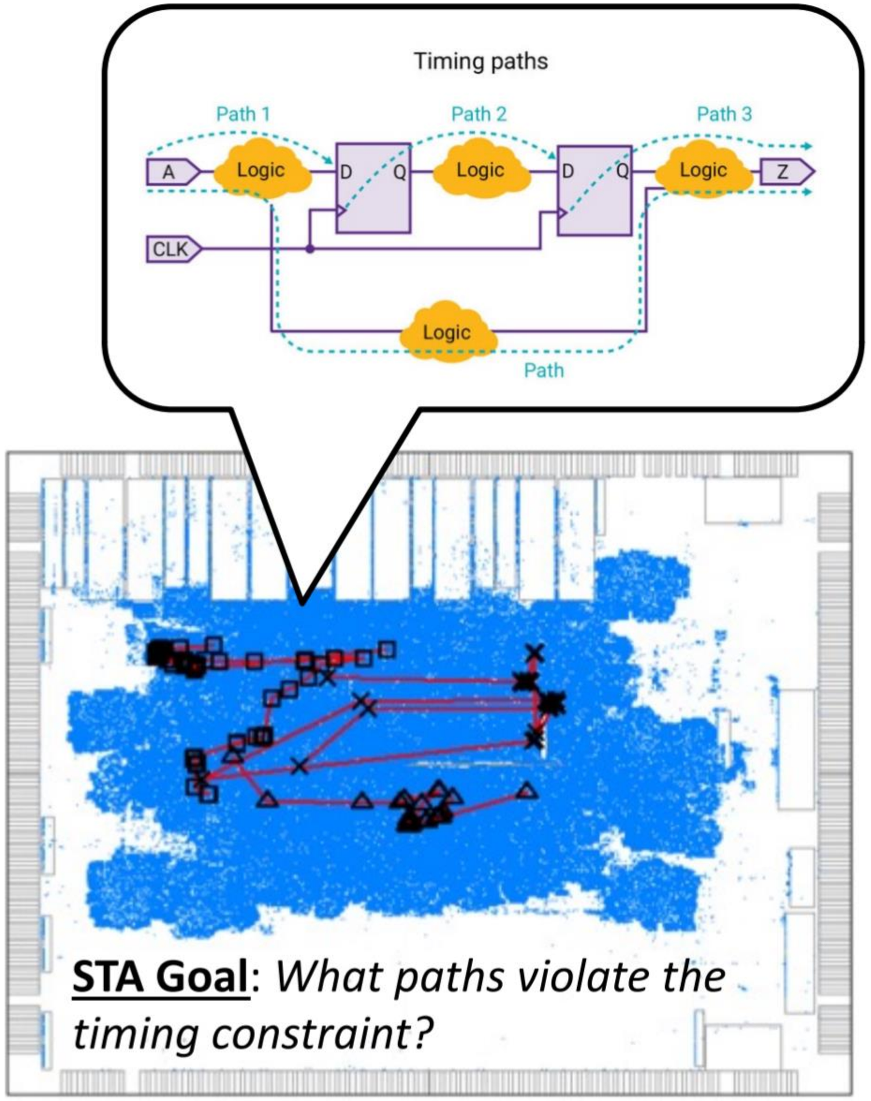
General Heterogeneous Framework for Path-based Timing Analysis

Yasin Zamani, PhD Student
Department of Electrical & Computer Engineering
University of Utah, Salt Lake City, UT
Supervisor: Dr. Tsung-Wei Huang

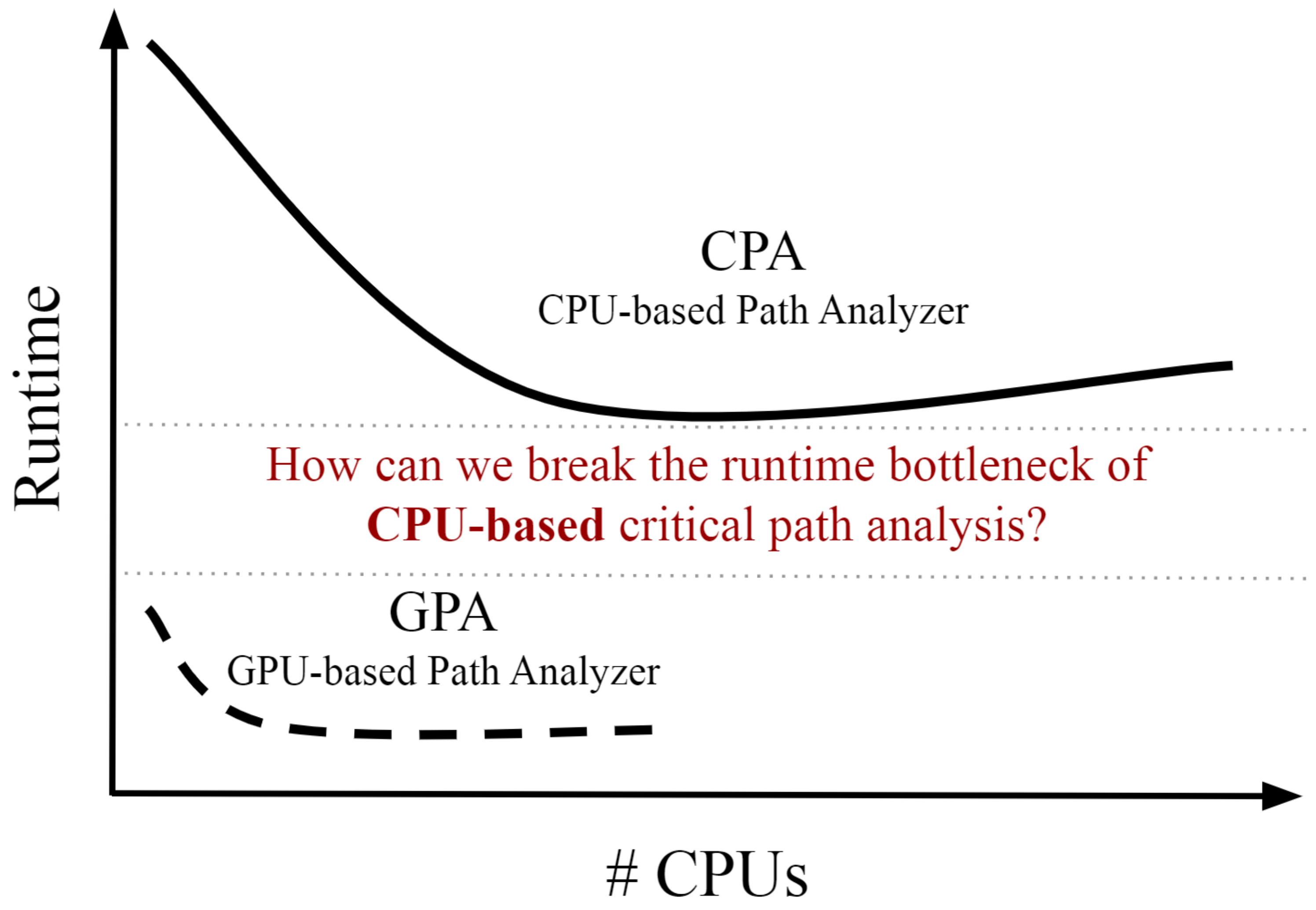
Path-based Timing Analysis

- Static timing analysis (**STA**) computes the propagation of time signals in a circuit from its primary inputs to its primary outputs through various circuit elements and interconnect.
- Graph-based analysis (**GBA**) is a step of STA used to perform a worst-case (i.e., early and late) analysis of a circuit over all possible paths to update the graph with timing information.
- Path-based analysis (**PBA**) is another step of STA that is typically applied after GBA to reanalyze timing with path-specific updates, such as *common path pessimism removal (CPPR)*.



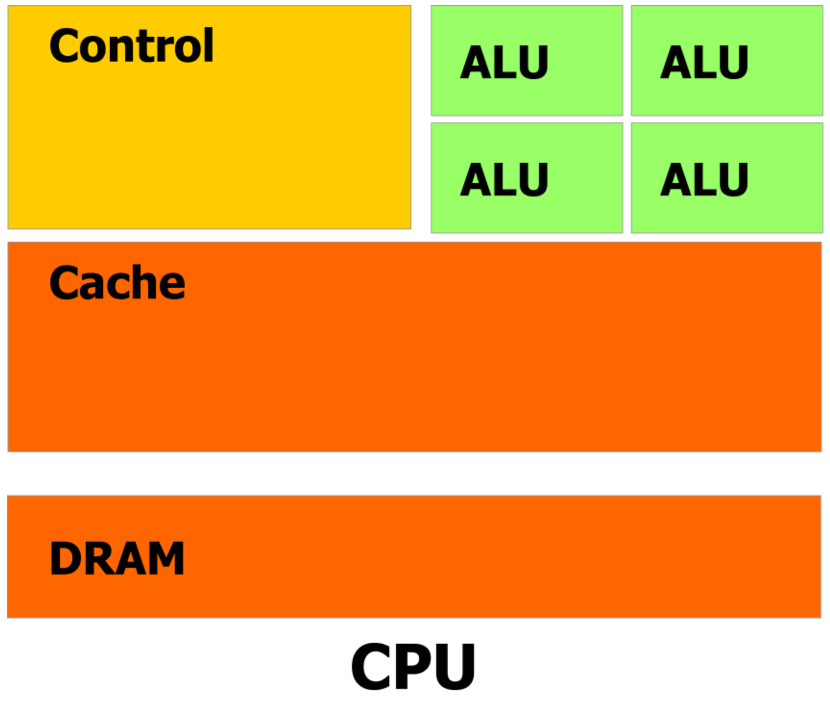
STA Goal: *What paths violate the timing constraint?*

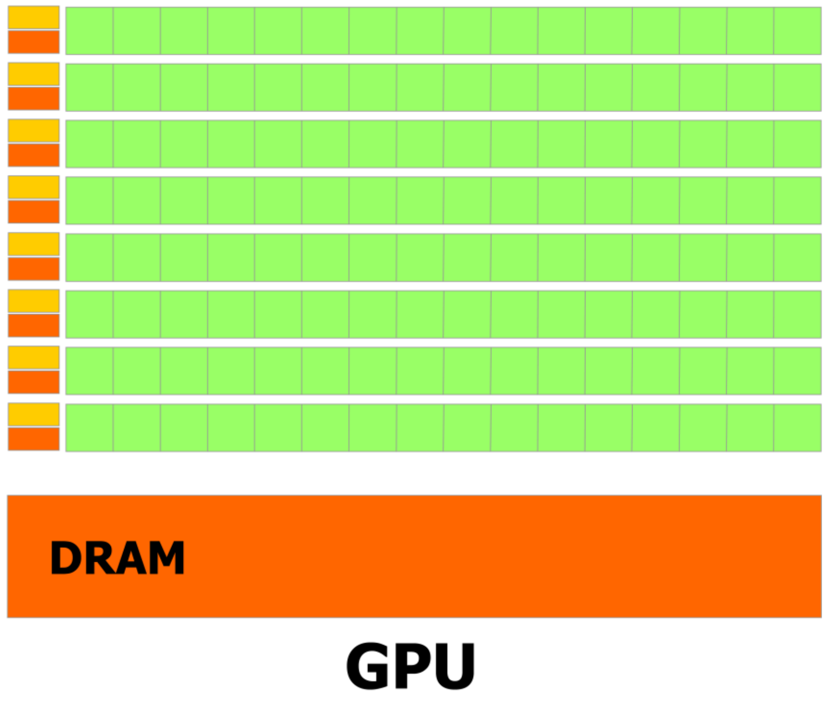
Runtime Challenges of PBA



CPU vs GPU

- CPU is built for **compute-driven** applications
  - A few powerful threads to compute critical blocks fast
- GPU is built for **throughput-driven** applications
  - Many lightweight threads to compute data at one time



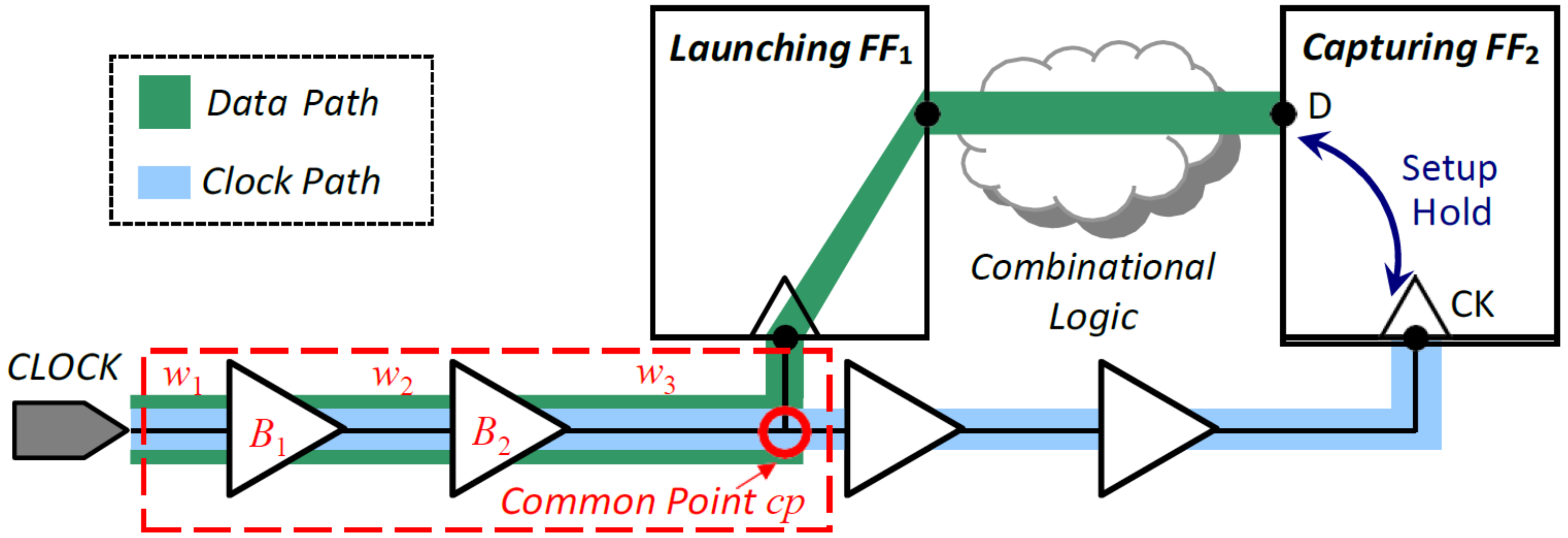


Problem Formulation

- **Hold Test**
  - Data must be stable **after** clock signal arrives at the capturing flip-flop
- **Setup Test**
  - Data must be stable **before** clock signal arrives at the capturing flip-flop

➢ **Input**
  ➢ Set of **critical paths** generated after GBA

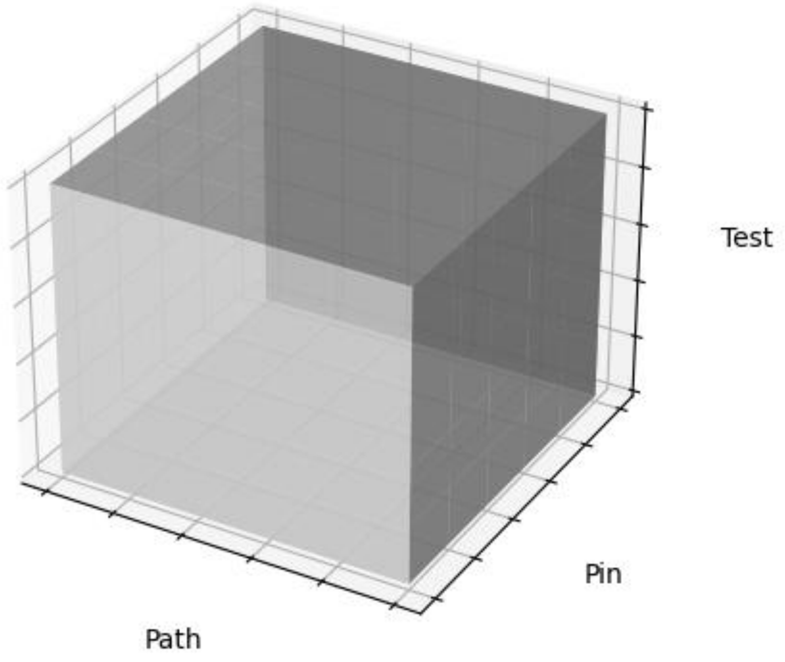
➢ **Output**
  ➢ Sorted critical paths after path-specific update (here we apply to **CPPR**)



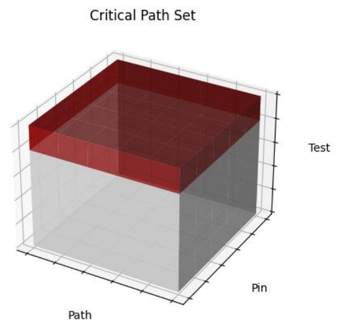
Parallel Decomposition

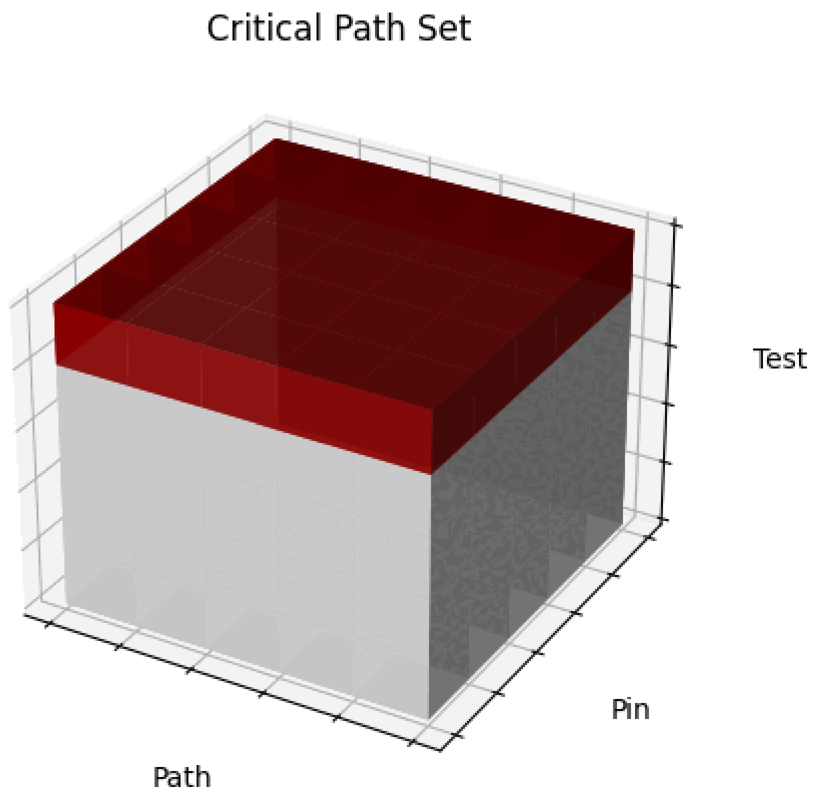
The set of critical paths is represented in terms of **timing-test**, **critical path**, and **pin** of path traces

Critical Path Set

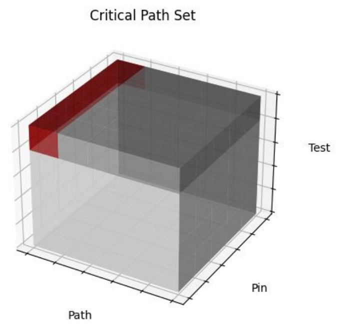


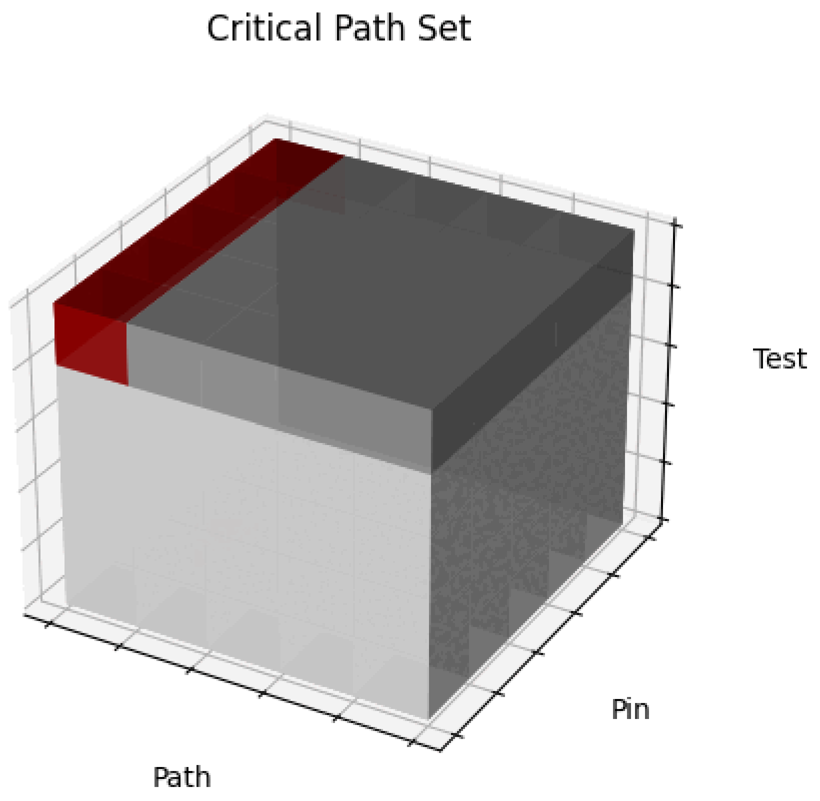
Parallel Decomposition in Term of Tests



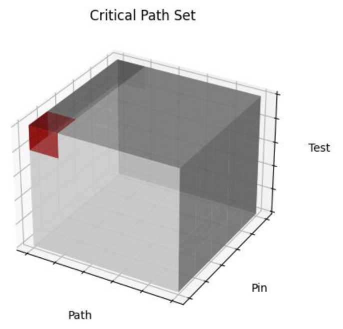


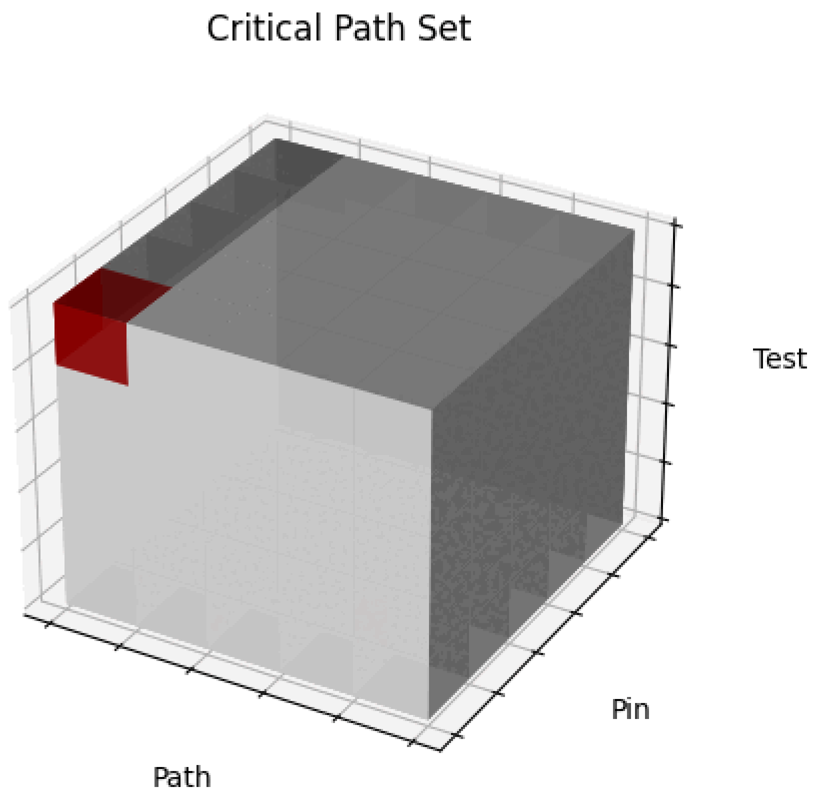
Parallel Decomposition in Term of Paths





Parallel Decomposition in Term of Pins





Challenges

- Computing critical paths involve very **irregular patterns** because different paths can have different lengths across different timing tests (e.g., hold/setup checks).
- To support a **large number of paths**, we need efficient data structures to fit computations into relatively limited GPU memory.
- **GPU architectures** are very different compared to CPUs, in terms of thread scheduling, synchronization, and

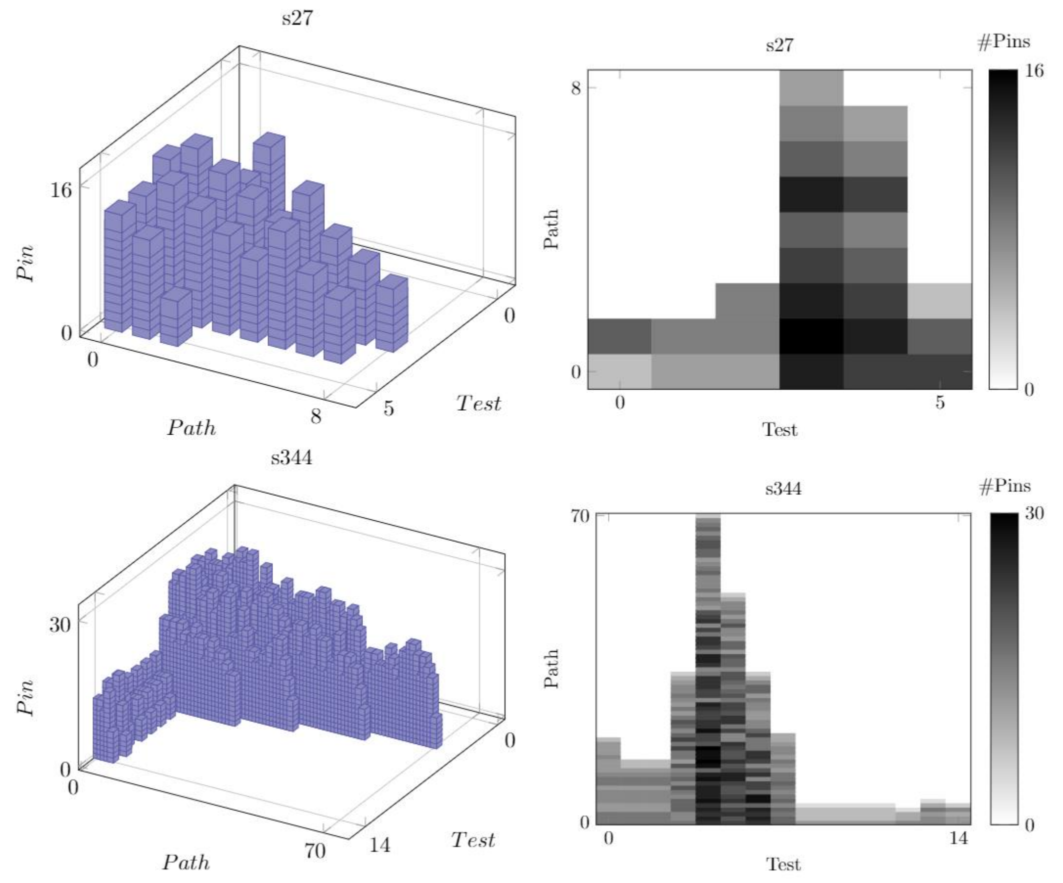


Fig. 5: Uneven distribution of paths among tests causes irregular parallelism on GPU. s27 and s344 are benchmarks from from TAU 2014 CAD Contest [6]

Example

□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□

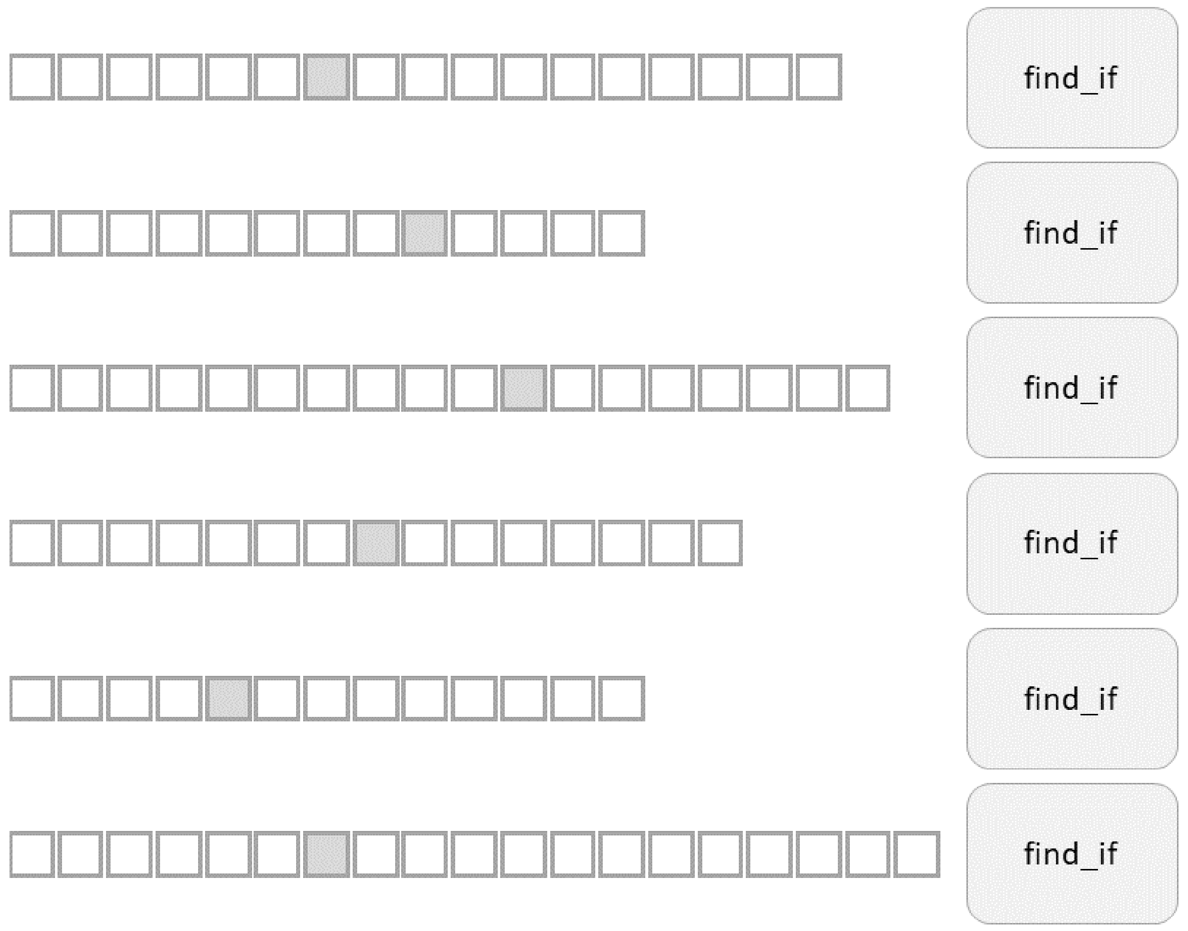
□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□

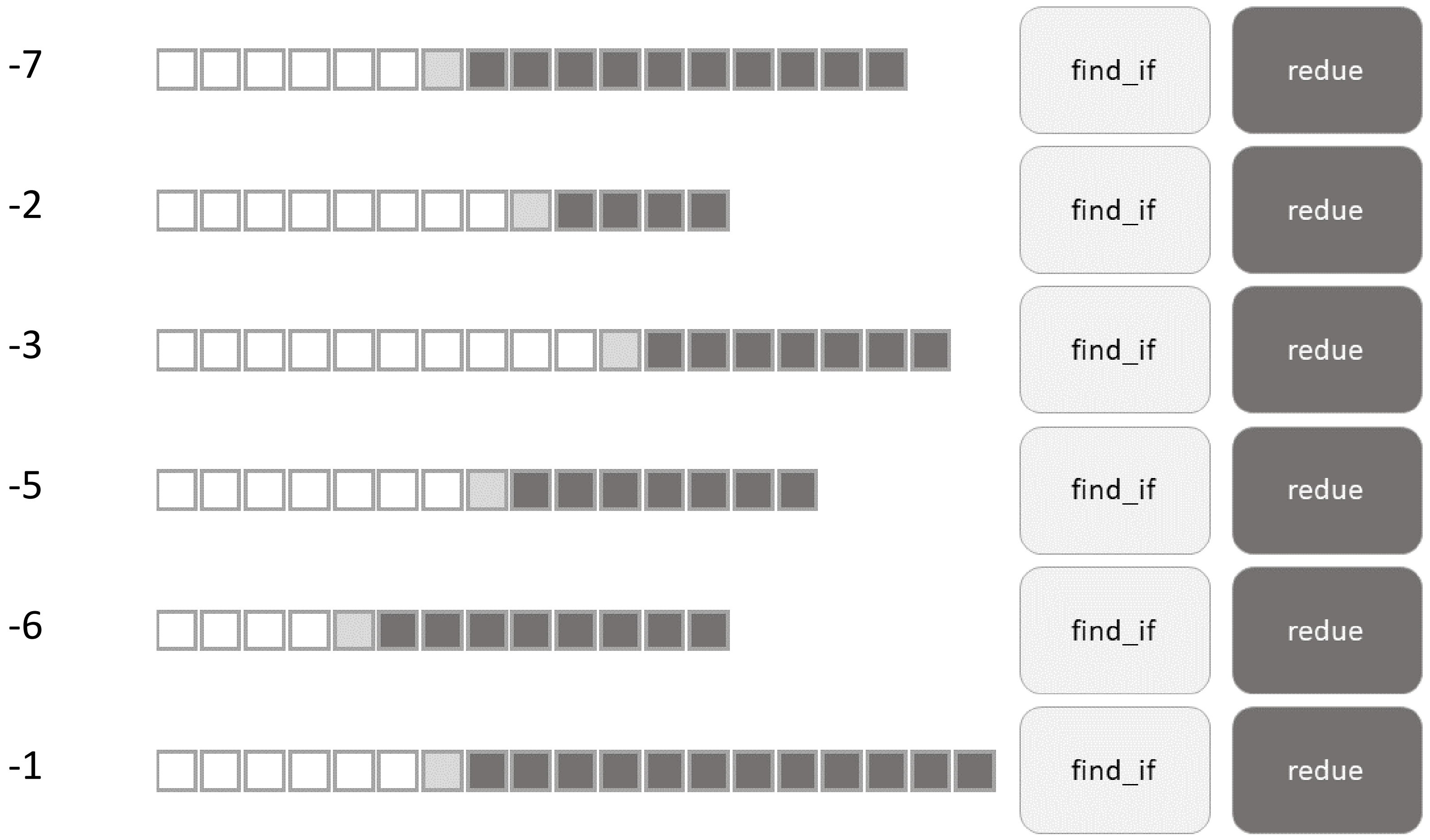
□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□

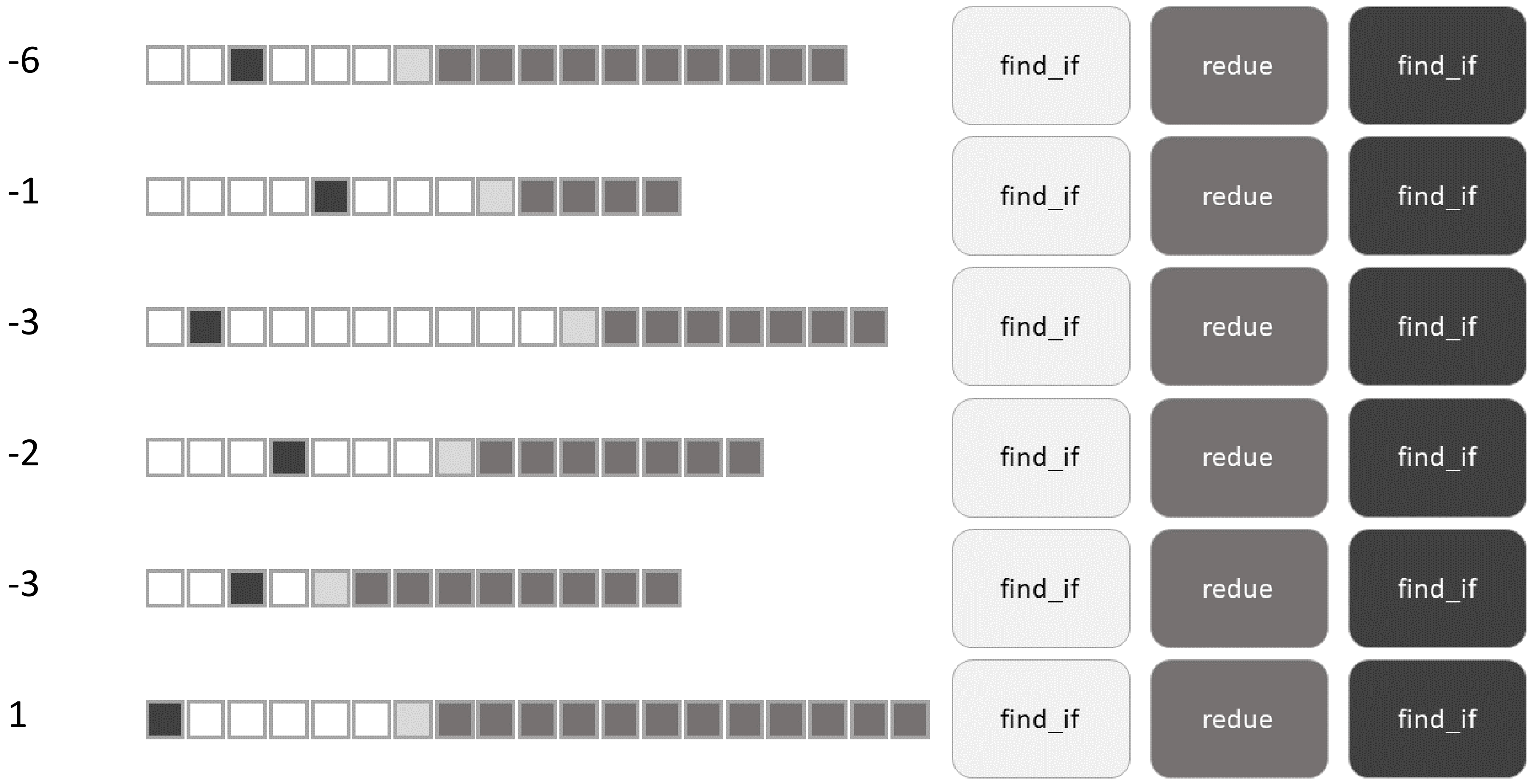
Example …



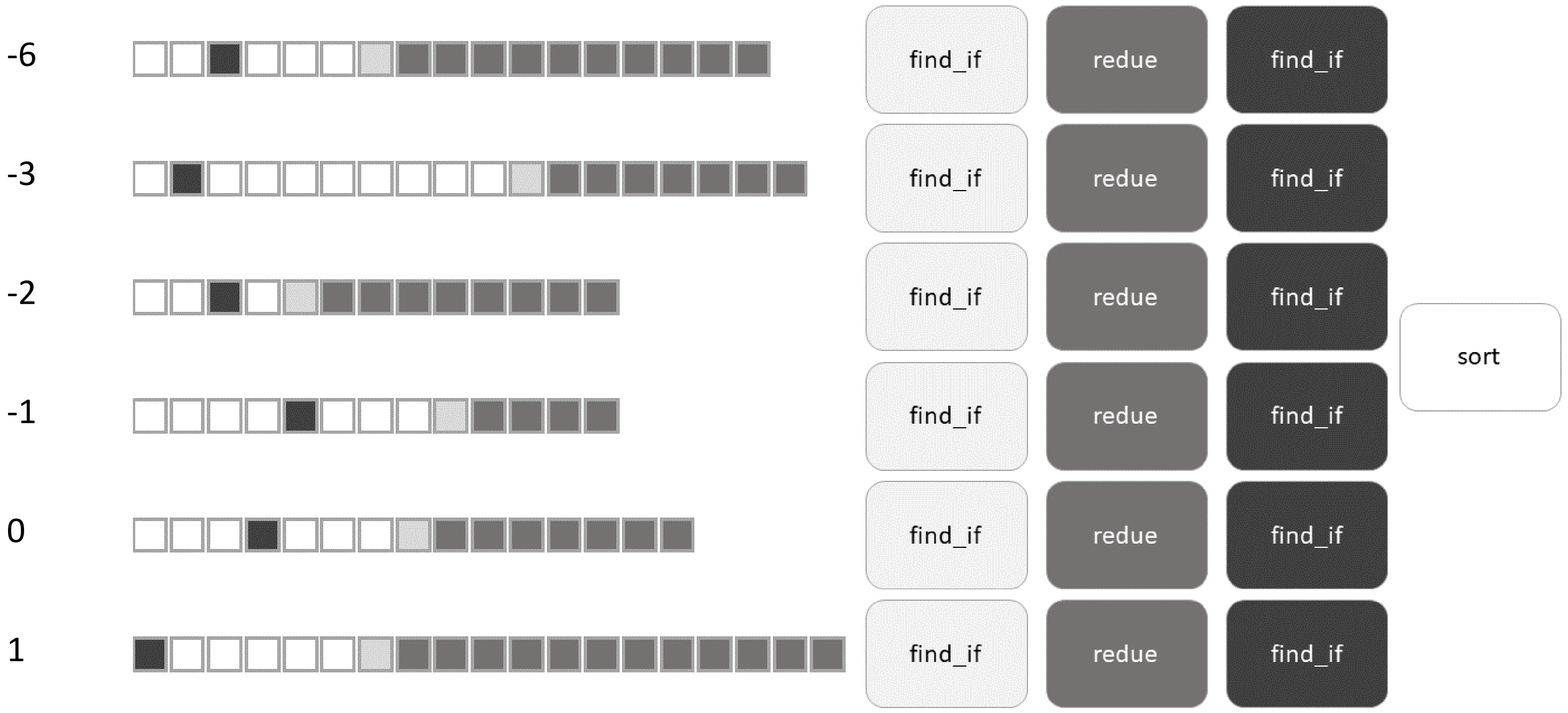
Example …



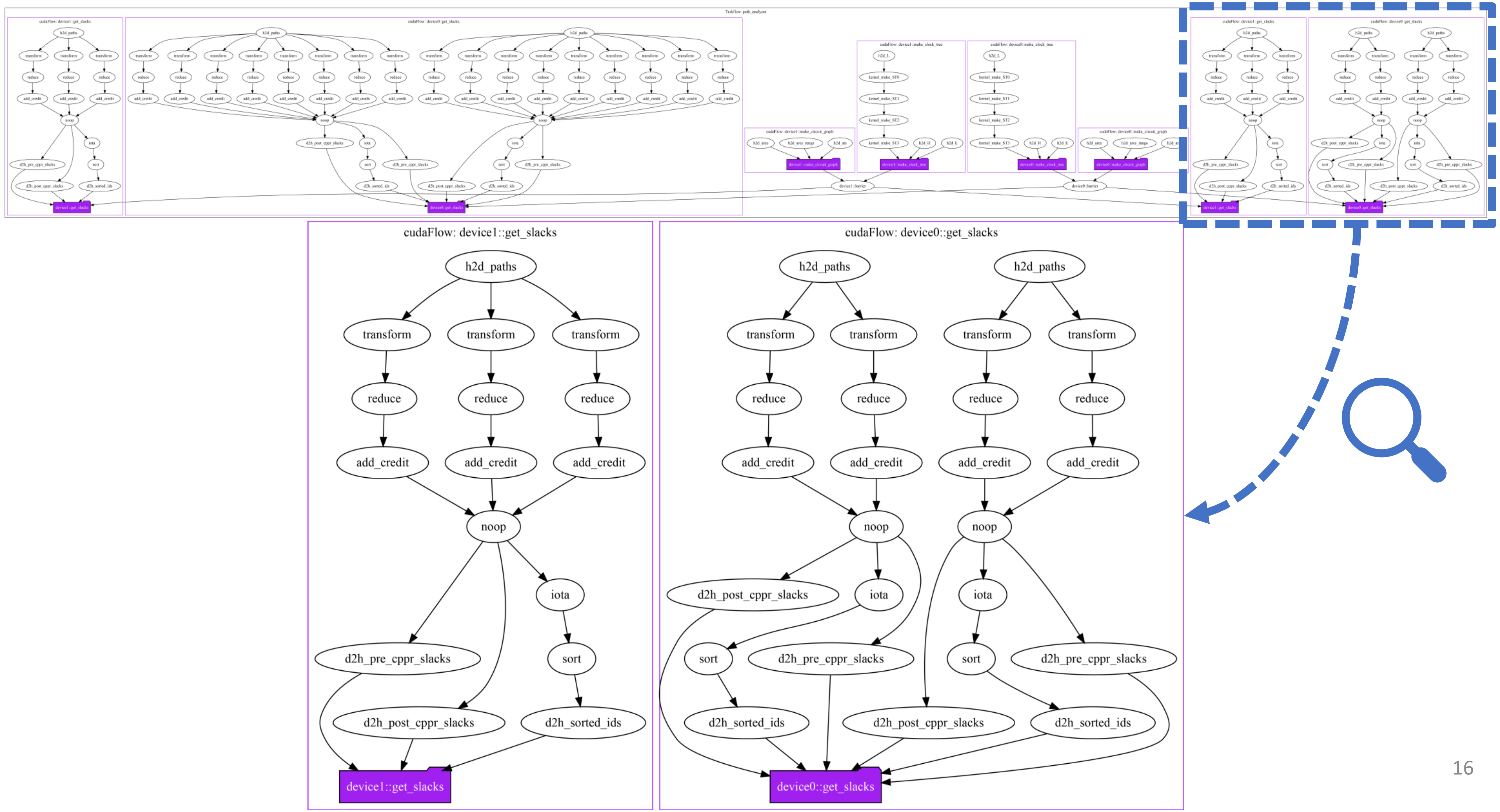
Example



Example …



Task Graph-based Decomposition and Efficient GPU Kernels



Taskflow: path_analyzer
cudaFlow: device1::get_slacks
h2d_paths
transform
reduce
add_credit
noop
iota
sort
d2h_pre_cppr_slacks
d2h_post_cppr_slacks
d2h_sorted_ids
device1::get_slacks
cudaFlow: device0::get_slacks
device0::get_slacks
cudaFlow: device1::make_circuit_graph
device1::make_circuit_graph
cudaFlow: device1::make_clock_tree
device1::make_clock_tree
cudaFlow: device0::make_clock_tree
device0::make_clock_tree
cudaFlow: device0::make_circuit_graph
device0::make_circuit_graph
device1::barrier
device0::barrier

Results

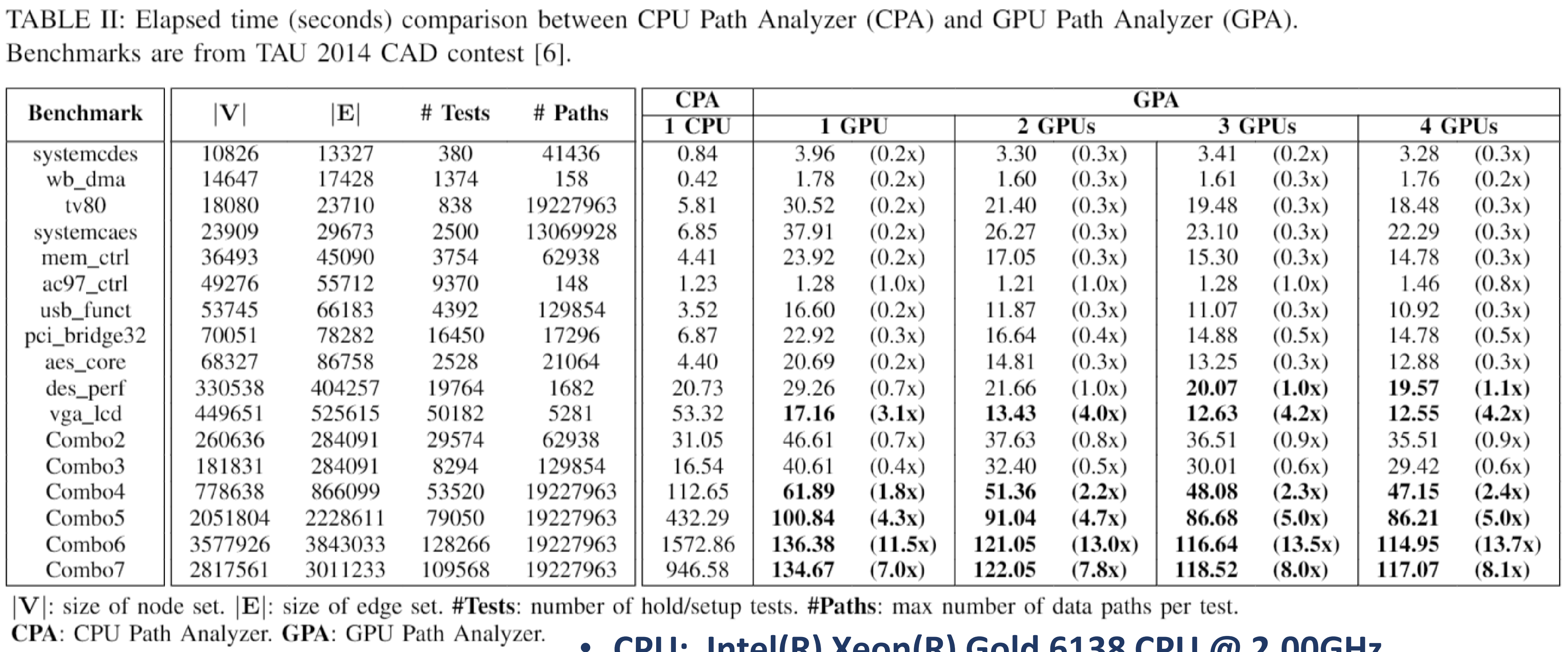


TABLE II: Elapsed time (seconds) comparison between CPU Path Analyzer (CPA) and GPU Path Analyzer (GPA). Benchmarks are from TAU 2014 CAD contest [6].

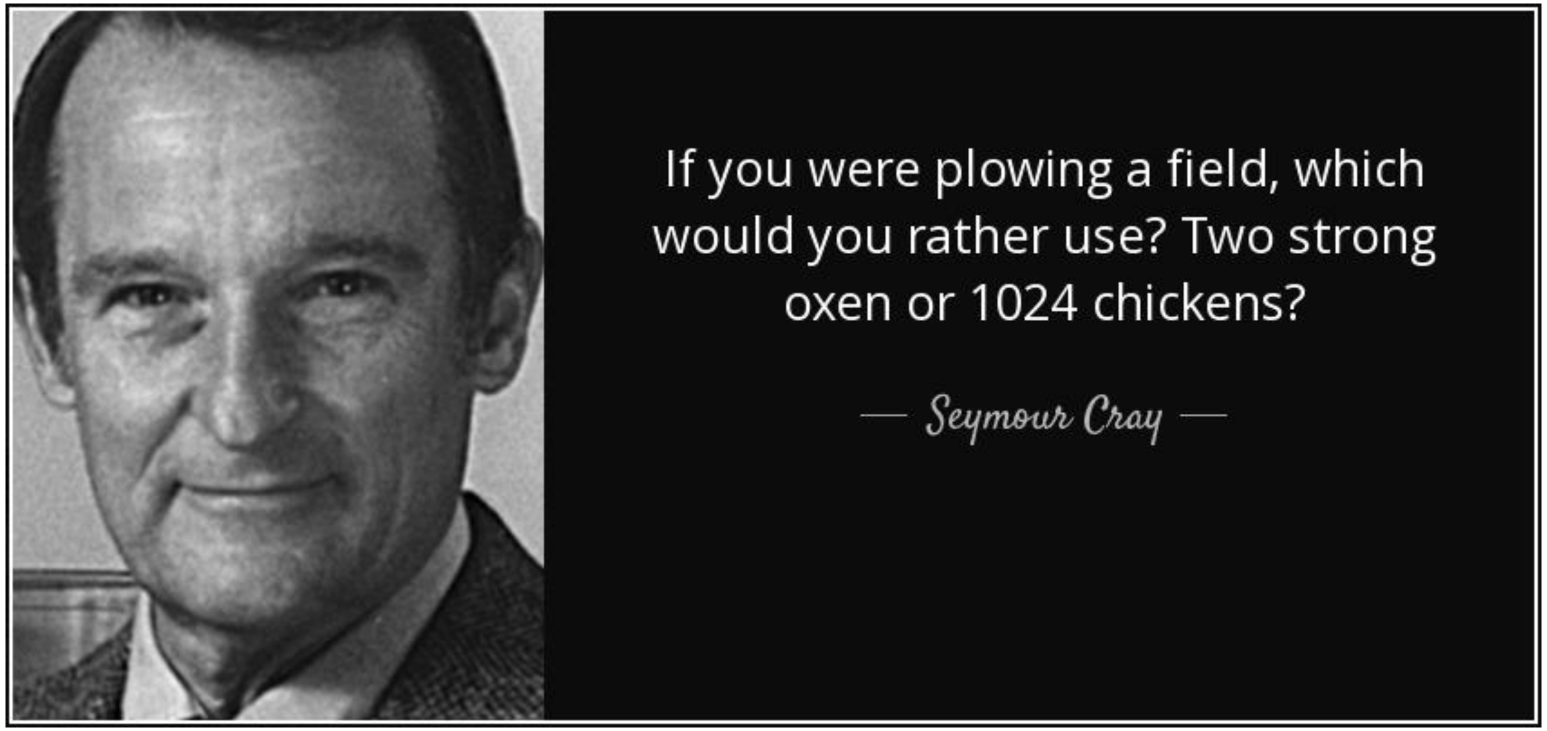
Benchmark	|V|	|E|	# Tests	# Paths	CPA	GPA							
					1 CPU	1 GPU		2 GPUs		3 GPUs		4 GPUs	
systemcdes	10826	13327	380	41436	0.84	3.96	(0.2x)	3.30	(0.3x)	3.41	(0.2x)	3.28	(0.3x)
wb_dma	14647	17428	1374	158	0.42	1.78	(0.2x)	1.60	(0.3x)	1.61	(0.3x)	1.76	(0.2x)
tv80	18080	23710	838	19227963	5.81	30.52	(0.2x)	21.40	(0.3x)	19.48	(0.3x)	18.48	(0.3x)
systemcaes	23909	29673	2500	13069928	6.85	37.91	(0.2x)	26.27	(0.3x)	23.10	(0.3x)	22.29	(0.3x)
mem_ctrl	36493	45090	3754	62938	4.41	23.92	(0.2x)	17.05	(0.3x)	15.30	(0.3x)	14.78	(0.3x)
ac97_ctrl	49276	55712	9370	148	1.23	1.28	(1.0x)	1.21	(1.0x)	1.28	(1.0x)	1.46	(0.8x)
usb_funct	53745	66183	4392	129854	3.52	16.60	(0.2x)	11.87	(0.3x)	11.07	(0.3x)	10.92	(0.3x)
pci_bridge32	70051	78282	16450	17296	6.87	22.92	(0.3x)	16.64	(0.4x)	14.88	(0.5x)	14.78	(0.5x)
aes_core	68327	86758	2528	21064	4.40	20.69	(0.2x)	14.81	(0.3x)	13.25	(0.3x)	12.88	(0.3x)
des_perf	330538	404257	19764	1682	20.73	29.26	(0.7x)	21.66	(1.0x)	**20.07**	**(1.0x)**	**19.57**	**(1.1x)**
vga_lcd	449651	525615	50182	5281	53.32	**17.16**	**(3.1x)**	**13.43**	**(4.0x)**	**12.63**	**(4.2x)**	**12.55**	**(4.2x)**
Combo2	260636	284091	29574	62938	31.05	46.61	(0.7x)	37.63	(0.8x)	36.51	(0.9x)	35.51	(0.9x)
Combo3	181831	284091	8294	129854	16.54	40.61	(0.4x)	32.40	(0.5x)	30.01	(0.6x)	29.42	(0.6x)
Combo4	778638	866099	53520	19227963	112.65	**61.89**	**(1.8x)**	**51.36**	**(2.2x)**	**48.08**	**(2.3x)**	**47.15**	**(2.4x)**
Combo5	2051804	2228611	79050	19227963	432.29	**100.84**	**(4.3x)**	**91.04**	**(4.7x)**	**86.68**	**(5.0x)**	**86.21**	**(5.0x)**
Combo6	3577926	3843033	128266	19227963	1572.86	**136.38**	**(11.5x)**	**121.05**	**(13.0x)**	**116.64**	**(13.5x)**	**114.95**	**(13.7x)**
Combo7	2817561	3011233	109568	19227963	946.58	**134.67**	**(7.0x)**	**122.05**	**(7.8x)**	**118.52**	**(8.0x)**	**117.07**	**(8.1x)**

|V|: size of node set. |E|: size of edge set. **#Tests**: number of hold/setup tests. **#Paths**: max number of data paths per test.
CPA: CPU Path Analyzer. **GPA**: GPU Path Analyzer.

- **CPU: Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz**
- **GPU: NVIDIA GeForce RTX 2080 Ti (Compute Capability 7.5)**

On-Going Work

- Our future work plans to improve our data structures and algorithms to overcome the **unbalanced workload** and **limitation of GPU memory** challenges
- Apply the framework to other PBA applications
- Evaluate the PBA algorithm on large Nvidia benchmarks using **A100**
- We are planning to submit the preliminary work to **ASP-DAC 2022**



If you were plowing a field, which would you rather use? Two strong oxen or 1024 chickens?

— Seymour Cray —