BOOM-Explorer: RISC-V BOOM Microarchitecture Design Space Exploration Framework

Chen Bai[1], Qi Sun[1], Jianwang Zhai[2], Yuzhe Ma[1], Bei Yu[1], Martin D.F. Wong[1]

[1]The Chinese University of Hong Kong
[2]Tsinghua University
{cbai,byu}@cse.cuhk.edu.hk

Nov. 1, 2021

Outline

Introduction

ISA

- X86
- ARM
- RISC-V
  : open-source & numerous designs

ISA

- X86
- ARM
- RISC-V
  😊: open-source & numerous designs

Microarchitecture

- An **implementation** of an ISA in a processor.
- It can affect the performance, power dissipation, area, and *etc.* of a processor.

- Parametric among different modules.
  - Cache structures
  - Decoders
  - Execution units
  - Load and store unit
  - Key buffers, queues & stacks
  - *etc.*
- Discrete candidate values.
- They are important to performance, power and area (PPA) values.

Related Work

In industry:

- Rely on engineering experience of CPU architects.

In academia:

- ANN-based model [İpek et al. 2006]
- Regression-based model [Lee and Brooks 2007]
- AdaBoost-based model [Li et al. 2016]

Limitations

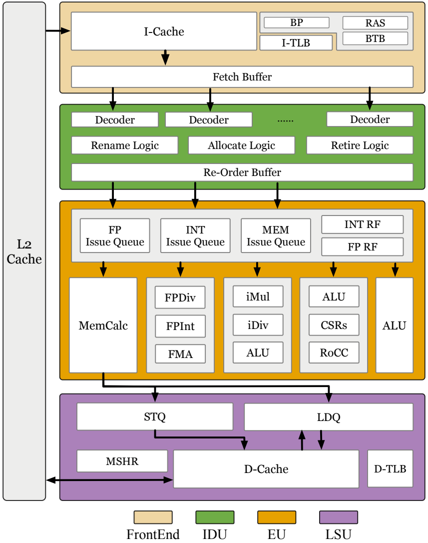
Industry solutions:

- Lacks scalability for newly emerged processors.

Academic solutions:

- Fail to embed prior knowledge of microarchitecture designs to algorithms.
- Lack discussions on striking a good balance between the performance and power dissipation of a microarchitecture design.

Preliminaries



Overview of RISC-V BOOM [Asanovic, Patterson, and C. Celio 2015] [C. P. Celio 2017]

Table: Microarchitecture Design Space of BOOM

Module	Component	Descriptions	Candidate Values
FrontEnd	FetchWidth	Number of instructions the fetch unit can retrieve once	4, 8
	FetchBufferEntry	Entries of the fetch buffer register	8, 16, 24, 32, 35, 40
	RasEntry	Entries of the Return Address Stack (RAS)	16, 24, 32
	BranchCount	Entries of the Branch Target Buffer (BTB)	8, 12, 16, 20
	ICacheWay	Associate sets of L1 I-Cache	2, 4, 8
	ICacheTLB	Entries of Table Look-aside Buffer (TLB) in L1 I-Cache	8, 16, 32
	ICacheFetchBytes	Unit of line capacity that L1 I-Cache supports	2, 4
IDU	DecodeWidth	Number of instructions the decoding unit can decode once	1, 2, 3, 4, 5
	RobEntry	Entries of the reorder buffer	32, 64, 96, 128, 130
	IntPhyRegister	Number of physical integer registers	48, 64, 80, 96, 112
	FpPhyRegister	Number of physical floating-point registers	48, 64, 80, 96, 112
EU	MemIssueWidth	Number of memory-related instructions that can issue once	1, 2
	IntIssueWidth	Number of integer-related instructions that can issue once	1, 2, 3, 4, 5
	FpIssueWidth	Number of floating-point-related instructions that can issue once	1, 2
LSU	LDQEntry	Entries of the Loading Queue (LDQ)	8, 16, 24, 32
	STQEntry	Entries of the Store Queue (STQ)	8, 16, 24, 32
	DCacheWay	Associate sets of L1 D-Cache	2, 4, 8
	DCacheMSHR	Entries of Miss Status Handling Register (MSHR)	2, 4, 8
	DCacheTLB	Entries of Table Look-aside Buffer (TLB) in L1 D-Cache	8, 16, 32

Table: Constraints of BOOM design specifications

Rule	Descriptions
1	FetchWdith $\geq$ DecodeWidth
2	RobEntry $\mid$ DecodeWidth [+]
3	FetchBufferEntry $>$ FetchWidth
4	FetchBufferEntry $\mid$ DecodeWidth
5	fetchWidth $=$ 2$\times$ ICacheFetchBytes
6	IntPhyRegister $=$ FpPhyRegister
7	LDQEntry $=$ STQEntry
8	MemIssueWidth $=$ FpIssueWidth

[+] The symbol "|" means RobEntry should be divisible by DecodeWidth

Definition (Microarchitecture)

A combination of candidate values defined in Table 1.
A legal microarchitecture is encoded as a feature vector $\boldsymbol{x} \in \mathcal{D}$.

Definition (Power)

The power is to be defined as

$$P = P_{\text{dynamic}} + P_{\text{short-circuit}} + P_{\text{leakage}}. \tag{1}$$

Definition (Clock Cycle)

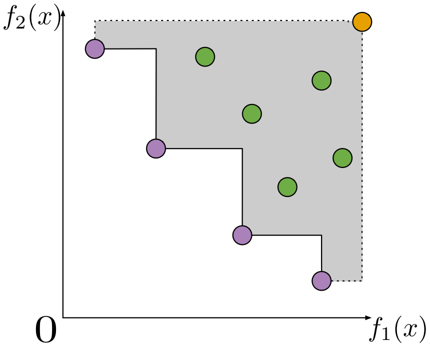
Clock cycles are consumed when a BOOM design runs a benchmark.

Definition (Pareto Optimality)

An objective vector $f(\boldsymbol{x})$ is said to be dominated by $f(\boldsymbol{x}')$ in a n-dimensional space if

$$\begin{aligned} \forall i \in [1,n], \quad f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{x}') \\ \exists j \in [1,n], \quad f_j(\boldsymbol{x}) < f_j(\boldsymbol{x}') \end{aligned} \tag{2}$$

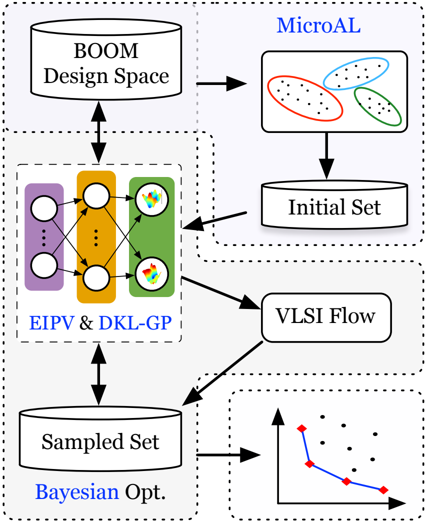
and we denote $\boldsymbol{x}' \succcurlyeq \boldsymbol{x}$.
Pareto-optimal set is $\boldsymbol{x} \in \mathcal{D}$ that are not dominated by any other.



Problem (BOOM Microarchitecture Design Space Exploration)

In the design space $\mathcal{D}$, find a series of microarchitectures $\boldsymbol{X}$ that form the Pareto optimality among power and performance $\boldsymbol{Y} \in \mathcal{Y}$.
Hence, $\boldsymbol{Y} = \{\boldsymbol{y} | \boldsymbol{y}' \nsucceq \boldsymbol{y}, \forall \boldsymbol{y}' \in \mathcal{Y}\}$, $\boldsymbol{X} = \{\boldsymbol{x} | f(\boldsymbol{x}) \in \boldsymbol{Y}, \forall \boldsymbol{x} \in \mathcal{D}\}$.

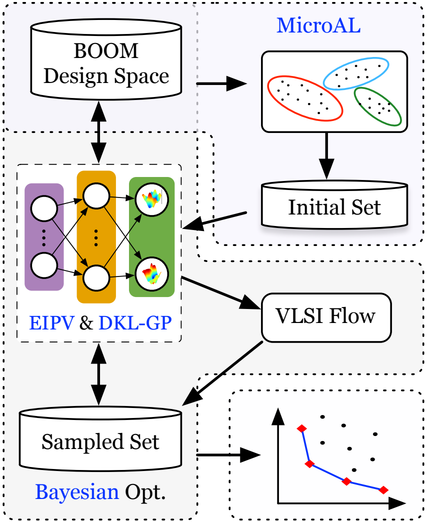
BOOM-Explorer



An Overview of BOOM-Explorer

Highlights

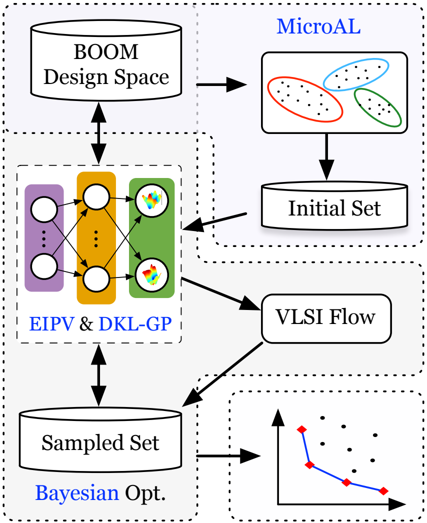
- Embed prior knowledge on Transductive Experimental Design [Yu, Bi, and Tresp 2006] – (MicroAL)



An Overview of BOOM-Explorer

Highlights

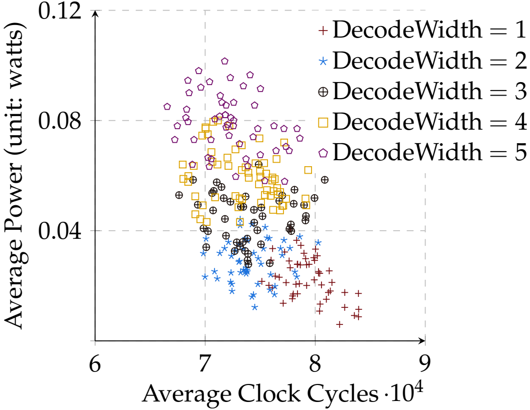
- Embed prior knowledge on Transductive Experimental Design [Yu, Bi, and Tresp 2006] – (MicroAL)
- Gaussian process with Deep Kernel Learning (DKL-GP)



An Overview of BOOM-Explorer

Highlights

- Embed prior knowledge on Transductive Experimental Design [Yu, Bi, and Tresp 2006] – (MicroAL)
- Gaussian process with Deep Kernel Learning (DKL-GP)
- Bayesian Optimization with Expectation Improvement on Pareto Hypervolume (EIPV)



Clustering *w.r.t.* DecodeWidth

Algorithm 1 **TED**$(\mathcal{U}, \mu, b)$

Require: $\mathcal{U}$ is the unsampled microarchitecture design space, μ is a normalization coefficient, and b is the number of samples to draw.

Ensure: $\mathcal{X}$: the sampled set with $|\mathcal{X}| = b$.

1: $\mathcal{X} \leftarrow \emptyset$, $\mathrm{K}_{uu'} \leftarrow f(u, u')$, $\forall u, u' \in \mathcal{U}$;
2: **for** $i = 1 \rightarrow b$ **do**
3: $\quad \boldsymbol{x}_* \leftarrow \arg\max_{x \in \mathcal{U}} \mathrm{Tr}[\mathrm{K}_{\mathcal{U}x}(\mathrm{K}_{xx} + \mu \boldsymbol{I})^{-1}\mathrm{K}_{x\mathcal{U}}]$; $\triangleright$ $\mathrm{K}_{\mathcal{U}x}$, K_{xx} and $\mathrm{K}_{x\mathcal{U}}$ are calculated via f *w.r.t.* corresponding columns in K
4: $\quad \mathcal{X} \leftarrow \mathcal{X} \cup \boldsymbol{x}_*$, $\mathcal{U} \leftarrow \mathcal{U} \setminus \boldsymbol{x}_*$;
5: $\quad \mathrm{K} \leftarrow \mathrm{K} - \mathrm{K}_{\mathcal{U}\boldsymbol{x}_*}(\mathrm{K}_{\boldsymbol{x}_*\boldsymbol{x}_*} + \mu \boldsymbol{I})^{-1}\mathrm{K}_{\boldsymbol{x}_*\mathcal{U}}$;
6: **end for**
7: **return** The sampled set $\mathcal{X}$;

Algorithm 2 **MicroAL**$(\mathcal{U}, \mu, b)$

Require: $\mathcal{U}$ is the unsampled microarchitecture design space, μ is a normalization coefficient, b is the number of samples that to draw.
Ensure: $\mathcal{X}$: the sampled set with $|\mathcal{X}| = b$.

$\mathcal{X} \leftarrow \emptyset$;
initialize k clusters randomly with the centroids set $\boldsymbol{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_k\}$ from $\mathcal{U}$;
while not converged **do**
  $c^i = \underset{j \in \{1,2,\ldots,k\}}{\arg\min}\ \Phi\left(\boldsymbol{x}_i - \boldsymbol{c}_j\right),\ \forall \boldsymbol{x}_i \in \mathcal{U}$; ▷ Φ is the distance function considering DecodeWidth
  $\boldsymbol{c}_j = \dfrac{\sum_{i=1}^{|\mathcal{U}|} \mathbb{1}\{c^i = j\} \boldsymbol{x}_i}{\sum_{i=1}^{|\mathcal{U}|} \mathbb{1}\{c_i = j\}},\ \forall j \in \{1, 2, ..., k\}$;
end while
$\mathcal{C} \leftarrow$ neighborhood of $\boldsymbol{c}_i \in \boldsymbol{C},\ \forall i \in \{1, 2, ..., k\}$;
for $\mathcal{K}$ in $\mathcal{C}$ **do**
  $\hat{\mathcal{X}} = \textbf{TED}(\mathcal{K}, \mu, \lfloor \frac{b}{k} \rfloor)$; ▷ Algorithm 1
  $\mathcal{X} = \mathcal{X}_{\rangle} \cup \hat{\mathcal{X}}$;
end for
return The sampled set $\mathcal{X}$;

Gaussian Process (GP)

Feature vectors: $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_n\}$,
Corresponding power or clock cycles $\boldsymbol{y} = \{y_1, y_2, \ldots, y_n\}$.
Gaussian distributions can be constructed *w.r.t.* Equation (3),

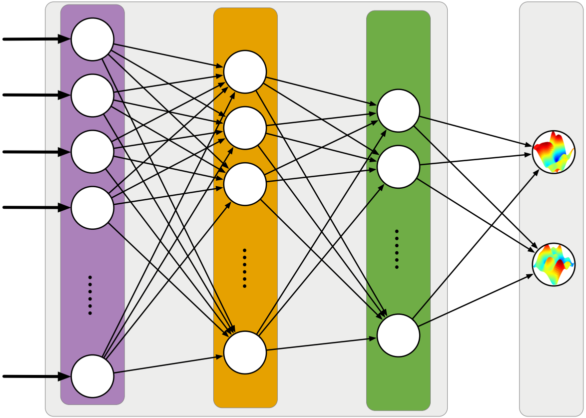
$$\boldsymbol{f} = [f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots f(\boldsymbol{x}_n)]^T \sim \mathcal{N}(\boldsymbol{\mu}, \mathrm{K}_{\boldsymbol{X}\boldsymbol{X}|\boldsymbol{\theta}}). \tag{3}$$

Given a newly sampled $\boldsymbol{x}_*$, the predictive joint distribution f_* can be obtained,

$$f_*|\boldsymbol{y} \sim \mathcal{N}(\begin{bmatrix} \boldsymbol{\mu} \\ \mu_* \end{bmatrix}, \begin{bmatrix} \mathrm{K}_{\boldsymbol{X}\boldsymbol{X}|\boldsymbol{\theta}} + \sigma_e^2 \boldsymbol{I} & \mathrm{K}_{\boldsymbol{X}\boldsymbol{x}_*|\boldsymbol{\theta}} \\ \mathrm{K}_{\boldsymbol{x}_*\boldsymbol{X}|\boldsymbol{\theta}} & k_{\boldsymbol{x}_*\boldsymbol{x}_*|\boldsymbol{\theta}} \end{bmatrix}). \tag{4}$$

Embed Deep Kernel Learning in GP [Wilson et al. 2016]

$$k_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \rightarrow k_{\boldsymbol{w},\boldsymbol{\theta}}(\varphi(\boldsymbol{x}_i, \boldsymbol{w}), \varphi(\boldsymbol{x}_j, \boldsymbol{w})) \tag{5}$$



Overview of DKL-GP

Our Target

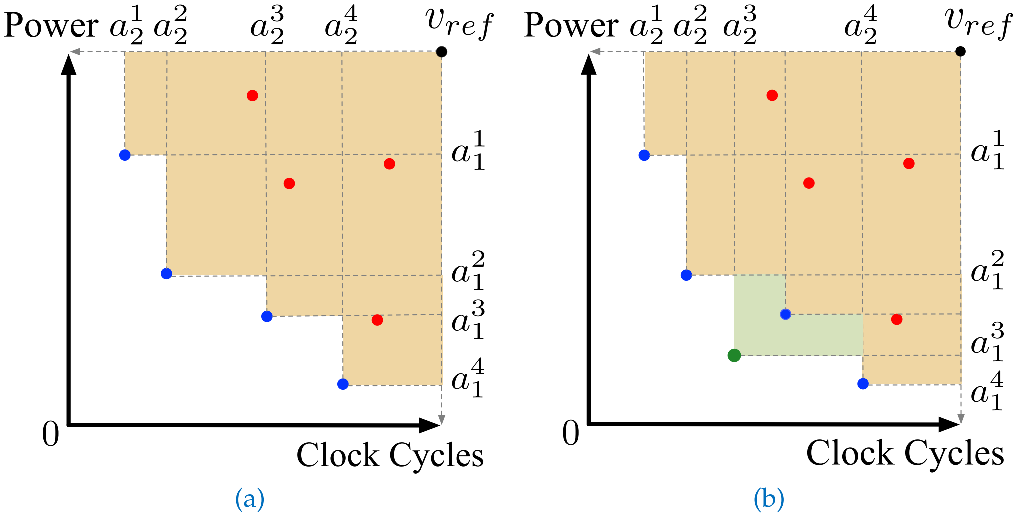
- Performance & Power dissipation.
- A naive objective functions:

$$L = \alpha \cdot \text{Performance} + \beta \cdot \text{Power} \tag{6}$$

- Limitations: Personal bias on α and β.

Pareto Hypervolume [Shah and Ghahramani 2016]

$$\mathrm{PVol}_{\boldsymbol{v}_{\text{ref}}}(\mathcal{P}(\mathcal{Y})) = \int_{\mathcal{Y}} \mathbb{1}[\boldsymbol{y} \succcurlyeq \boldsymbol{v}_{\text{ref}}][1 - \prod_{\boldsymbol{y}_* \in \mathcal{P}(\mathcal{Y})} \mathbb{1}[\boldsymbol{y}_* \nsucceq \boldsymbol{y}]]\mathrm{d}\boldsymbol{y} \tag{7}$$



(a) Red circles: dominated microarchitectures, blue circles: currently explored Pareto-optimal set, orange region: dominated by the Pareto-optimal set.
(b) Green circles: newly-explored Pareto microarchitecture.

Expected Improvement of Pareto Hypervolume (EIPV) as Acquisition Function
[Shah and Ghahramani 2016]

EIPV as our acquisition function:

$$\mathrm{EIPV}(\boldsymbol{x}'|\mathcal{D}) = \mathbb{E}_{p(f(\boldsymbol{x}')|\mathcal{D})}[\mathrm{PVol}_{v_{\mathrm{ref}}}(\mathcal{P}(\mathcal{Y}) \cup f(\boldsymbol{x}')) - \mathrm{PVol}_{v_{\mathrm{ref}}}(\mathcal{P}(\mathcal{Y}))] \tag{8}$$

Rephrase EIPV by decomposing the power-performance space as grid cells:

$$\mathrm{EIPV}(\boldsymbol{x}'|\mathcal{D}) = \sum_{C \in \mathcal{C}_{\mathrm{nd}}} \int_{C} \mathrm{PVol}_{v_C}(\boldsymbol{y}) p(\boldsymbol{y}, |\mathcal{D}) \mathrm{d}\boldsymbol{y} \tag{9}$$

Algorithm 3 BOOM Explorer$(\mathcal{D}, T, \mu, b)$

Require: $\mathcal{D}$ is the microarchitecture design space, T is the maximal iteration number, μ is a normalization coefficient and b is the number of samples to draw.
Ensure: Pareto-optimal set $\boldsymbol{X}$ that forms Pareto optimality among $\mathcal{D}$.

```
X_0 ← MicroAL(D, μ, b);                                                    ▷ Algorithm 2
Push X_0 to VLSI flow to obtan corresonding power and clock cycles Y;
L ← X_0;
U ← D \ L;
for i = 1 ← T do
    Establish and train DKL-GP on L with Y;
    x_* ← arg max_{x∈U} EIPV(x|U);                                         ▷ Equation (9)
    Push x_* to VLSI flow to obtain corresponding power and clock cycles and add to Y;
    L ← L ∪ x_*, U ← U \ x_*;
end for
Construct Pareto-optimal set X from L;
return Pareto-optimal set X;
```

Experiments

Tools

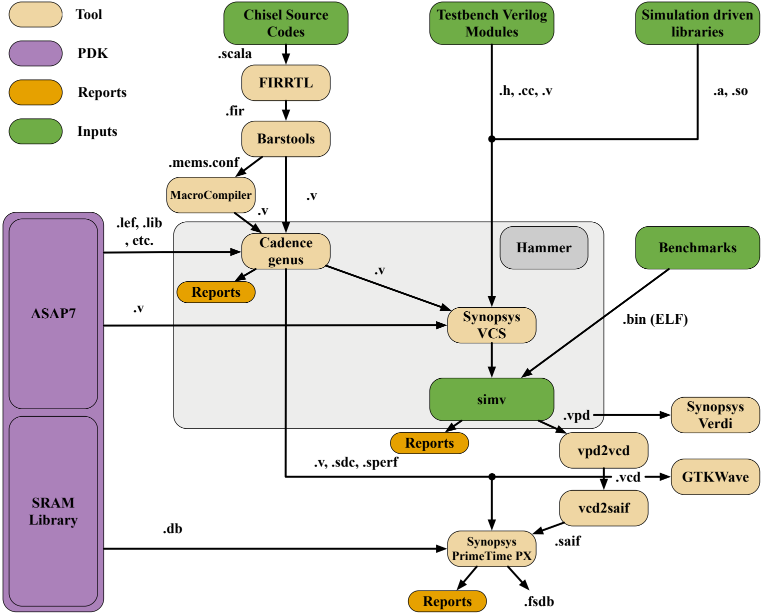
- Technology: ASAP7 PVT Cells [Vashishtha, Vangala, and Clark 2017]
- Scala environment: sbt v1.4.4
- Synthesis: Cadence Genus 18.12-e0121
- RTL simulator: VCS M-2017.03
- Power measurer: PrimeTime PX R-2020.09-SP1

Data

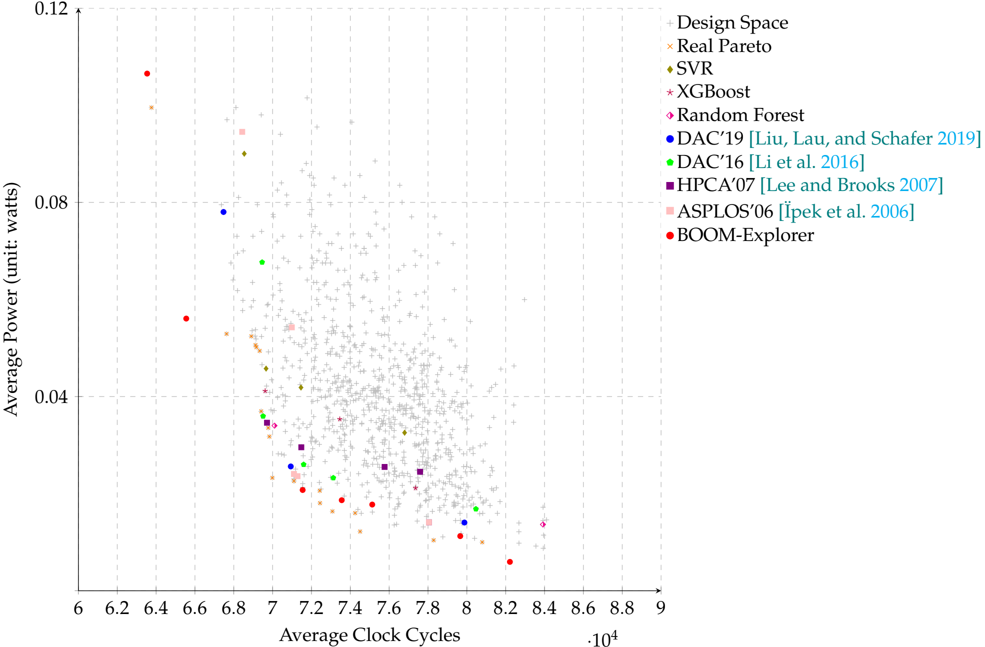
- Data: 994 RTL designs, running at 2GHz
- Representative Benchmarks: median, whetstone, mt-vvadd, mm

Baselines

- ANN-based model [İpek et al. 2006]
- Regression-based model [Lee and Brooks 2007]
- AdaBoost-based model [Li et al. 2016]
- The HLS-predictive model-based method [Liu, Lau, and Schafer 2019]
- Traditional ML model: SVR, XGBoost, Random Forest



Evaluation Flow


0.12
0.08
0.04
Average Power (unit: watts)
6
6.2
6.4
6.6
6.8
7
7.2
7.4
7.6
7.8
8
8.2
8.4
8.6
8.8
9
Average Clock Cycles
$\cdot 10^4$
Design Space
Real Pareto
SVR
XGBoost
Random Forest
DAC'19 [Liu, Lau, and Schafer 2019]
DAC'16 [Li et al. 2016]
HPCA'07 [Lee and Brooks 2007]
ASPLOS'06 [Ïpek et al. 2006]
BOOM-Explorer

Average distance to reference set (ADRS)

$$\mathrm{ADRS}(\Gamma, \Omega) = \frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} \min_{\omega \in \Omega} f(\gamma, \omega) \tag{10}$$

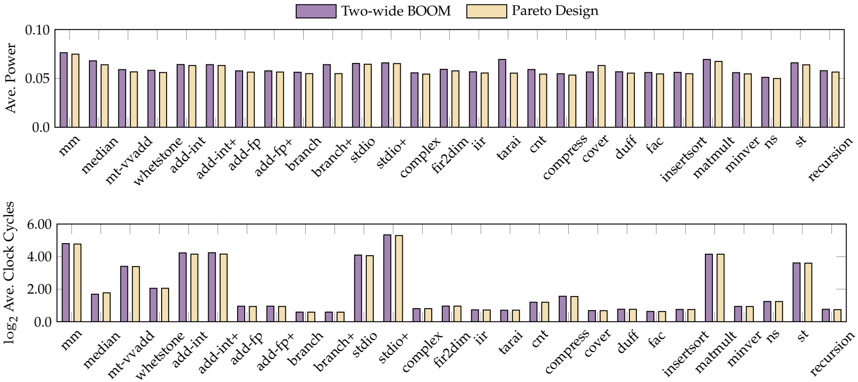
Table: Normalized Experimental Results

Methodologies	Normalized ADRS	Normalized ORT [+]
SVR	0.2399	1.0000
Random Forest	0.2263	0.9763
XGBoost	0.2171	1.010
ASPLOS'06 [Ïpek et al. 2006]	0.1948	0.9437
HPCA'07 [Lee and Brooks 2007]	0.1907	0.8544
DAC'16 [Li et al. 2016]	0.1473	3.0102
DAC'19 [Liu, Lau, and Schafer 2019]	0.1884	0.8973
BOOM-Explorer w/o MicroAL	0.1441	**0.3307**
BOOM-Explorer	**0.1145**	0.3556

[+] ORT: Overall Running Time

Table: Comparison with a two-wide BOOM

Micro-architecture Design	Design Parameters	Average Power (unit: watts)	Average Clock Cycles
Two-wide BOOM [1]	[4, 16, 32, 12, 4, 8, 2, 2, 64, 80, 64, 1, 2, 1, 16, 16, 4, 2, 8]	6.0700×10^{-2}	74915.2963
Pareto Design	[4, 16, 16, 8, 2, 8, 2, 2, 32, 64, 64, 1, 3, 1, 24, 24, 8, 4, 8]	5.8600×10^{-2}	73333.7407



Pareto Design vs. Two-wide BOOM *w.r.t.* Power and Performance

Why Pareto Design Performs Better?

- More hardware resources for LDQ and STQ.
- Larger associative sets and MSHR entries for D-Cache to alleviate access conflicts.
- Assign less resources for RAS and BTB $\rightarrow$ application driven.

Why BOOM-Explorer is effective?

- MicroAL: Embed prior knowledge on microarchitecture design.
- DKL-GP: A robust non-parametric black-box model.
- EIPV: A good design of acquisition function in achieving a trade-off between power and performance.

THANK YOU!

Krste Asanovic, David A Patterson, and Christopher Celio (2015). *The berkeley out-of-order machine (BOOM): An industry-competitive, synthesizable, parameterized RISC-V processor*. Tech. rep. University of California at Berkeley.

Christopher Patrick Celio (2017). *A Highly Productive Implementation of an Out-of-Order Processor Generator*. eScholarship, University of California.

Engin İpek et al. (2006). "Efficiently exploring architectural design spaces via predictive modeling". In: *ACM SIGOPS Operating Systems Review* 40.5, pp. 195–206.

Benjamin C Lee and David M Brooks (2007). "Illustrative design space studies with microarchitectural regression models". In: *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 340–351.

Dandan Li et al. (2016). "Efficient design space exploration via statistical sampling and AdaBoost learning". In: *ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6.

Shuangnan Liu, Francis CM Lau, and Benjamin Carrion Schafer (2019). "Accelerating FPGA Prototyping through Predictive Model-Based HLS Design Space Exploration". In: *ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6.

Amar Shah and Zoubin Ghahramani (2016). "Pareto frontier learning with expensive correlated objectives". In: *International Conference on Machine Learning (ICML)*, pp. 1919–1927.

Vinay Vashishtha, Manoj Vangala, and Lawrence T Clark (2017). "ASAP7 predictive design kit development and cell design technology co-optimization". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 992–998.

Andrew Gordon Wilson et al. (2016). "Deep kernel learning". In: *Artificial intelligence and statistics*. PMLR, pp. 370–378.

Kai Yu, Jinbo Bi, and Volker Tresp (2006). "Active Learning via Transductive Experimental Design". In: *International Conference on Machine Learning (ICML)*, pp. 1081–1088.