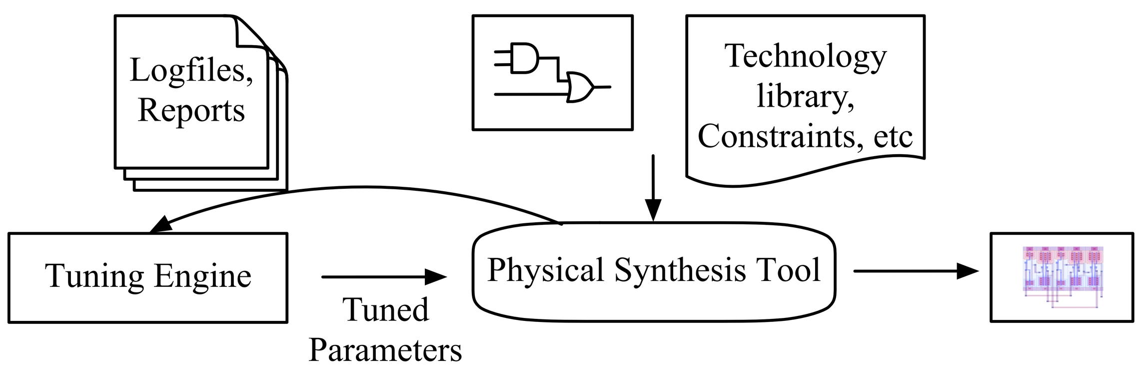
RankTuner: When Design Tool Parameter Tuning Meets Preference Bayesian Optimization

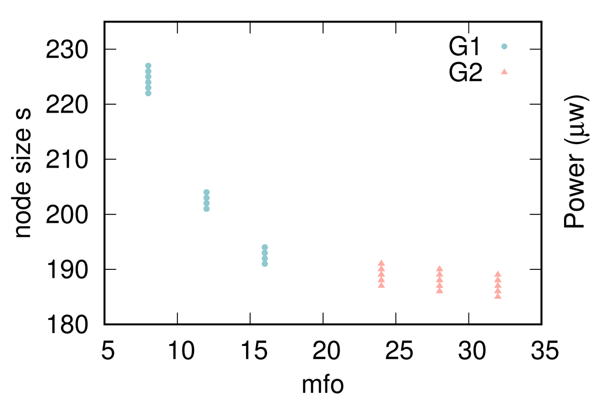
Peng XU, supervised by Prof. Bei Yu

SID:1155187705, pxu22@cse.cuhk.edu.hk

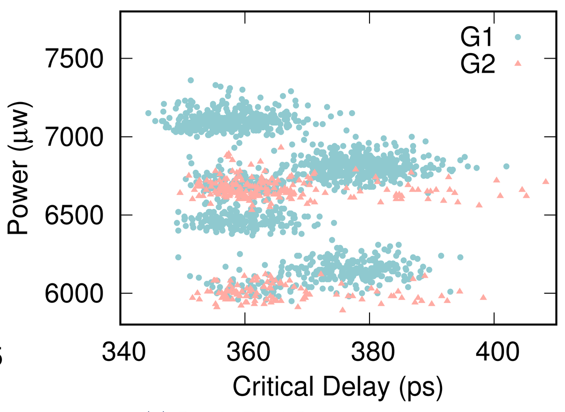
Starting from EDA Tool Parameter Tuning [3]



(a) EDA tool parameter tuning

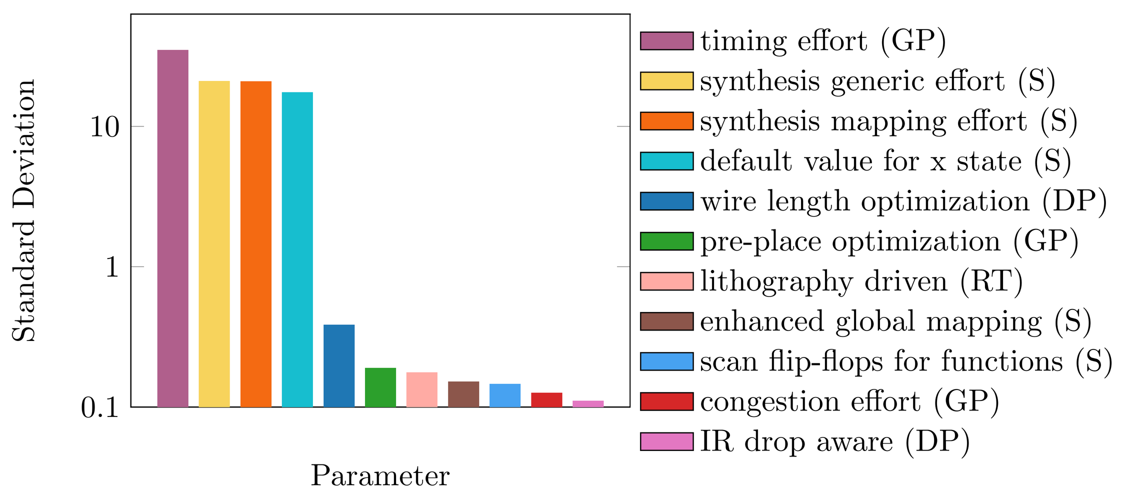


(b) Front-End Perspective



(c) Back-End Perspective

High-dimensional Black-box Optimization [12]



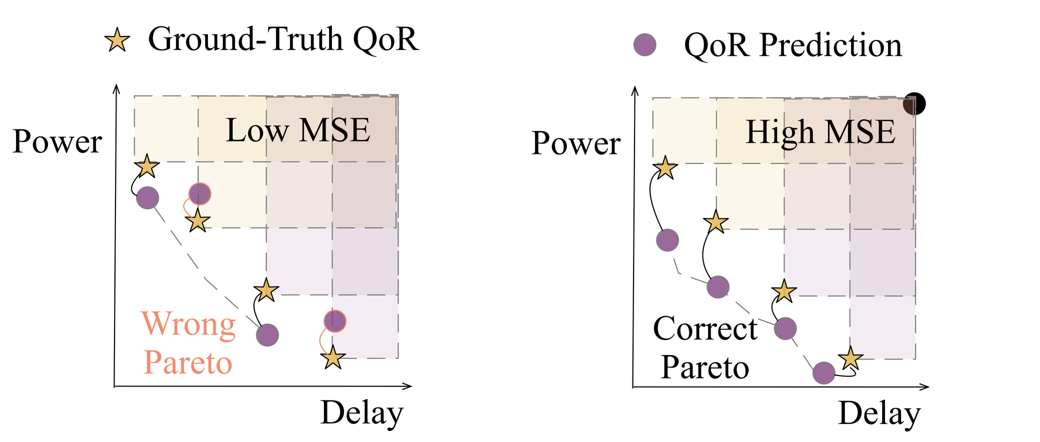
- **High-dimensional**: A lot of values of design parameters need to be determined or tuned ($n_{\text{Params}} \geq 150$)
- Multiple quality-of-result (QoR) metrics (e.g., area, power, and delay) to be optimized
- **"Black-box"** parameter-to-performance mappings: no explicit function expressions
- **Time-consuming** EDA tool evaluation, i.e., expensive data annotation

EDA Tool Parameter Tuning

- EDA tools provide effective and complex optimization options
- Efficient Tool Parameter Tuning
  - XGBoost [10]
  - Neural Networks (NN) [5]
  - Gaussian process (GP) [3]
- These approaches typically view tool parameter tuning as a regression task!

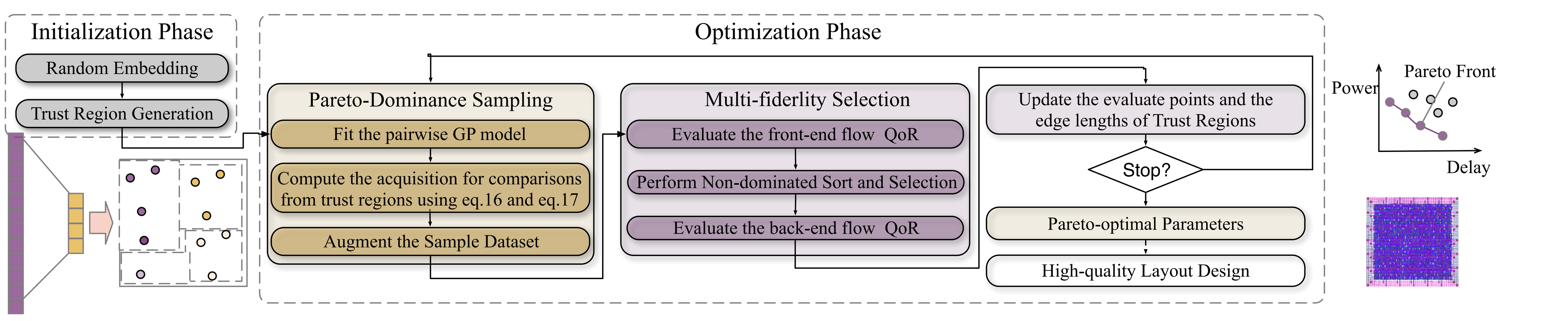
Motivation

- Existing methods focus on predicting the **exact** QoR values
  - The enormous options make it difficult to train an accurate model [3]
  - A lack of uncertainty modeling leads to inaccurate Pareto relationship [9]
- What do we need? Ranking-based tuning framework!
  - Preference Bayesian Optimization → Pairwise GP + Duel-Thompson Sampling

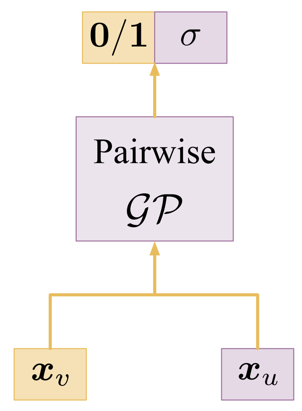


The Overall Flow of Our RankTuner Framework

1. Random Embedding Generation
2. Trust-region Initialization
3. Informative Comparison Selection between Regions
4. Multi-fidelity Evaluation and Update



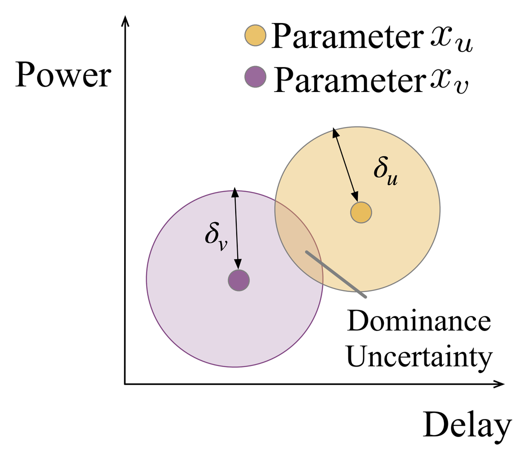
The Pairwise Gaussian Process



A pairwise likelihood function is defined as:

$$p_{\text{ideal}}(\vec{x}_v \succeq \vec{x}_u | f(\vec{x}_v), f(\vec{x}_u)) = \begin{cases} 1 & \text{if } f(\vec{x}_v) \geq f(\vec{x}_u) \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The Dominating Uncertainty Region

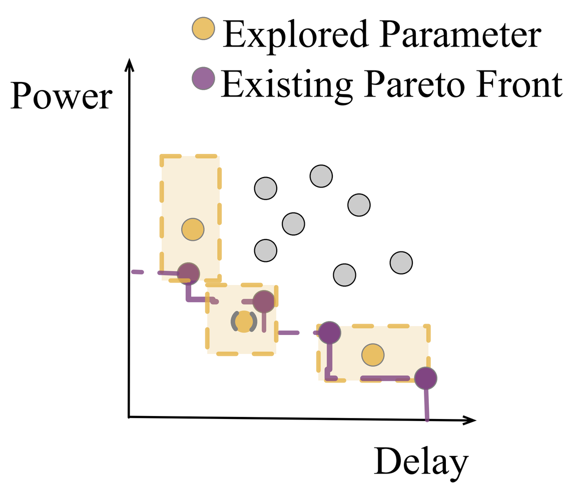


Using a Gaussian noise to model the dominance uncertainty, the pairwise likelihood function could be formulated as:

$$\begin{aligned} \Phi(z_k) &= p\left(\vec{x}_v \succeq \vec{x}_u \mid f(\vec{x}_v), f(\vec{x}_u)\right), \\ &= \iint p_{\text{ideal}}\left(\vec{x}_v \succeq \vec{x}_k \mid f(\vec{x}_v)+\delta_v, f(\vec{x}_u)+\delta_u\right) \\ &\quad \mathcal{N}\left(\delta_v; 0, \sigma^2\right) \mathcal{N}\left(\delta_u; 0, \sigma^2\right) \mathrm{d}\delta_v \mathrm{d}\delta_u, \end{aligned} \quad (2)$$

where $z_k = \frac{f(\vec{x}_{\vec{v}})-f(\vec{x}_{\vec{u}})}{\sqrt{2}\sigma}$ and $\Phi(z) = \int_{-\infty}^{z} N(\gamma; 0, 1)\mathrm{d}\gamma$.

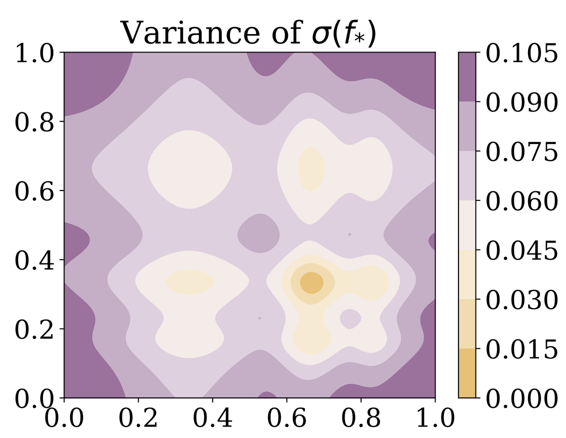
Acquisition Function for Pareto-dominance Comparison



Exploration and Exploitation of Comparisons

- Searching across the entire search space of parameter tuning requires an effective balance between **exploration and exploitation**
- The key aspect is to select informative parameter pairs for comparison

Pareto-Dominace Thompson Sampling



1. **Selecting** $\vec{x}$ The first element of the new comparison, $\vec{x}_{\text{next}}$, is selected as:

$$\vec{x}_{\text{next}} = \arg\max_{\vec{x} \in \mathcal{X}} \int_{\mathcal{X}} \pi_{\tilde{f}}\left(\left[\vec{x}, \vec{x}'\right]\right) \mathrm{d}\vec{x}'. \quad (3)$$

2. **Selecting** $\vec{x}'$: The second element is selected as the parameter configuration that maximizes the variance of $\sigma(f_\star)$ in the direction of $\vec{x}_{\text{next}}$,

$$\vec{x}'_{\text{next}} = \arg\max_{\vec{x}'_\star \in \mathcal{X}} \mathbb{V}\left[\sigma(f_\star) \mid \left[\vec{x}_\star, \vec{x}'_\star\right], \vec{x}_\star = \vec{x}_{\text{next}}\right]. \quad (4)$$

Experimental Setup

- Benchmarks: RISC-V processors (***RISCV32I*** [8] and ***Rocket*** [1]), and ***BlackParrot*** [7] processors (***BP***).
- The QoR-related metrics are used to compare the parameter tuning methods as in [12]:
  - Hypervolume (HV)
  - Maximum performance improvement (MPI1), Maximum power improvement (MPI2), Maximum area improvement (MAI).
  - Maximum performance-power improvement (MPPI), and Maximum performance-area improvement (MPAI)

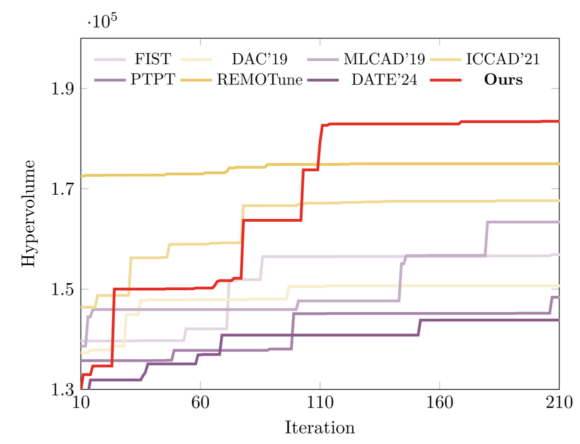
Comparison Between Ours and Previous Methods

Table 1. Comparison of Parameter Tuning Methods on ***RISCV32I*** Benchmark.

Method	FIST	DAC'19	MLCAD'19	ICCAD'21	PTPT	REMOTune	DATE'24	**Ours**
HV (10^5)	1.57	1.55	1.63	1.68	1.48	1.75	1.44	**1.84**
$\mathrm{HV}_{0,1}$ (10^3)	2.85	2.72	3.00	2.95	2.70	3.05	2.63	**3.44**
$\mathrm{HV}_{0,2}$ (10^3)	2.94	2.99	3.00	3.07	2.95	3.12	2.84	**3.43**
$\mathrm{HV}_{1,2}$ (10^3)	2.97	2.97	3.00	3.14	2.79	**3.23**	2.77	3.00
MPI1(%)	3.16	2.54	5.00	3.81	3.56	4.38	2.08	**13.64**
MPI2(%)	3.90	2.12	5.12	5.23	0.85	**6.27**	0.68	5.04
MAI(%)	5.47	7.18	4.64	7.10	5.15	**7.45**	4.74	5.12
MPPI(%)	6.94	4.51	9.88	8.83	4.37	10.38	1.30	**13.73**
MPAI (%)	8.46	9.53	9.41	10.63	8.52	11.53	5.43	**12.26**

- RankTuner consistently outperforms them across all benchmarks up to 40.34% improvement of hypervolume.
- RankTuner acquires 4.89% and 3.59% higher hypervolumes than the best baseline method, REMOTuner [12], on ***RISCV32I*** and ***Rocket*** benchmarks.

The Attained Hypervolume *v.s.* Iteration

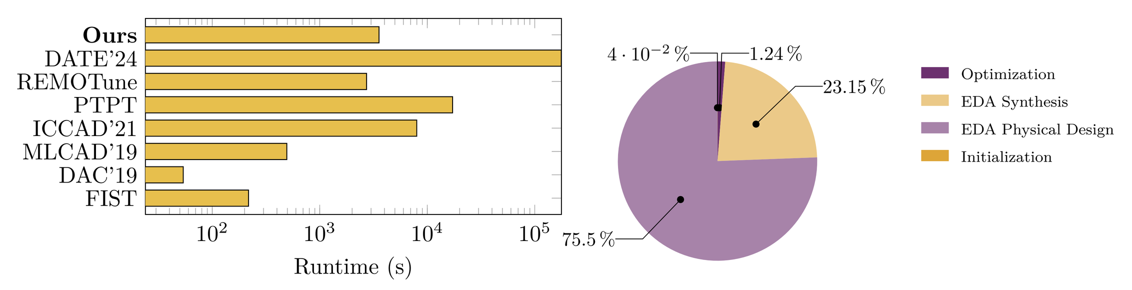


The RankTuner framework also offers a notable advantage in constantly improving the explored Pareto front:

- The RankTuner framework offers a notable advantage in constantly improving the explored Pareto front.
- Although RankTuner has nearly the lowest initial HV value, it continuously improves during the exploration process and eventually surpasses all other methods at around 100 iterations.

The Runtime Comparison & Breakdown

- RankTuner is nearly 4.83× faster than PTPT [3] due to the parallel exploration
- The most consuming part is the EDA Physical Design part, which takes 75.5% of the total runtime. The initialization and optimization time only take about 1.25% in total.



References

[1] Krste Asanovic, Rimas Avizienis, Jonathan Bachrach, Scott Beamer, David Biancolin, Christopher Celio, Henry Cook, Daniel Dabbelt, John Hauser, Adam Izraelevitz, et al. The rocket chip generator. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17*, 4, 2016.

[2] Luo Donger, Sun Qi, Xu Qi, Chen Tinghuan, and Geng Hao. Attention-based eda tool parameter explorer: From hybrid parameters to multi-qor metrics. 2024.

[3] Hao Geng, Tinghuan Chen, Yuzhe Ma, Binwu Zhu, and Bei Yu. Ptpt: physical design tool parameter tuning via multi-objective bayesian optimization. *IEEE TCAD*, 42(1):178–189, 2022.

[4] Jinwook Jung, Andrew B. Kahng, Seungwon Kim, and Ravi Varadarajan. METRICS2.1 and flow tuning in the IEEE CEDA robust design flow and OpenROAD. In *Proc. ICCAD*, 2021.

[5] Jihye Kwon, Matthew M. Ziegler, and Luca P. Carloni. A learning-based recommender system for autotuning design flows of industrial high-performance processors. In *Proc. DAC*, 2019.

[6] Yuzhe Ma, Ziyang Yu, and Bei Yu. CAD tool design space exploration via bayesian optimization. In *Proc. MLCAD*, 2019.

[7] Daniel Petrisko, Farzam Gilani, Mark Wyse, Dai Cheol Jung, Scott Davidson, Paul Gao, Chun Zhao, Zahra Azad, Sadullah Canakci, Bandhav Veluri, Tavio Guarino, Ajay Joshi, Mark Oskin, and Michael Bedford Taylor. BlackParrot: An agile open-source RISC-V multicore for accelerator SoCs. *Proc. MICRO*, 40(4):93–102, 2020.

[8] James E. Stine, Ryan Ridley, and Teodor-Dumitru Ene. Osu datapath/control rv32 single-cycle and pipelined architecture in sv, 2021.

[9] Qi Sun, Tinghuan Chen, Siting Liu, Jianli Chen, Hao Yu, and Bei Yu. Correlated multi-objective multi-fidelity optimization for hls directives design. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 27(4):1–27, 2022.

[10] E. Ustun, S. Xiang, J. Gui, C. Yu, and Z. Zhang. LAMDA: Learning-assisted multi-stage autotuning for FPGA design closure. In *Proc. FCCM*, pages 74–77, 2019.

[11] Zhiyao Xie, Guan-Qi Fang, Yu-Hung Huang, Haoxing Ren, Yanqing Zhang, Brucek Khailany, Shao-Yun Fang, Jiang Hu, Yiran Chen, and Erick Carvajal Barboza. FIST: A feature-importance sampling and tree-based method for automatic design flow parameter tuning. In *Proc. ASPDAC*, 2020.