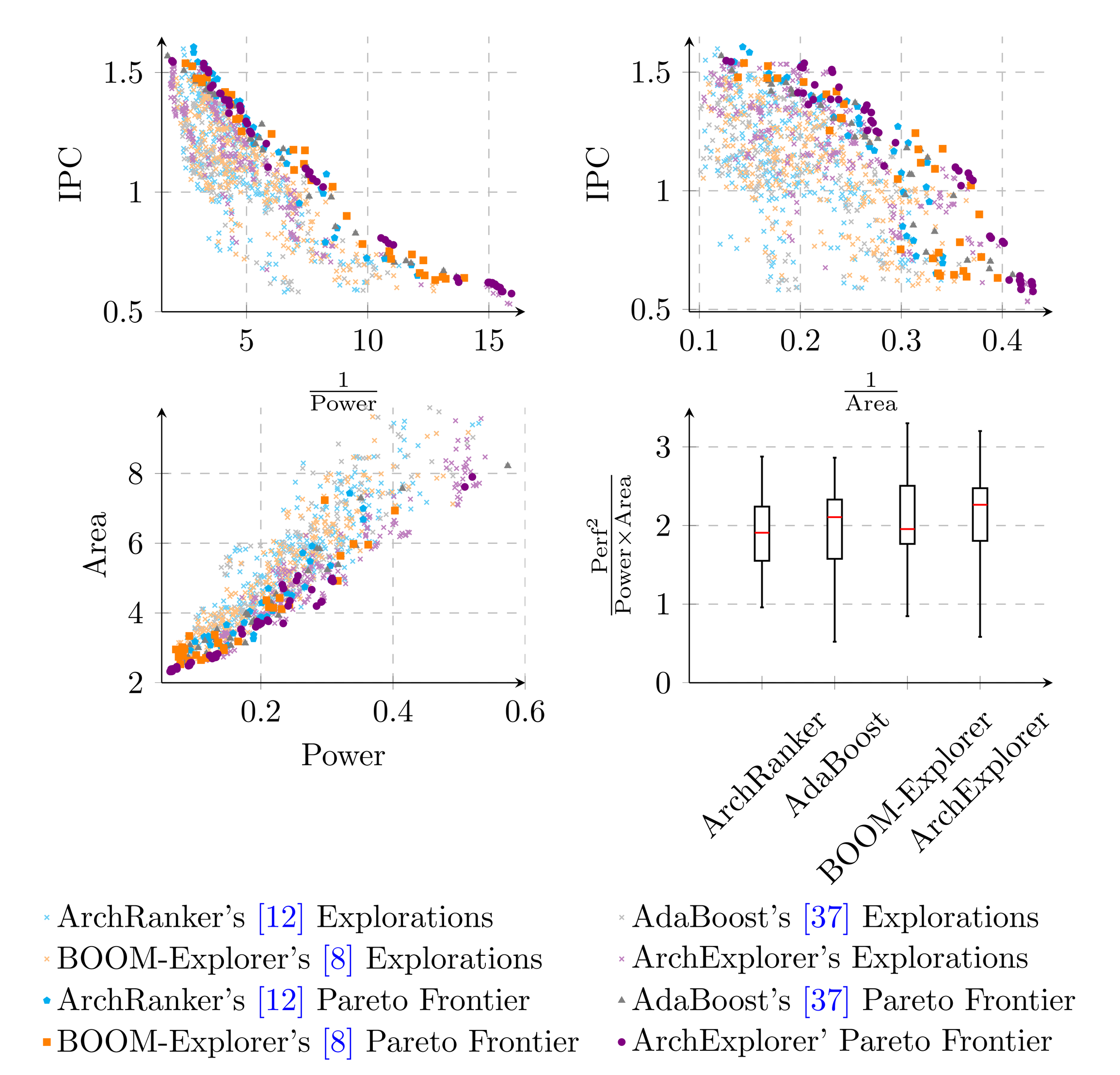

Figure 11. The visualization of Pareto frontiers and the distributions of PPA trade-offs for all methods.

Comparison w. Best Balanced Designs:

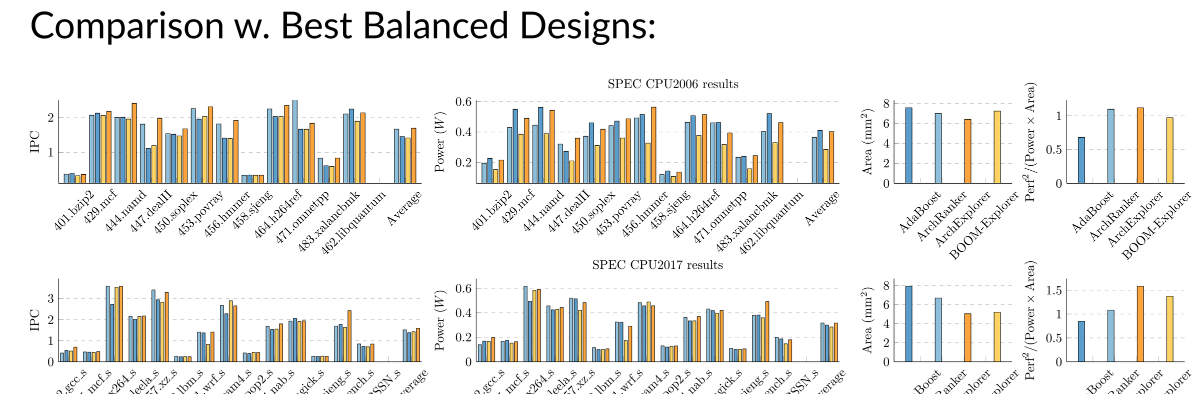

Figure 12. Comparisons between the Pareto designs in performance and power.

- ArchExplorer can find better PPA Pareto-optimal designs, achieving an average of 6.80% higher Pareto hypervolume using at most 74.63% fewer simulations compared to the state-of-the-art approaches.