Towards Automated RISC-V Microarchitecture Design with Reinforcement Learning

Chen Bai[1] Jianwang Zhai[2] Yuzhe Ma[3] Bei Yu[1] Martin D.F. Wong[4]

[1]The Chinese University of Hong Kong [2]Beijing University of Posts and Telecommunications
[3]The Hong Kong University of Science and Technology (Guangzhou)
[4]Hong Kong Baptist University

Introduction

Problem formulation:

RISC-V Microarchitecture Design

- Microprocessor Microarchitecture Design Space Exploration (DSE)

Given the microarchitecture design space and target workloads, how do we efficiently search for optimal microarchitectures that can satisfy the pre-determined performance, power, and area (PPA) design targets?

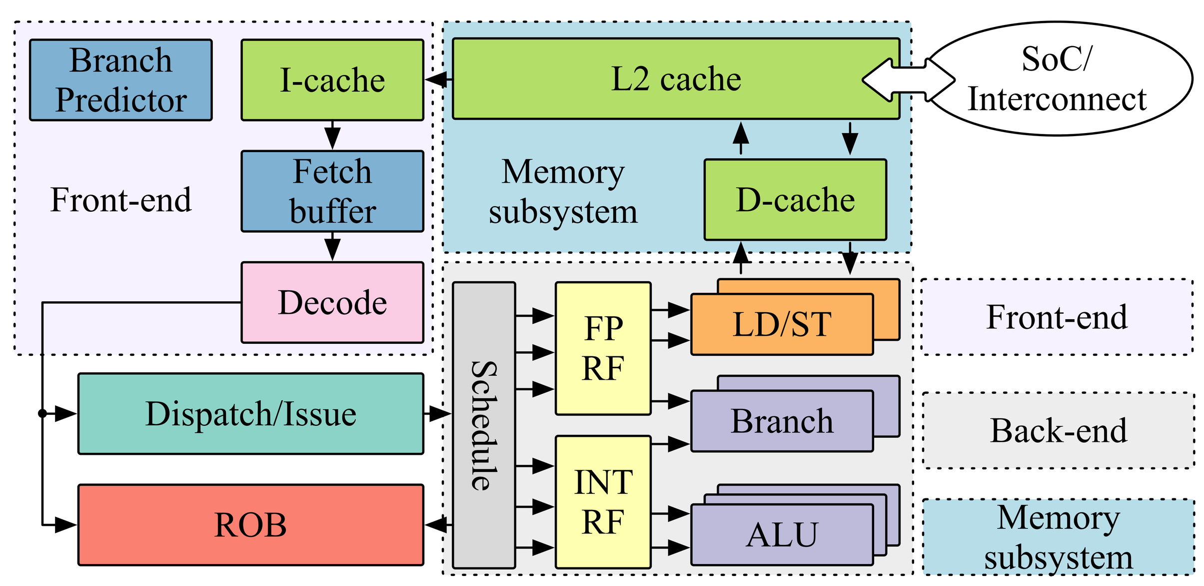

Figure 1. An overview of the example microprocessor microarchitecture, including different *components*.

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. → not tightly coupled with expertknowledge & mathematical limitation in the Gaussian process modeling [1].

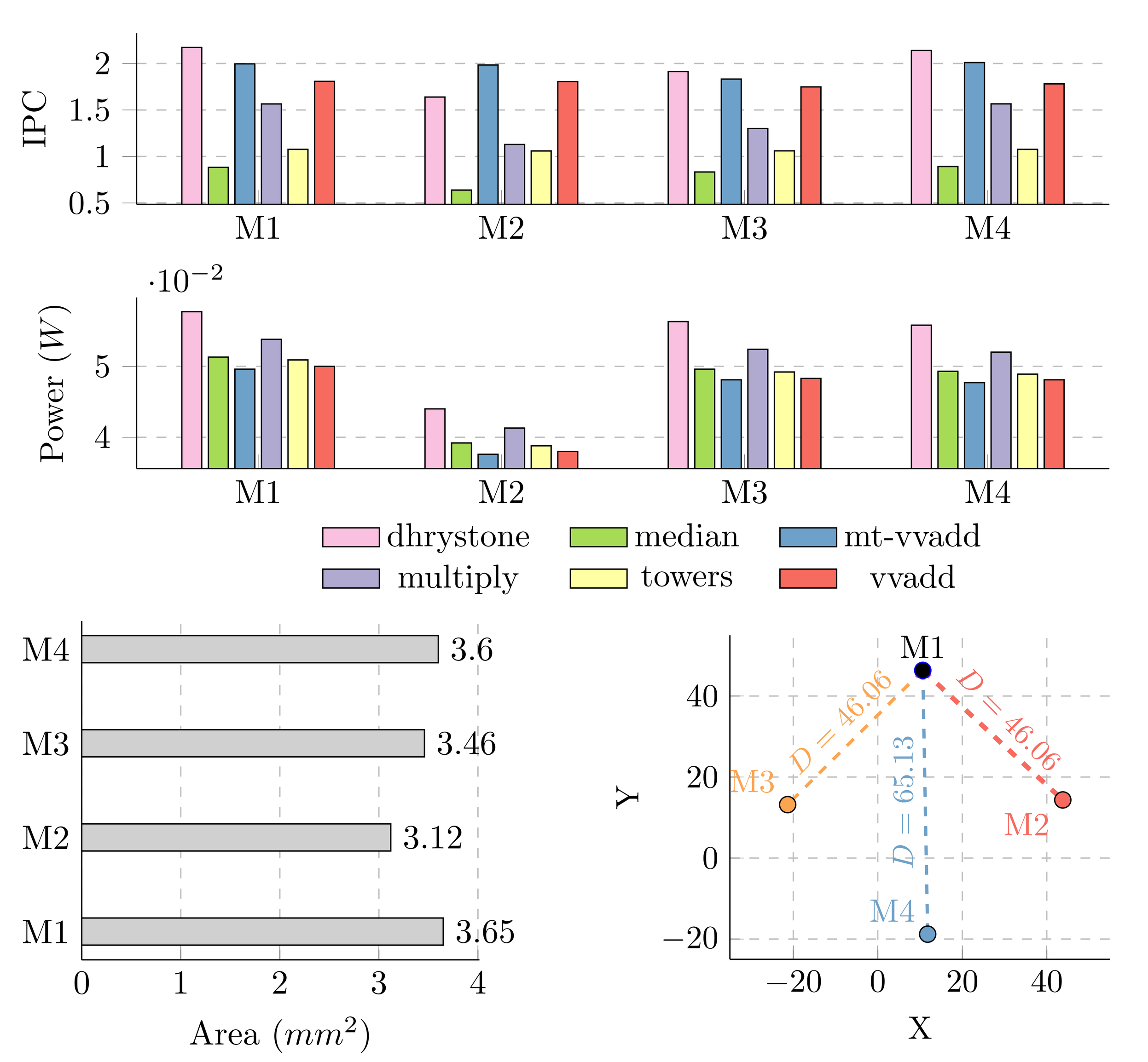

Figure 2. An example of different BOOM microarchitectures to demonstrate the claim.

Limitation of Gaussian Process Modeling

The kernel function of the Gaussian process mathematically attributes the PPA differences between two microarchitectures to the microarchitecture embedding distances.

Highlights of our new black-box methodology:

- Remove mathematical limitation in the Gaussian process modeling (*i.e.*, free of unrealistic assumptions).
- Our method is tightly coupled with expert knowledge: *microarchitecture scaling graph*.
- PPA design preference-driven exploration.
- Lightweight agent training environment design to accelerate the learning process.

Preliminaries

Our RISC-V Microarchitecture Design Space:

Design	Component	Parameters	Candidate
Rocket	Branch predictor	RAS	$0:12:3$ +
		BTB.nEntries	$0:56:14$
		BHT.nEntries	$0:1024:256$
	I-cache	nWays	$1,2,4$
		nTLBWays	$4:32:4$
	Functional unit	FPU	$1,2$
		mulDiv	$1,2,3$
		VM	$1,2$
	D-cache	nSets	$32,64$
		nWays	$1,2,4$
		nTLBWays	$4:32:4$
		nMSHRs	$1,2,3$
Small/Medium Large/Mega Giga SonicBOOM	Branch predictor	Type	$1,2,3$
		maxBrCount	$4:22:2$
	IFU	numFetchBufferEntries	$6:46:2$
		fetchWidth	$4,8$
		ftq.nEntries	$12:64:4$
	pipelineWidth		$1:5:1$
	ROB		$24:160:4$
	PRF	numIntPhysRegisters	$40:176:8$
		numFpPRF	$34:132:6$
	ISU	numFpPhysRegisters	$1:5:1$
		numEntries	$6:52:2$
		dispatchWidth	$1:5:1$
	LSU	LDQ	$6:32:2$
		STQ	$6:36:2$
	I-cache	nWays	$4,8$
		nSets	$32,64$
	D-cache	nWays	$4,8$
		nSets	$64,128$
		nMSHRs	$2:10:2$

* The values are start number:end number:stride, e.g., $0:12:3$ denotes the entries of RAS can be 0, 3, 6, etc., until 12.

Microarchitecture Scaling Graph:

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

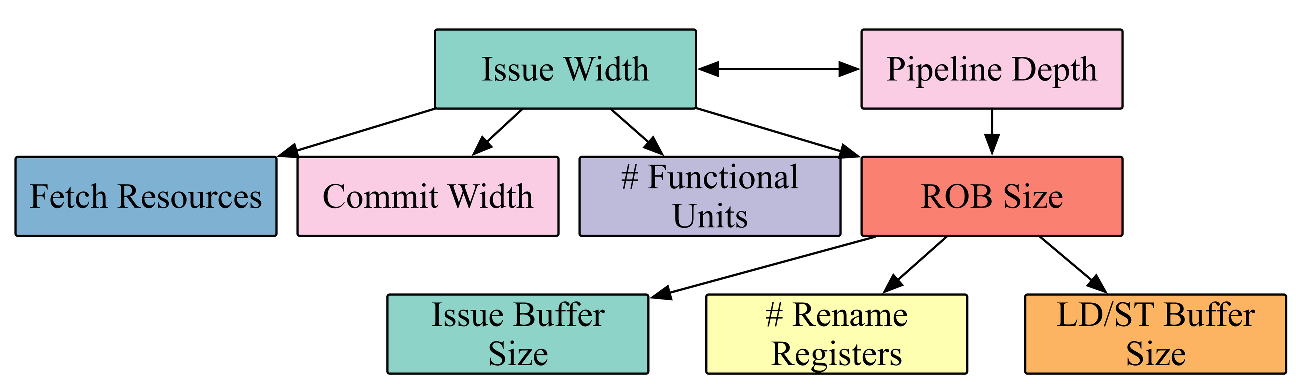

Figure 3. A microarchitecture scaling graph of an example out-of-order microprocessor.

Reinforcement Learning Methodology

Overview:

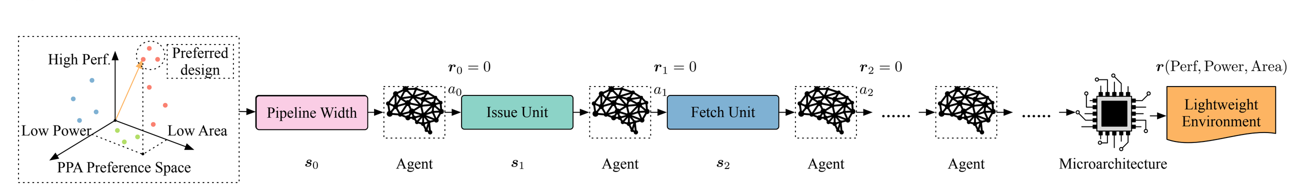

Figure 4. An overview of our reinforcement learning methodology.

Generalized Bellman Optimality Equality:

Generalized Bellman Optimality Equality

$$\boldsymbol{Q}(\boldsymbol{s},a,\boldsymbol{\phi}) = \boldsymbol{r}(\boldsymbol{s},a) + \zeta\mathbb{E}_{\boldsymbol{s}'\sim\mathcal{P}(\cdot|\boldsymbol{s},a)}\mathcal{T}(\boldsymbol{Q}(\boldsymbol{s}',a,\boldsymbol{\phi})),$$
$$\mathcal{T}(\boldsymbol{Q}(\boldsymbol{s}',a,\boldsymbol{\phi})) = \underset{\boldsymbol{Q}}{arg}\ \underset{a'\in A,\boldsymbol{\phi}'\in\Phi}{max}\ \boldsymbol{Q}(\boldsymbol{s}',a',\boldsymbol{\phi}')\boldsymbol{\phi}^\top \qquad (1)$$

ζ is the discount factor, $\boldsymbol{Q}(\boldsymbol{s},a,\boldsymbol{\phi})$ is the state-action vector, and $\boldsymbol{\phi}$ is the PPA design preference.

Optimization with Generalized Bellman Optimality Equality:

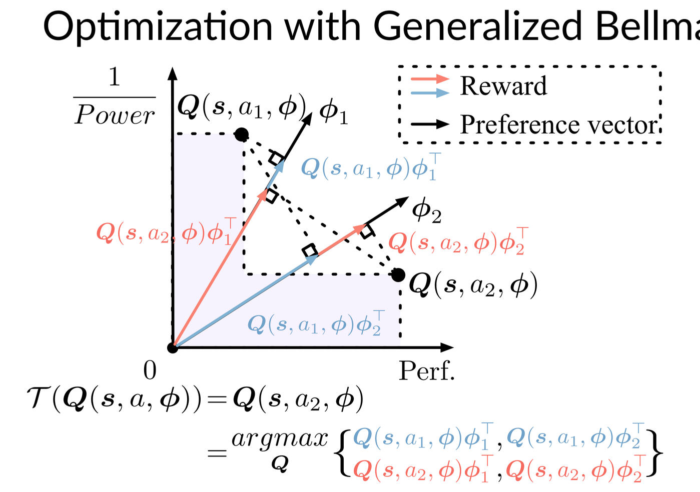

Figure 5. Optimization procedure.

- We adopt the asynchronous advantage actor-critic (A3C).
- We utilize the conditioned neural network design.
- We adopt lightweight environment to accelerate the agent training process.

Experiments

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:

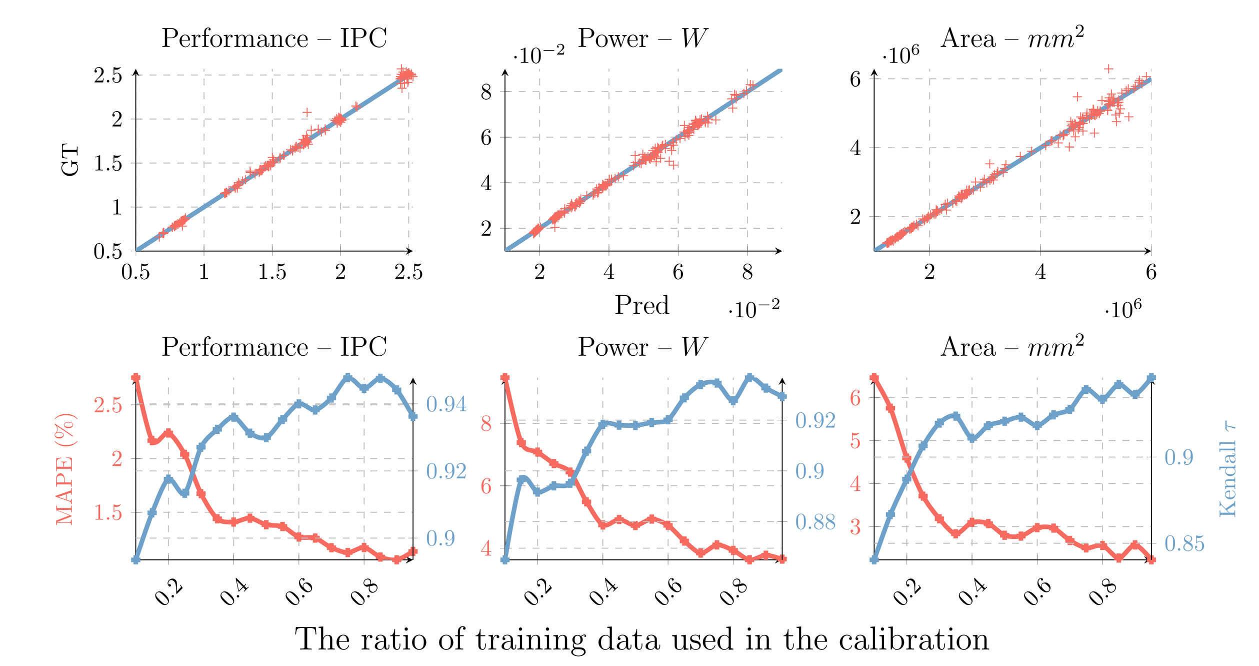

Figure 6. The accuracy of lightweight PPA models, and MAPE and Kendall τ curves *w.r.t.* the calibration data size.

RL Training:

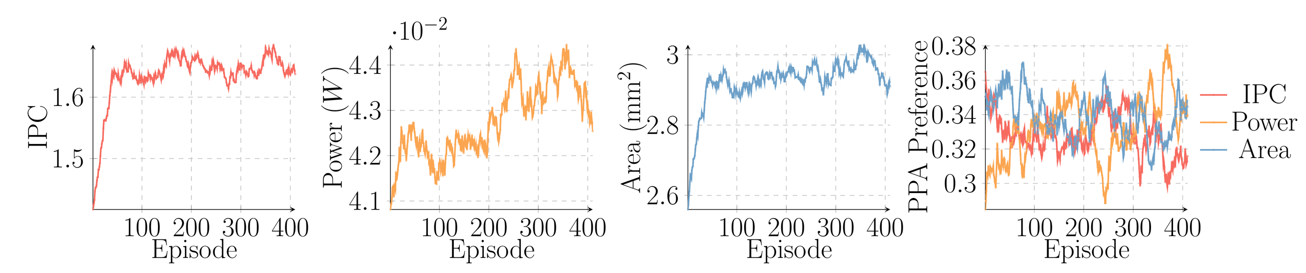

Figure 7. RL training curves for PPA values and sampled PPA preference vectors.

Main Results:

Table 1. Comparison w. Human Efforts & Prior Arts

Design	Method	Performance IPC	Power W	Area mm^2	Perf / Power Val.	Perf / Power Ratio	Perf / Area Val.	Perf / Area Ratio	(Perf × Perf) / (Power × Area) Val.	(Perf × Perf) / (Power × Area) Ratio	Runtime
Rocket	Human Efforts	0.7338	0.0027	0.9082	267.4708	– [1]	0.8080	–	216.1090	–	–
	ISCA'14	0.8157	0.0023	0.7943	**359.3222**	**1.3434×**	1.0270	1.2710×	369.0075	1.7075×	8.6111×
	DAC'16	0.5485	0.0018	0.5337	305.3090	1.1415×	1.0278	1.2721×	313.8042	1.4527×	5.8961×
	ICCAD'21	0.7278	0.0021	0.7448	352.7177	1.3187×	0.9771	1.2093×	344.6327	1.5947×	1.5011×
	Ours	0.7278	0.0023	0.5762	313.6958	1.1728×	**1.2631**	**1.5633×**	**396.2335**	**1.8335×**	**1.0000**
Small SonicBOOM	Human Efforts	0.7837	0.0203	1.5048	38.6057	–	0.5209	–	20.1062	–	–
	ISCA'14	0.8197	0.0150	1.2838	54.7692	1.4187×	0.6385	1.2260×	34.9710	1.7393×	5.8033×
	DAC'16	0.8076	0.0147	1.2512	54.8119	1.4198×	0.6454	1.2393×	35.3765	1.7594×	4.7918×
	ICCAD'21	0.8469	0.0200	1.5026	42.3436	1.0968×	0.5636	1.0821×	23.8645	1.1869×	1.3053×
	Ours	0.8403	0.0152	1.2538	**55.2813**	**1.4320×**	**0.6702**	**1.2868×**	**37.0491**	**1.8427×**	**1.0000**
Medium SonicBOOM	Human Efforts	1.1938	0.0256	1.9332	46.6952	–	0.6175	–	28.8363	–	–
	ISCA'14	1.2362	0.0196	1.6242	**62.9622**	**1.3484×**	**0.7611**	**1.2324×**	**47.9192**	**1.6618×**	5.6879×
	DAC'16	1.3757	0.0254	1.9247	54.0894	1.1584×	0.7148	1.1574×	38.6609	1.3407×	4.6966×
	ICCAD'21	1.4454	0.0271	2.1583	53.3342	1.1422×	0.6697	1.0844×	35.7170	1.2386×	1.2793×
	Ours	1.2872	0.0206	1.7351	62.5886	1.3404×	0.7419	1.2014×	46.4339	1.6103×	**1.0000**
Large SonicBOOM	Human Efforts	1.4871	0.0446	3.2055	33.3430	–	0.4639	–	15.4686	–	–
	ISCA'14	1.4900	0.0309	2.5420	48.2184	1.4461×	0.5861	1.2634×	28.2626	1.8271×	5.8920×
	DAC'16	1.4919	0.0324	2.6744	45.9976	1.3795×	0.5578	1.2024×	25.6592	1.6588×	4.8651×
	ICCAD'21	1.9162	0.0409	3.6715	46.8507	1.4051×	0.5219	1.1250×	24.4520	1.5808×	1.3252×
	Ours	1.5882	0.0314	2.5643	**50.6324**	**1.5185×**	**0.6193**	**1.3350×**	**31.3580**	**2.0272×**	**1.0000**
Mega SonicBOOM	Human Efforts	1.9500	0.0578	4.8059	33.7571	–	0.4058	–	13.6972	–	–
	ISCA'14	2.4957	0.0566	5.3676	44.0942	1.3062×	0.4650	1.1459×	20.5020	1.4968×	5.5443×
	DAC'16	2.4995	0.0562	5.3797	44.4483	1.3167×	0.4646	1.1451×	20.6513	1.5077×	4.5780×
	ICCAD'21	2.4823	0.0607	4.7008	40.9170	1.2121×	**0.5281**	**1.3014×**	21.6066	1.5774×	1.2470×
	Ours	2.5232	0.0557	5.2512	**45.3005**	**1.3420×**	0.4805	1.1842×	**21.7674**	**1.5892×**	**1.0000**
Giga SonicBOOM	Human Efforts	1.8717	0.0716	5.0691	26.1538	–	0.3692	–	9.6572	–	–
	ISCA'14	2.2528	0.0622	6.0010	36.2192	1.3849×	0.3754	1.0167×	13.5970	1.4080×	5.6321×
	DAC'16	2.2522	0.0773	5.5995	29.1480	1.1145×	**0.4022**	**1.0893×**	11.7236	1.2140×	4.6505×
	ICCAD'21	2.2650	0.0745	5.8652	30.4162	1.1630×	0.3862	1.0459×	11.7460	1.2163×	1.2668×
	Ours	2.2692	0.0595	5.7459	**38.1587**	**1.4590×**	0.3949	1.0695×	**15.0696**	**1.5605×**	**1.0000**

[1] "–" denotes not applicable.

Comparison w. Best Balanced Designs:

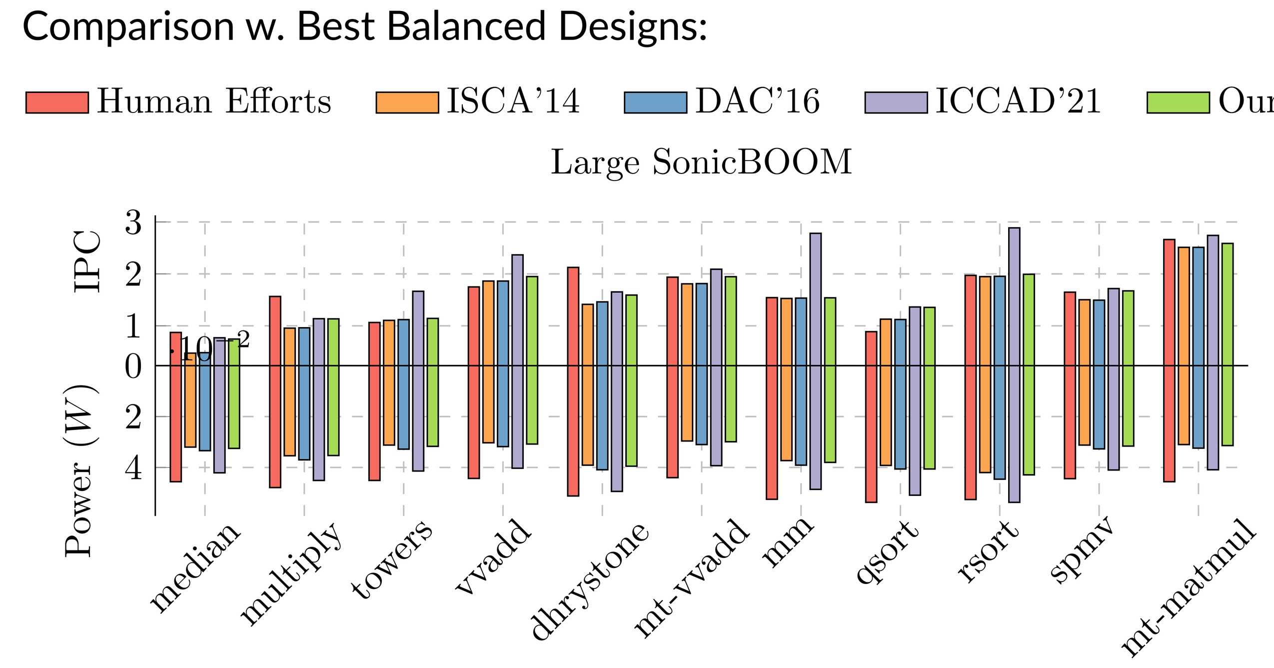

Figure 8. Analysis with more workloads for large-scale SonicBOOM.

References

[1] Chen Bai, Qi Sun, Jianwang Zhai, Yuzhe Ma, Bei Yu, and MD Wong. BOOM-Explorer: RISC-V BOOM Microarchitecture Design Space Exploration Framework. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–9, 2021.

The 38th Annual AAAI Conference on Artificial Intelligence, Vancouver, Canada