A Systemic Approach to Maximize Heterogeneous System Performance

PhD Candidate: Thomas Randall

Dissertation Committee: Rong Ge (Chair), Prasanna Balaprakash, Xingfu Wu, Kai Liu, Feng Luo

Acknowledgements

"FULL-W2V: Fully Exploiting Data Reuse for W2V on GPU-Accelerated Systems" **BEST PAPER** in the Proceedings of the International Conference on Supercomputing 2021; Thomas Randall, Tyler Allen, and Rong Ge

"Transfer-learning-based Autotuning using Gaussian Copula" in the Proceedings of the International Conference on Supercomputing 2023; Thomas Randall, Jaehoon Koo, Brice Videau, Michael Kruse, Xingfu Wu, Paul Hovland, Mary Hall, Rong Ge, Prasanna Balaprakash

"Copy Cat: Limitations of LLMs in Performance Predictions," Poster appearing in Department of Energy Cybersecurity and Technology Innovation 2024; Thomas Randall, Rong Ge and Prasanna Balprakash

"Thermal Behaviors in Liquid Immersion Cooling under Various Workloads: a Case Study" in the Proceedings of the International Green and Sustainable Computing Conference 2024; Thomas Randall, Bennett Cooper, Naman Kulshreshtha, and Rong Ge

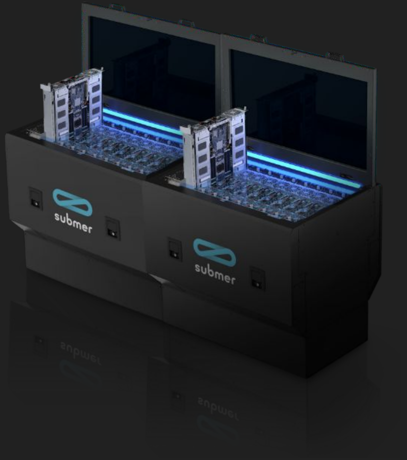
"Is In-Context Learning Feasible for HPC Performance Autotuning?" to appear in HPC for AI Foundation Models & LLMs for Science 2025 as part of IPDPS 2025; Thomas Randall, Akhilesh Bondapalli, Rong Ge, and Prasanna Balaprakash

Outline

- **Background and Overview**
- Systemic Approach
  - Hardware
  - Software
  - Tuning
- Concluding Discussion
- Q&A

Systemic Approach

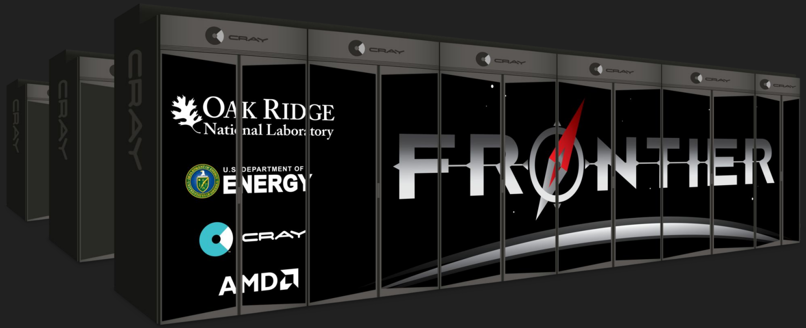
- New **hardware creates** new opportunities



- Ensure **software leverages** available performance
- **Tune their integration** for optimal results

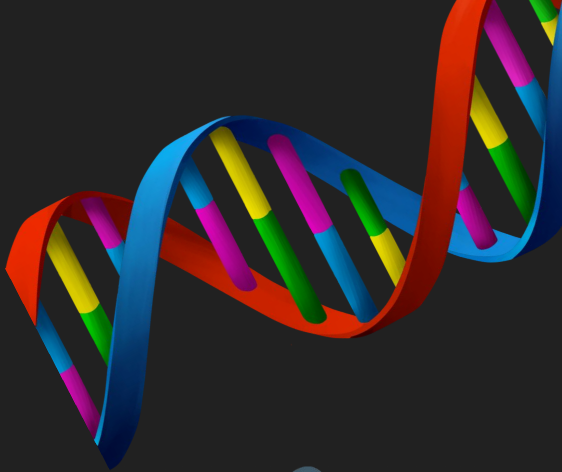
Target Environments

- Focus on High Performance Computing (HPC)
    - Cloud / Hyperscalers / Government



Why Do We Need It?

- Evolving landscape
  - From single core to GPU-accelerated
- Strong foundation
  - “Golden Age” of Microarchitecture
  - Mature practices
  - Years of expertise
- Future demand
  - New opportunities
  - Re-learning old tricks
  - Expert efforts don’t scale



Goals and Challenges

- Increase Hardware & Software synergy
  - Not always intuitive
  - Trade offs
- Prepare with flexibility in mind
  - Allow future improvement
- Minimal cost for maximum benefit
  - Near-infinite maximum cost

Core Components and Contributions

- Hardware
  - Increased cooling demand
  - **Understand SPLIC technology via benchmarks**
  - **First** study on applications
- Software
  - Underperforming hardware capability
  - **Redesign for acceleration**
  - **Extend** optimizations through hardware generations
- Tuning
  - “Glue” holds everything together
  - **Novel transfer learning technique**
  - **Predict & improve** low cost knowledge re-use

Significance and Impact

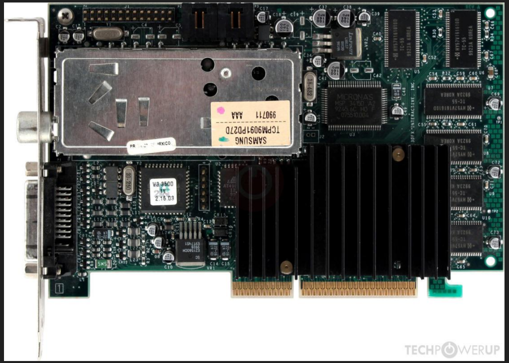
- Hardware
  - Identify real problems and opportunities
  - **Trade-off** thermal capacity / responsiveness
- Software
  - Reduce memory traffic **89%** vs SOTA
  - **5.72-6.51x** speedup across hardware generations
- Tuning
  - **12.81x** additional speedup over SOTA
  - **Predictable** success and failure conditions

Outline

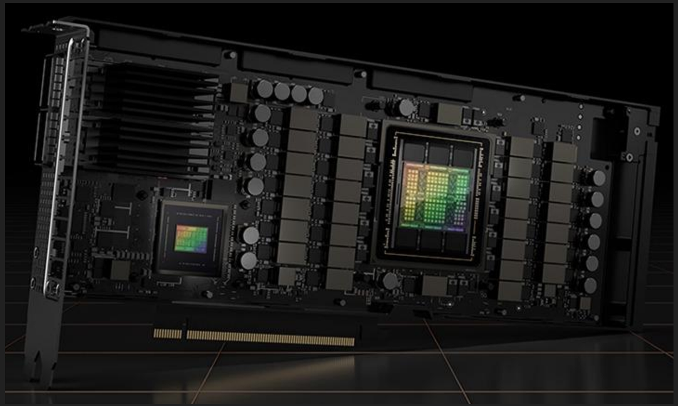
- Background and Overview
- Systemic Approach
    - **Hardware**
        - "Thermal Behaviors in Liquid Immersion Cooling under Various Workloads: a Case Study" in the Proceedings of the International Green and Sustainable Computing Conference 2024
    - Software
    - Tuning
- Concluding Discussion
- Q&A

Accelerators Demand

- Electric power → heat
    - Sacrifice performance or $
- 25 years → **KW**-scale TDPs!
- Air's capacity is exhausted
    - Need better cooling



Voodoo3 3500 TV AGP
1999, **15 Watts TDP**

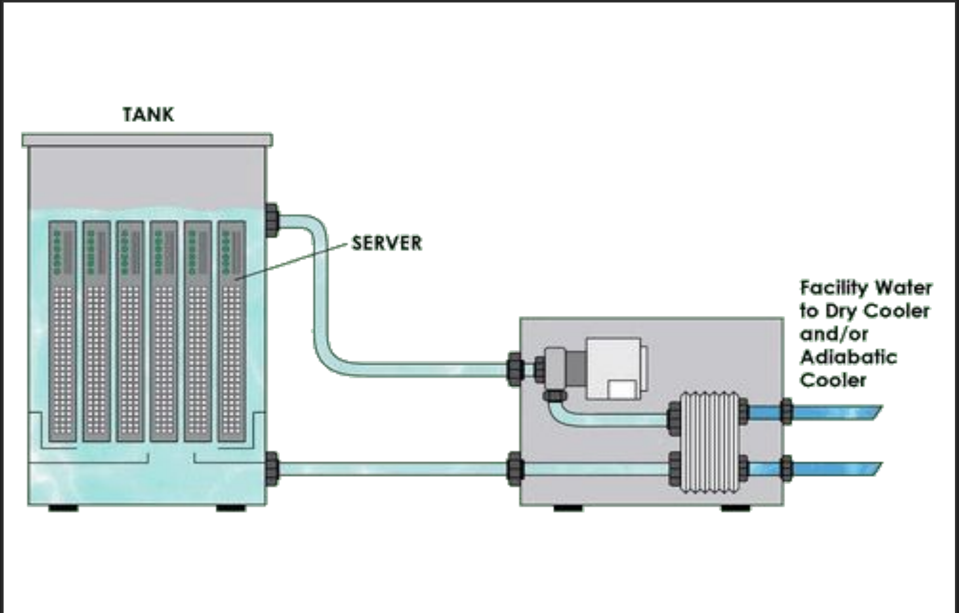


NVIDIA H100 GPU
2024, **350 Watts TDP**

Single Phase Liquid Immersion Cooling

- Direct contact with hardware
  - Dielectric
  - Efficient heat exchange
  - Recirculate coolant
    - No evaporation

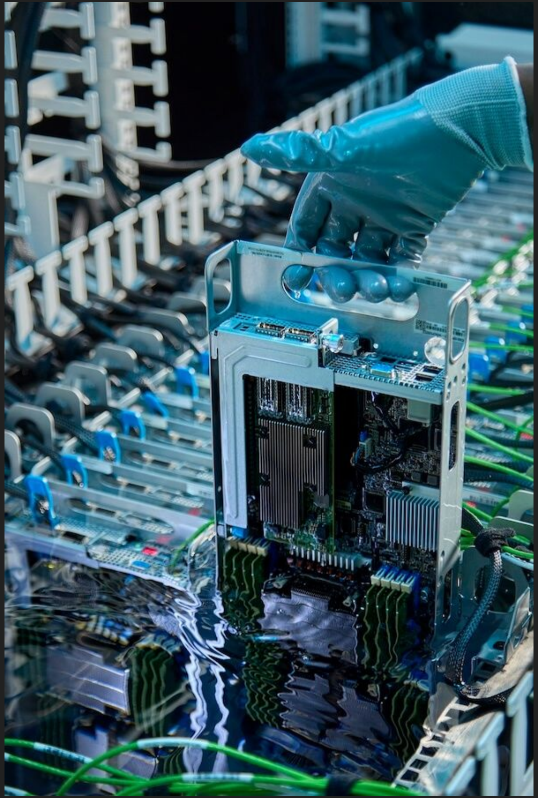
- Large industry focus



Previous Studies

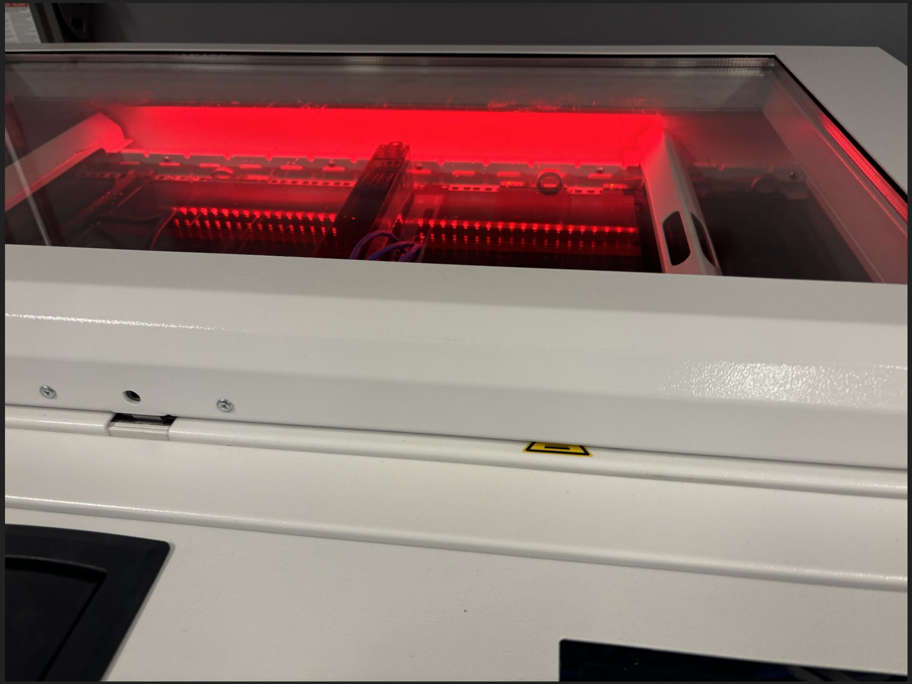
- Foundational effectiveness
- Design and properties of fluid
- Amount/arrangement of pumps
- Effects of long-term immersion on hardware
- Capital & Maintenance costs

- Lack understanding of:
  - Practical hotspots and behavior
  - Capacity vs Responsiveness
  - Range of operating conditions



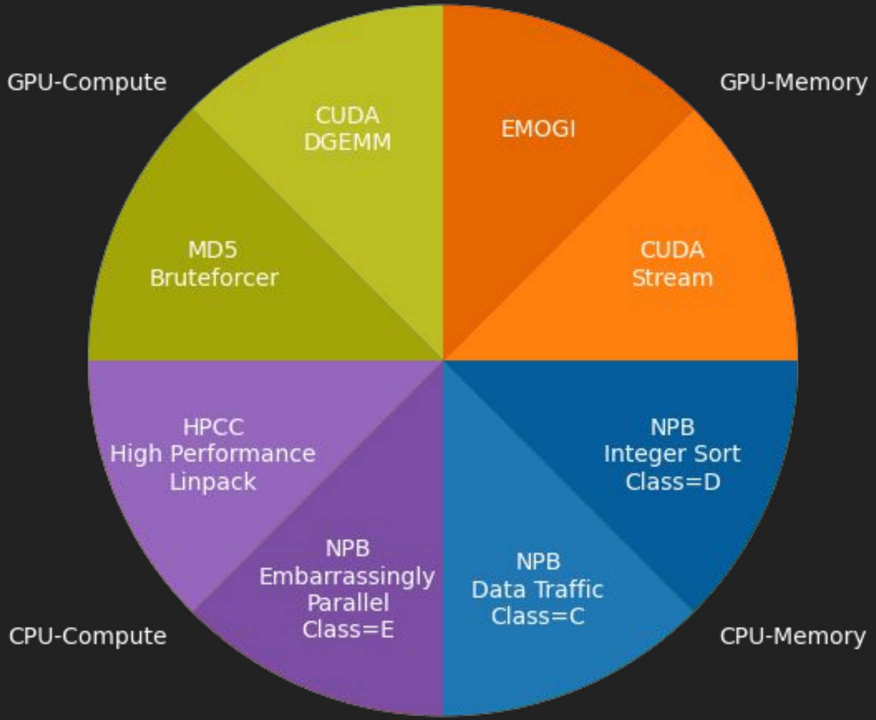
Empirical Study

- Submer SmartPod v3
  - White mineral oil
  - 11C facility water
- Traditional air-cooled rack
  - Similar hardware setup
  - 23C air
- Observing thermal behaviors
  - Under variety of HPC workload conditions
  - Heat exchange within the system, components



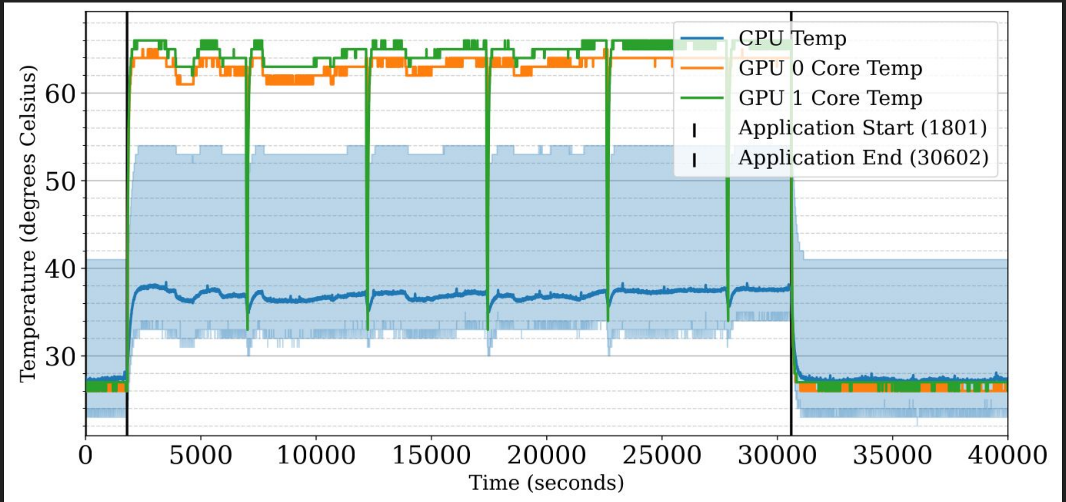
Metrics & Benchmarks

- Node-level
  - Temperatures
  - Overall power draw
- GPU Accelerators
  - Temperatures
  - Power usage
- SPLIC Tank
  - Coolant & water temperatures, flow rates
  - Pump RPM
- Estimate
  - Cooling-induced energy

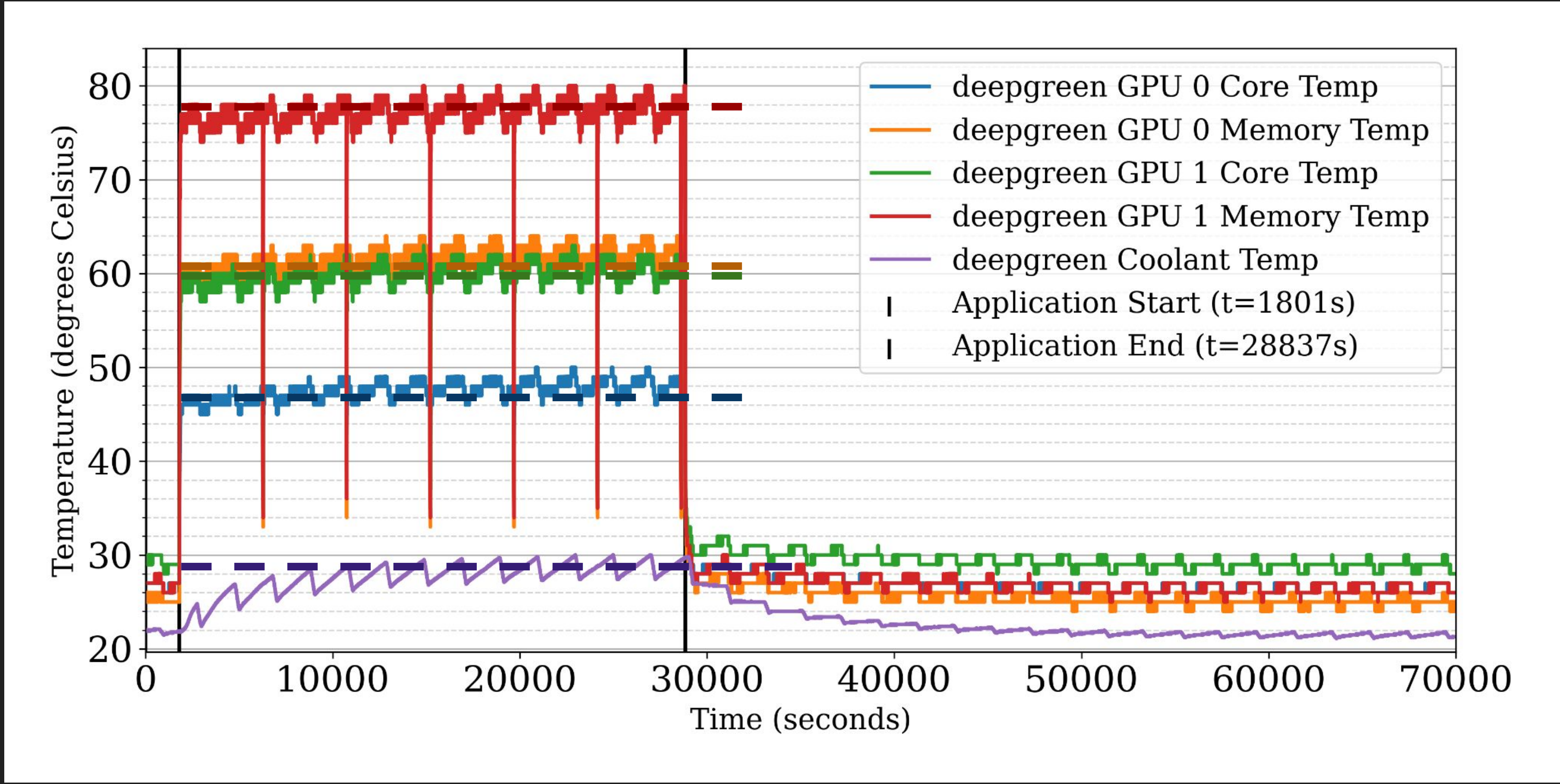


Limitations of Air

- Always cooling (safety)
- Minutes to cool temperatures
- Lower thermal capacity
  - Can’t move much more heat
- Well-tuned, approaching limits
  - Dated GPU hardware

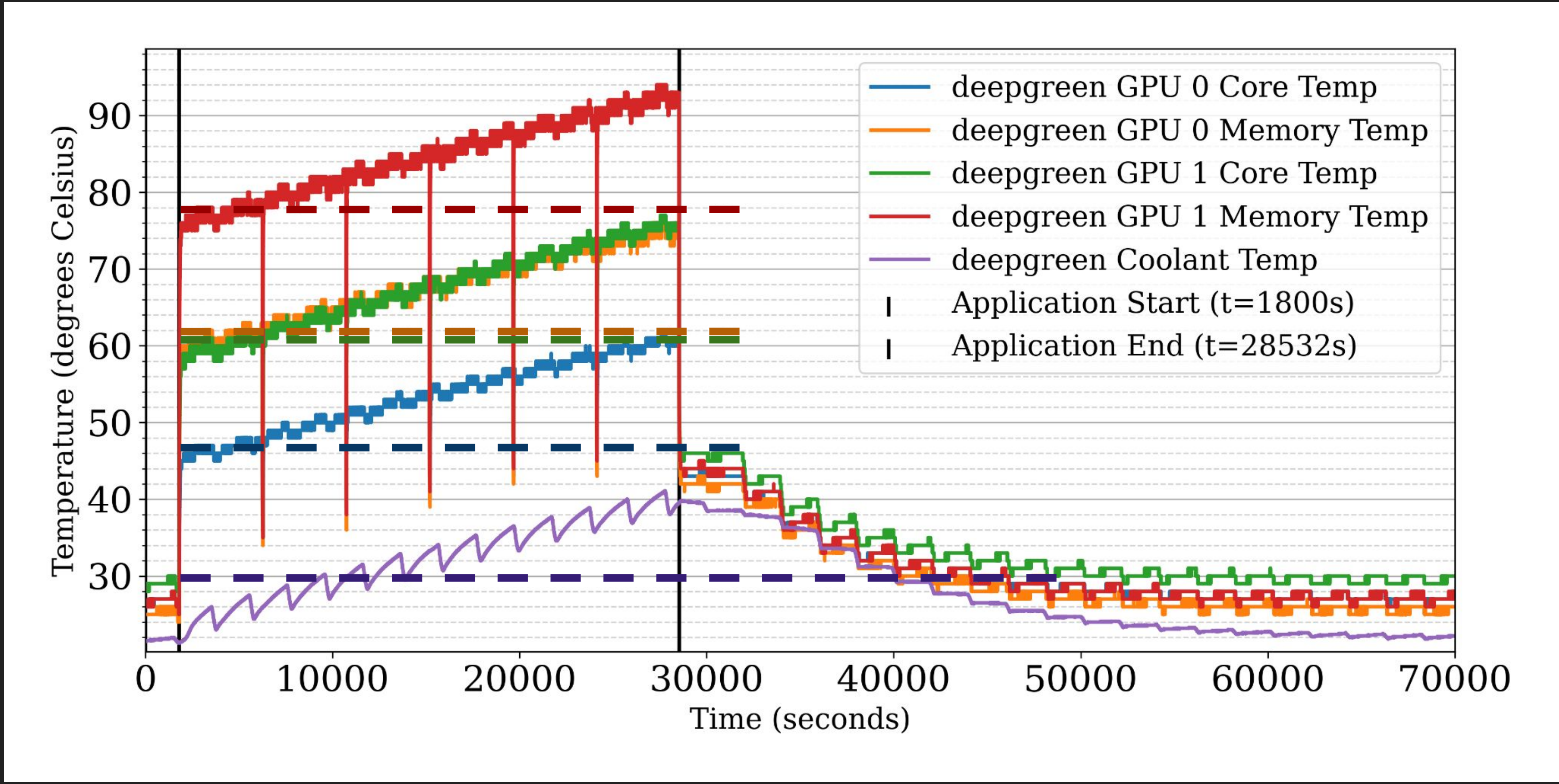


Everything Affected Similarly



- Normal SPLIC operation

Possible Risk to Hardware



- No chilled water while applications run

Energy Efficiency Not Guaranteed

- Passive cycles **waste power**
  - Low thermal burden
  - +15 kJ CUDA Stream
  - +19 kJ HPL
- Peak dissipation **more power efficient**
  - High thermal burden
  - -4 kJ CUDA DGEMM
  - -7 kJ NPB EP
- Tuning **required**

Takeaways

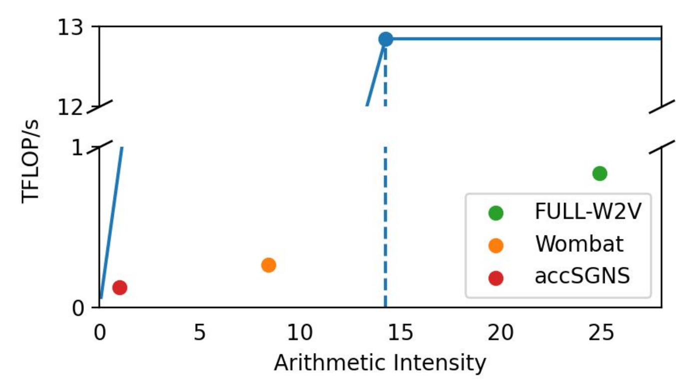
- Coolant equally distributes heat
- SPLIC has delayed responses, possibly wasted energy!
  - Requires tuning
- Compute introduces most heat
  - Other components may be hotter!
- Air limited by capacity

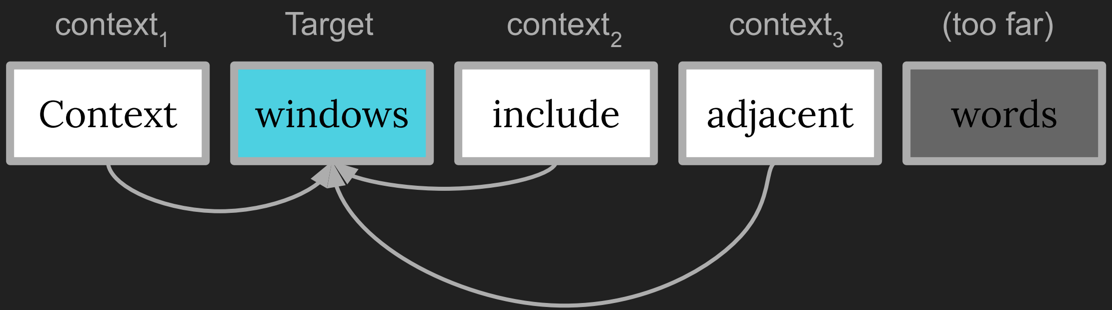
Outline

- Background and Overview
- Systemic Approach
  - Hardware
  - **Software**
    - "FULL-W2V: Fully Exploiting Data Reuse for W2V on GPU-Accelerated Systems" **BEST PAPER** in the Proceedings of the International Conference on Supercomputing 2021
  - Tuning
- Concluding Discussion
- Q&A

Word2Vec: CPU > GPU?

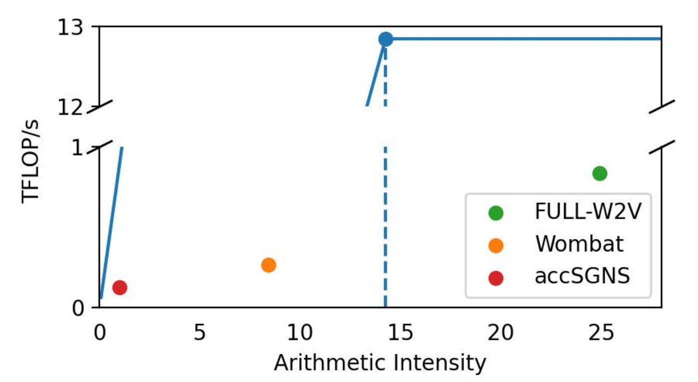
- 3 layer neural network
  - Words $w \rightarrow d$-dimensional embeddings e
- Data-intensive GPU ports
  - Suboptimal usage of **memory hierarchy**
  - We improve fundamental approach to hardware

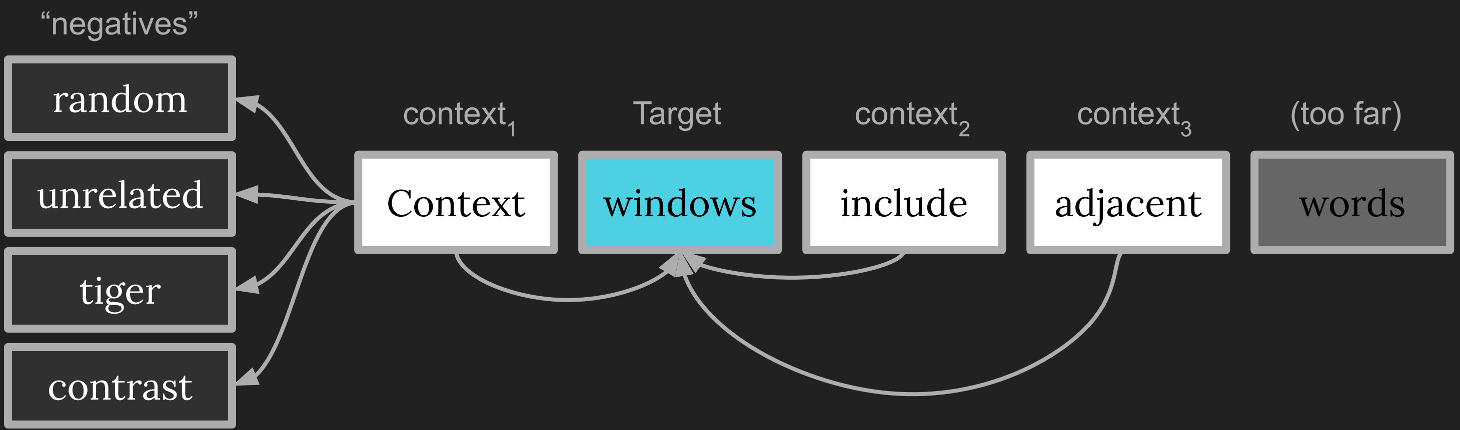




Word2Vec: CPU > GPU?

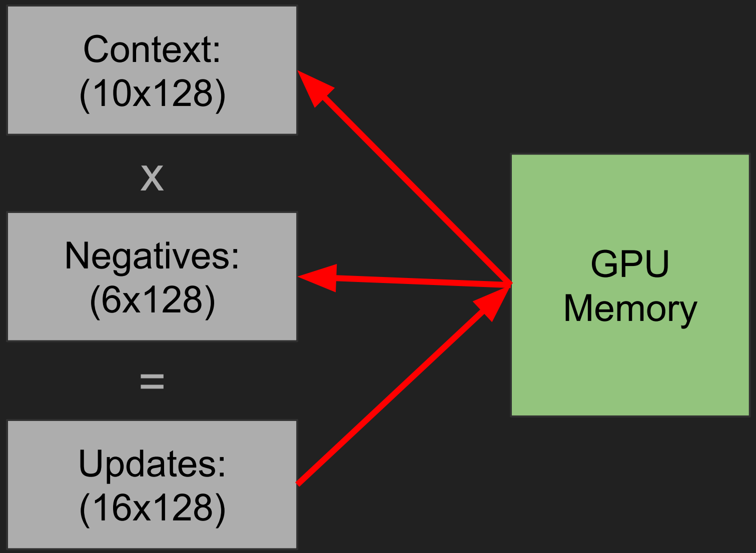
- 3 layer neural network
  - Words $w \rightarrow d$-dimensional embeddings e
- Data-intensive GPU ports
  - Suboptimal usage of **memory hierarchy**
  - We improve fundamental approach to hardware





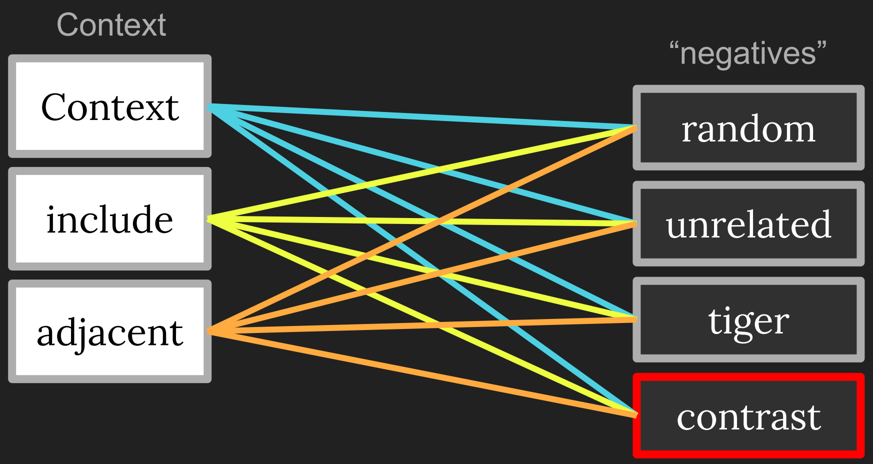
Core Problems for GPUs

- Low computational intensity
  - Small matrices
  - Low reuse
- High cost of memory latency
  - Cache-averse behaviors
  - No locality between matrices
- Preserve embedding quality
  - Extra parallelism → threaten convergence?



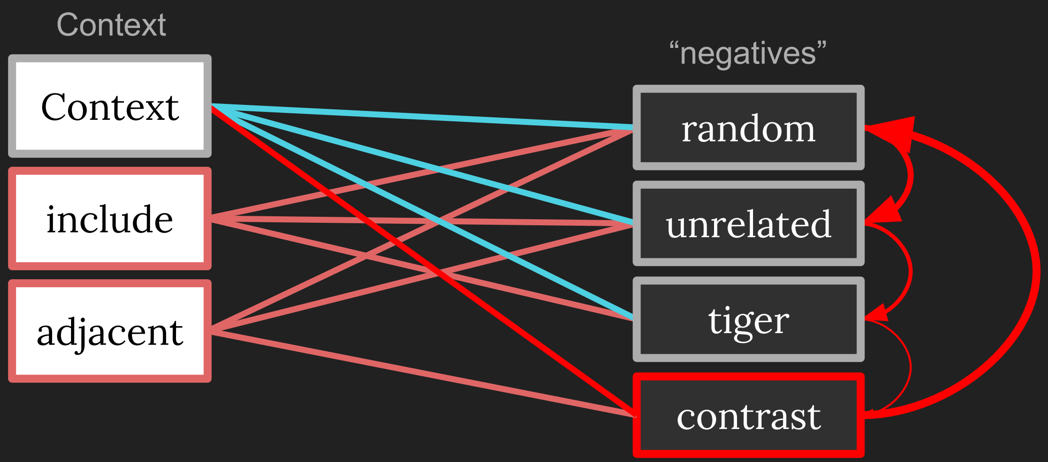
Independence of Negative Samples

- Challenge: **Random** selections
    - Cache misses!



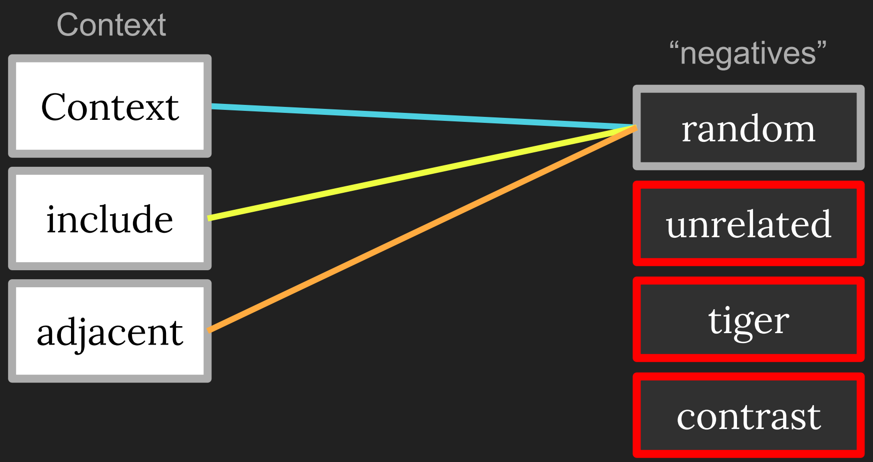
Independence of Negative Samples

- Challenge: **Random** selections
  - Cache misses!



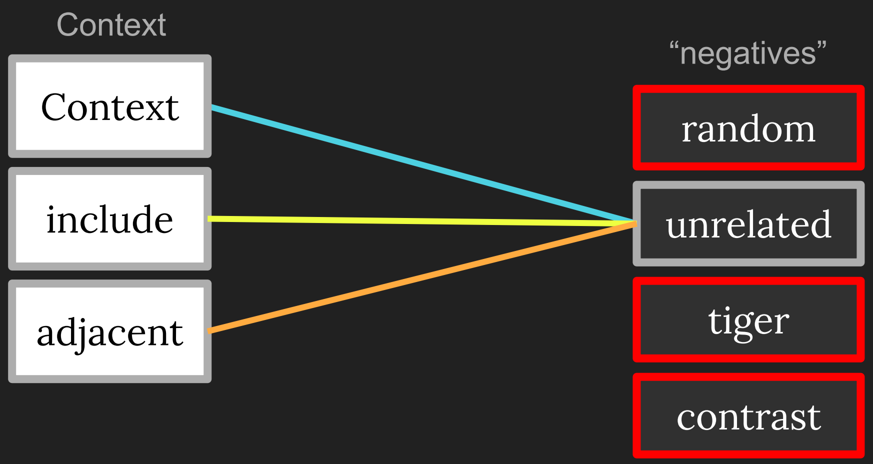
Independence of Negative Samples

- Challenge: **Random** selections
  - Cache misses!
- Opportunity: **Reusable** data



Independence of Negative Samples

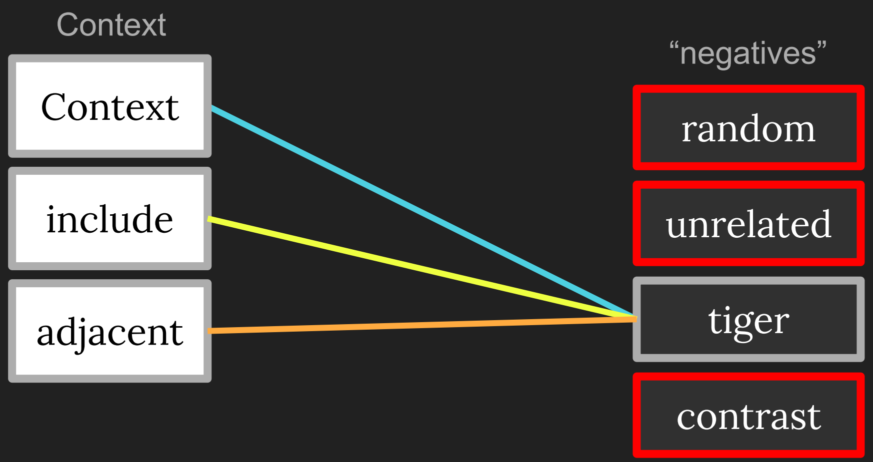
- Challenge: **Random** selections
  - Cache misses!
- Opportunity: **Reusable** data
  - Random = independent order



Independence of Negative Samples

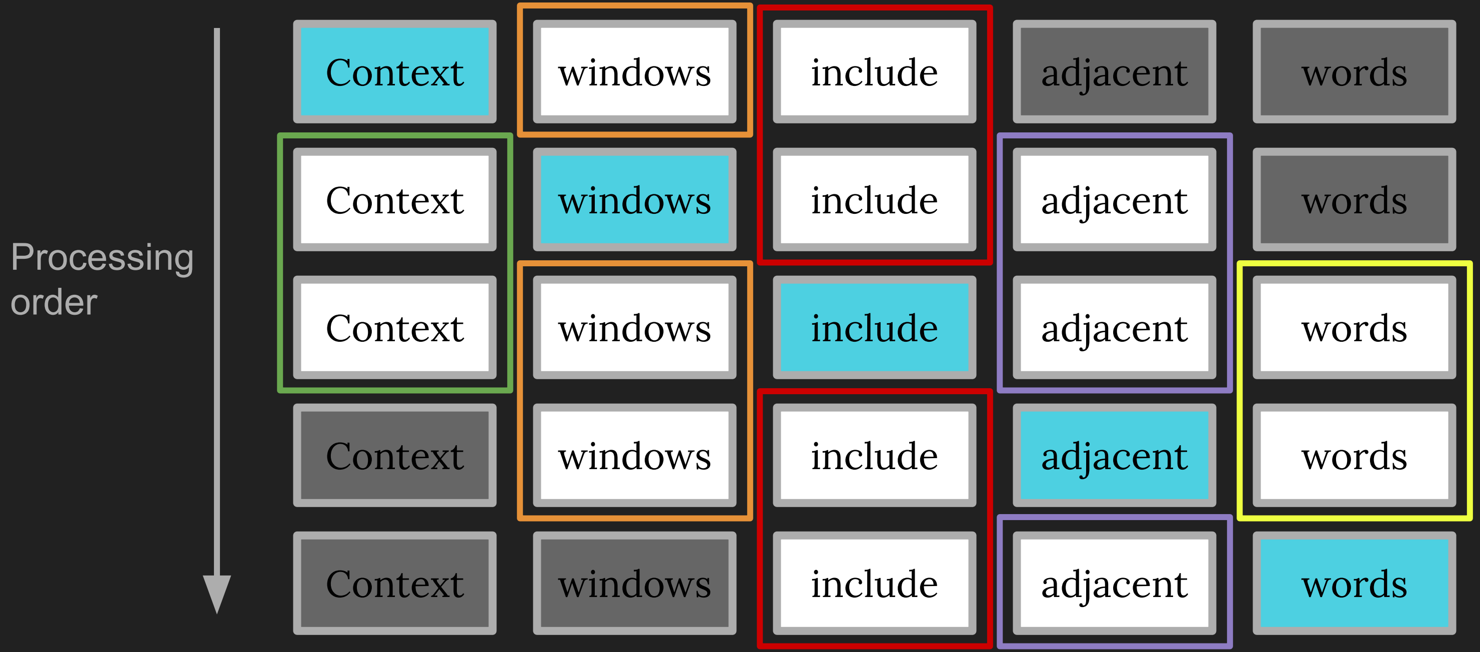
- Challenge: **Random** selections
  - Cache misses!
- Opportunity: **Reusable** data
  - Random = independent order

- Benefits
  - Fine-grain parallelism
  - **Maximize** register usage
  - Interleave computation & latency



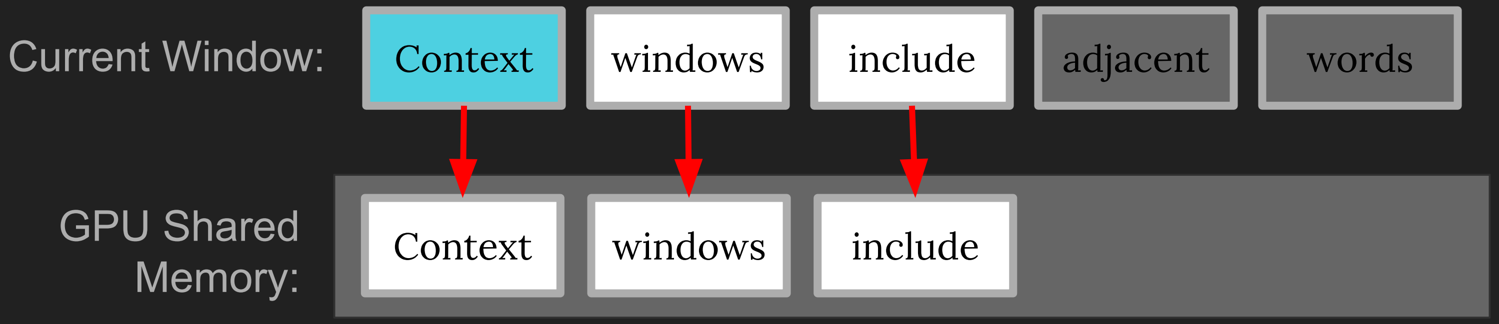
Lifetime Reuse of Context Words

- Context words **reappear**
    - Explicitly cache for full **duration**



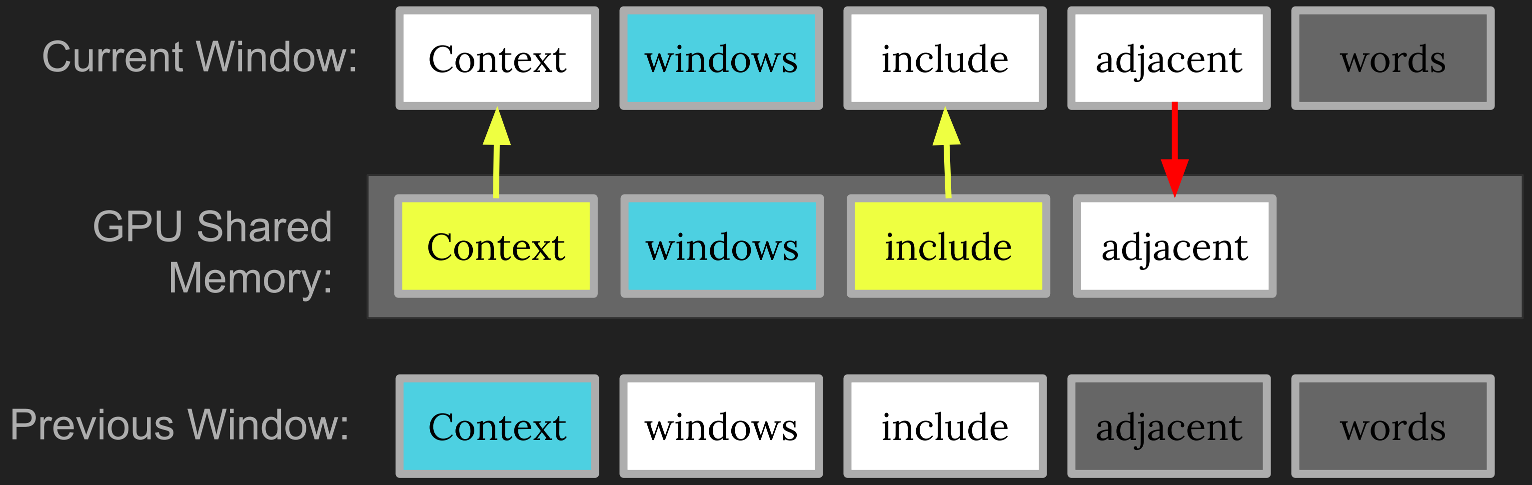
Lifetime Reuse of Context Words

- Context words **reappear**
  - Explicitly cache for full **duration**



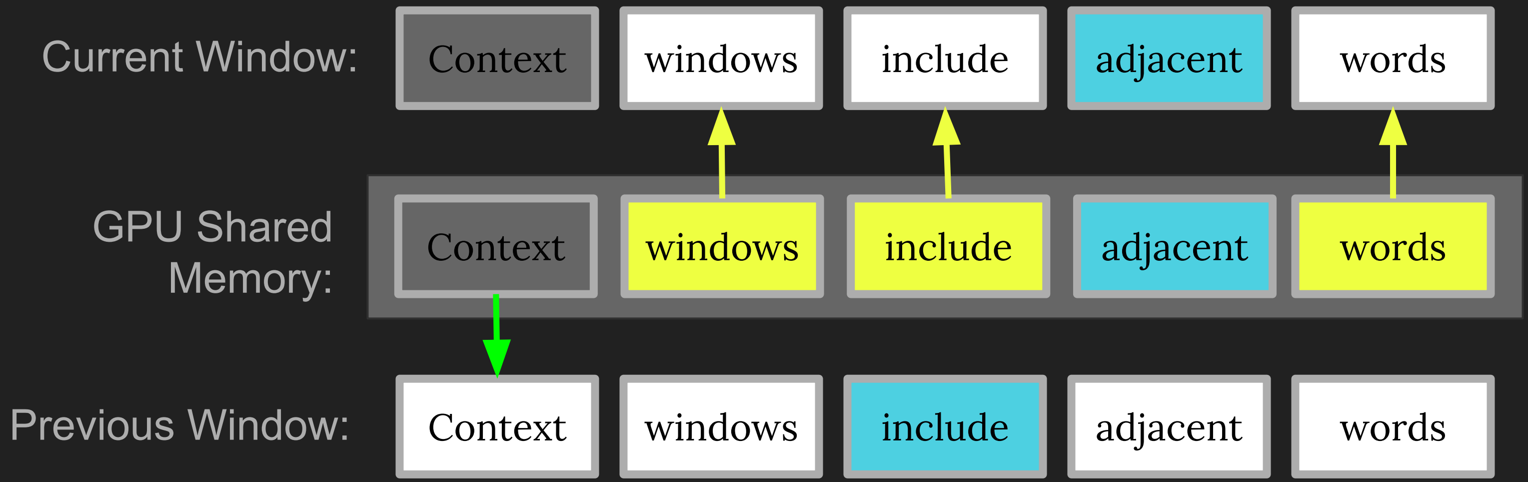
Lifetime Reuse of Context Words

- Context words **reappear**
  - Explicitly cache for full **duration**



Lifetime Reuse of Context Words

- Context words **reappear**
  - Explicitly cache for full **duration**
- Benefits
  - High cache hit rate
  - **89%** R/W reduction



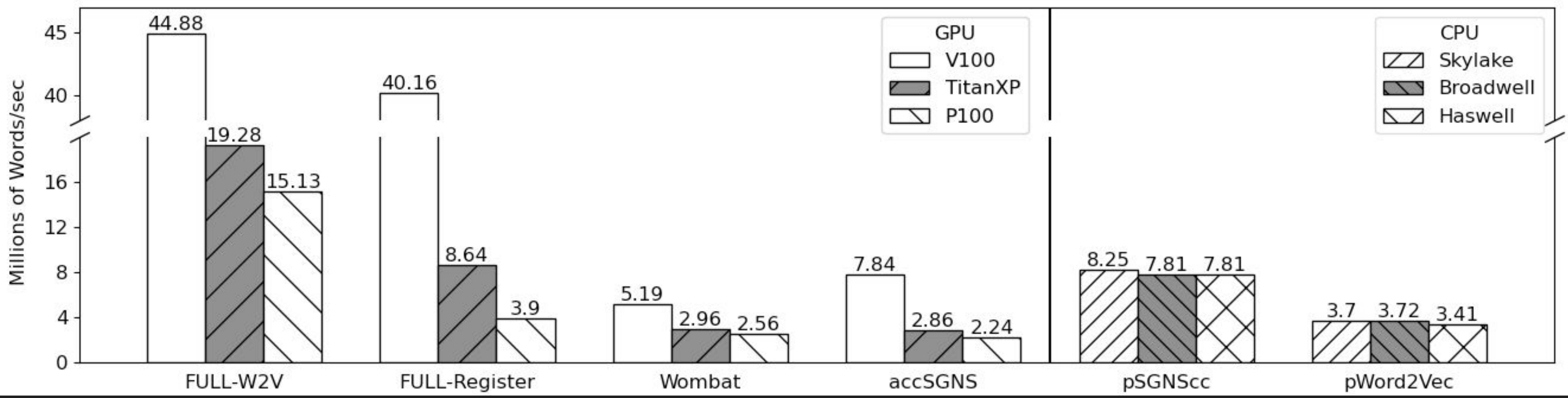
CPU Becomes Bottleneck

- Heterogeneous requirements
  - Reduce syscalls
  - String conversion
- **12.4–15.9X** peak throughput increase
  - Text8 for throughput; 1bw for semantic quality

Implementation	Text8 Batching Speed	1bw Batching Speed
FULL-W2V	210.340633	265.212834
Wombat	16.957496	16.653851
accSGNS	16.527374	15.263448

Highest Throughput Across Generations

- Independence of Negative Samples (INS)
  - **5.12X** faster than CPU on TitanXP
- INS + Lifetime Reuse of Context Words
  - Up to **6.51X** faster than CPU (all architectures)



(INS + LRCW) (INS only) Prior SOTA GPUs Prior SOTA CPUs

Takeaways

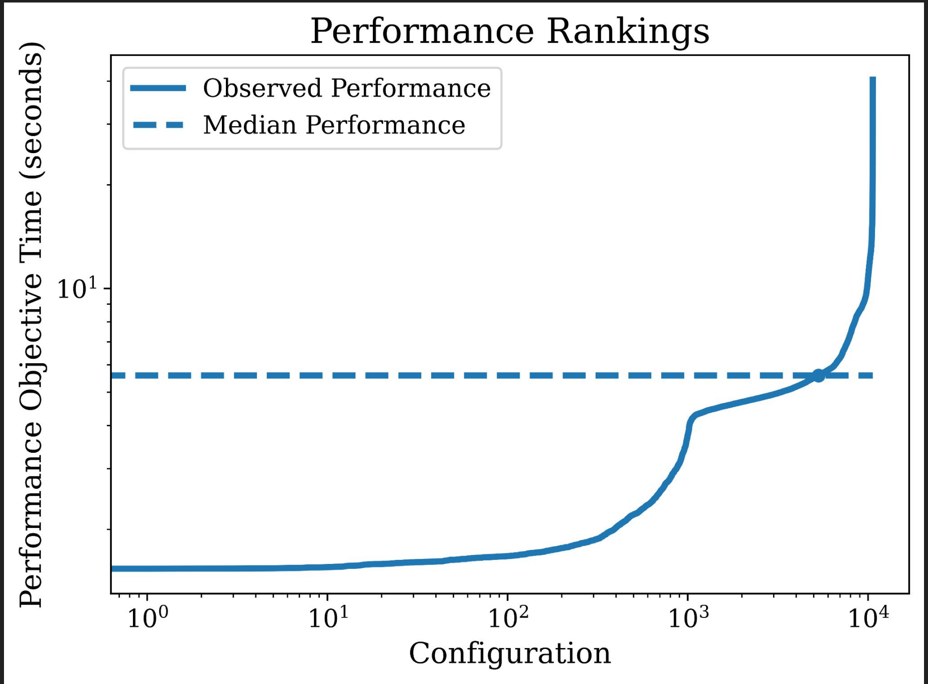
- Massive throughput improvements
  - Better alignment
  - Deep understanding necessary
- Benefit multiple HW generations
  - Register file and shared memory sizes increased
  - Memory latency bottleneck more important

Outline

- Background and Overview
- Systemic Approach
  - Hardware
  - Software
  - **Tuning**
    - "Transfer-learning-based Autotuning using Gaussian Copula" in the Proceedings of the International Conference on Supercomputing 2023
- Concluding Discussion
- Q&A

Put it Together

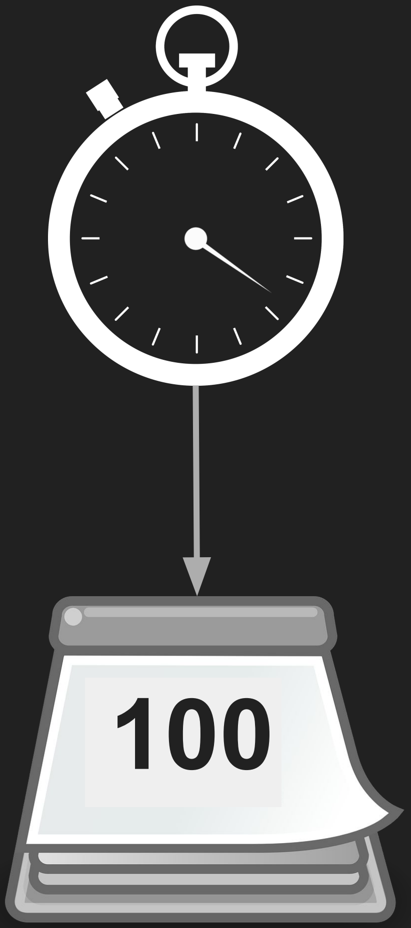
- Tuning is balancing trade-offs
- Best-performing configuration
  - Speedup / energy / latency / precision…
  - Often by an order of magnitude
- Long tail benefits



- Yet, not everything is tuned

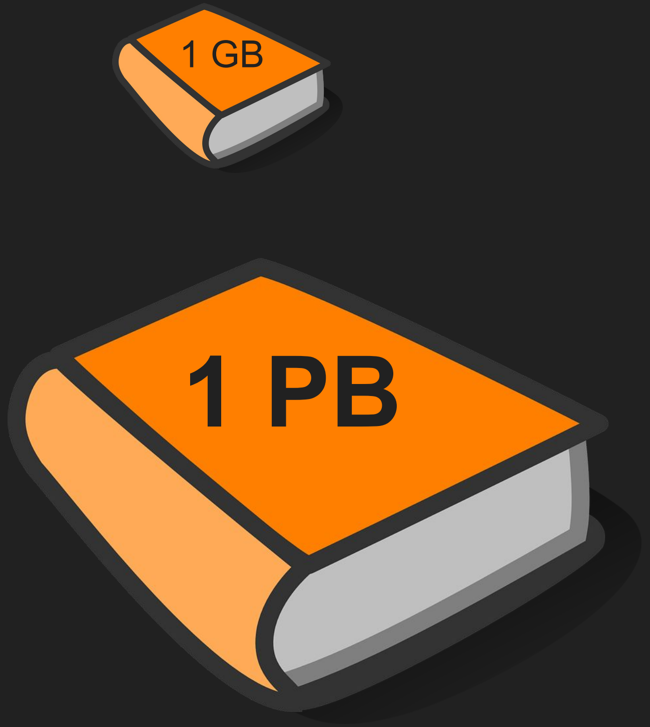
Cost of Tuning

- Small simple kernel
  - **25 seconds** to compile and measure
- Simple tuning space
  - **Ten** source code parameters
  - **10,000+** combinations, easily
- Exhaustive cost
  - **100+ days** of serial compute
  - What if I have different input?
  - What if I have 10,000 machines?



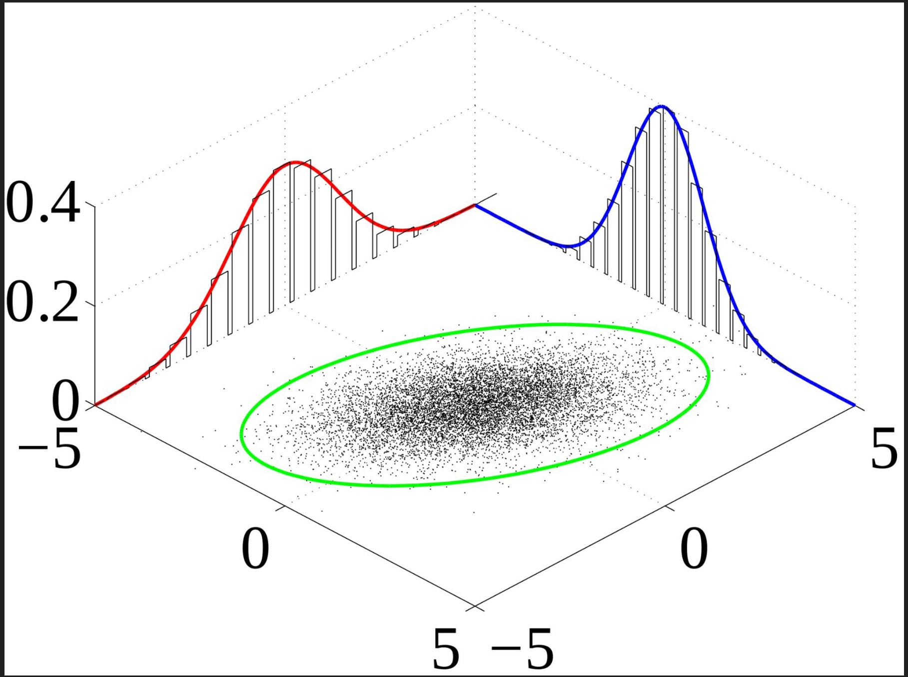
Opportunity for Reuse

- HPC applications tune multiple scales
- Transfer improves efficiency of tuning
  - Less resources
  - Still close to optimal
- Existing transfer techniques
  - Require calibration
  - Regression needs big data
  - Under-utilized



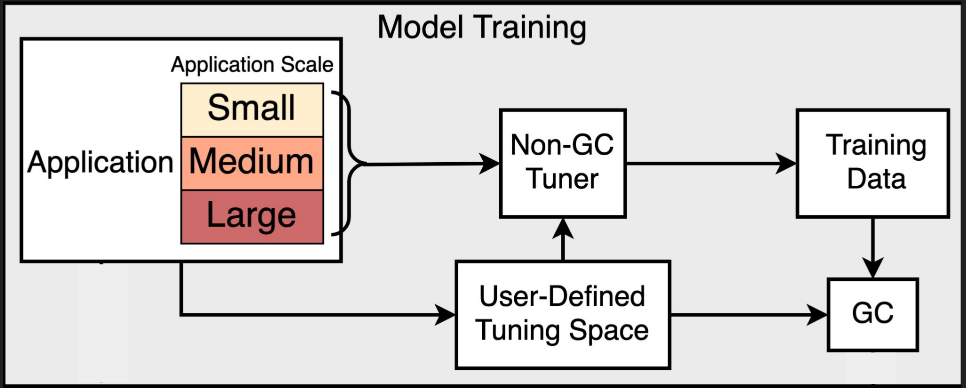
Gaussian Copula (GC) TL-Based Autotuning

- Probabilistic model
- Maximize **few-shot** between tasks
  - Common in HPC
- Transfer without regression
  - Reduce cost
    - Less data
    - Immediate performance
  - Estimate success
    - **Prior** to evaluations



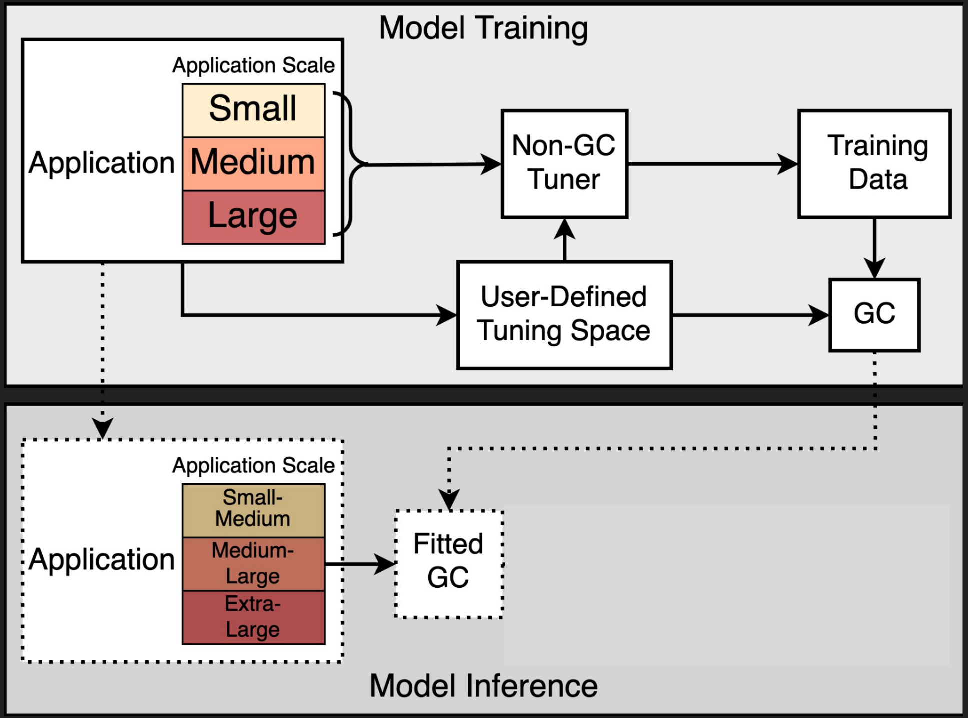
GC Few-Shot TL Autotuning

- Fit space and prior task data



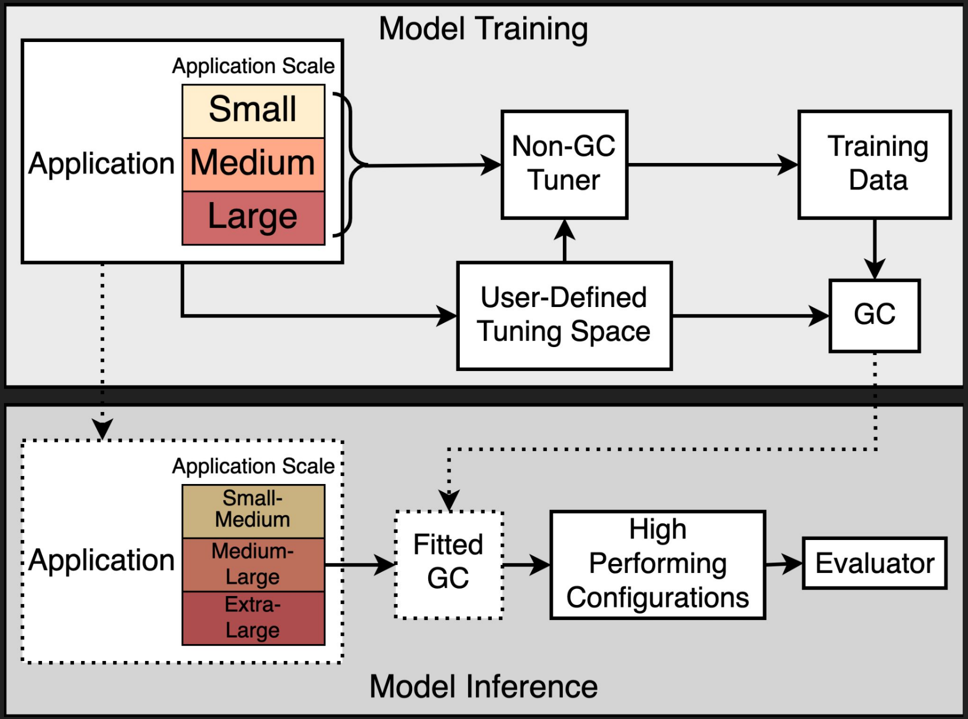
GC Few-Shot TL Autotuning

- Fit space and prior task data
  - Prompt with new task



GC Few-Shot TL Autotuning

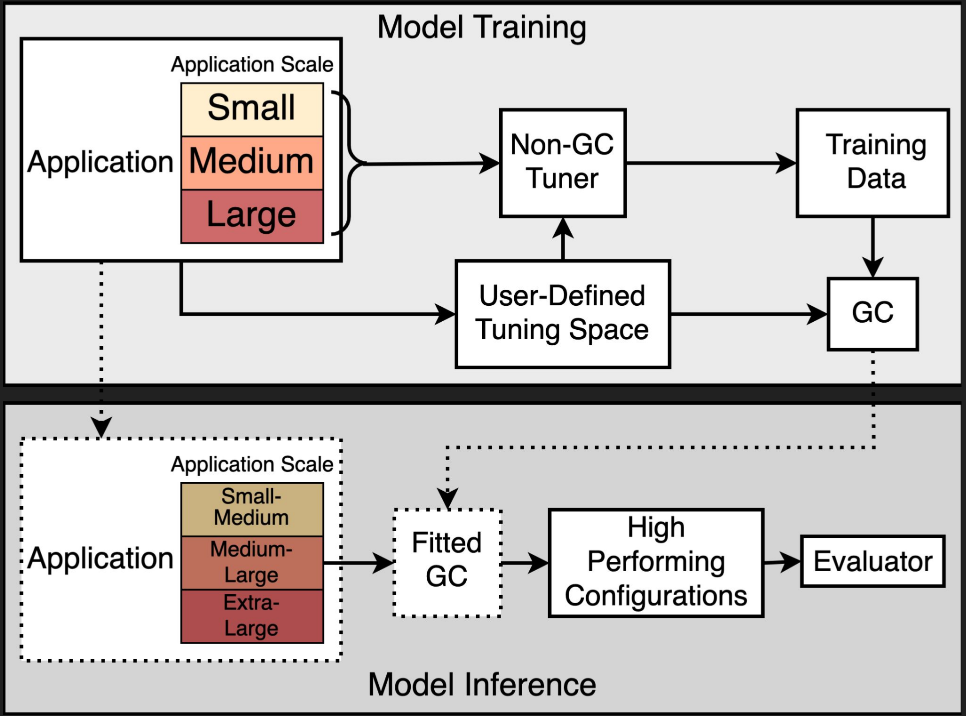
- Fit space and prior task data
  - Prompt with new task
  - Generate evaluation candidates



GC Few-Shot TL Autotuning

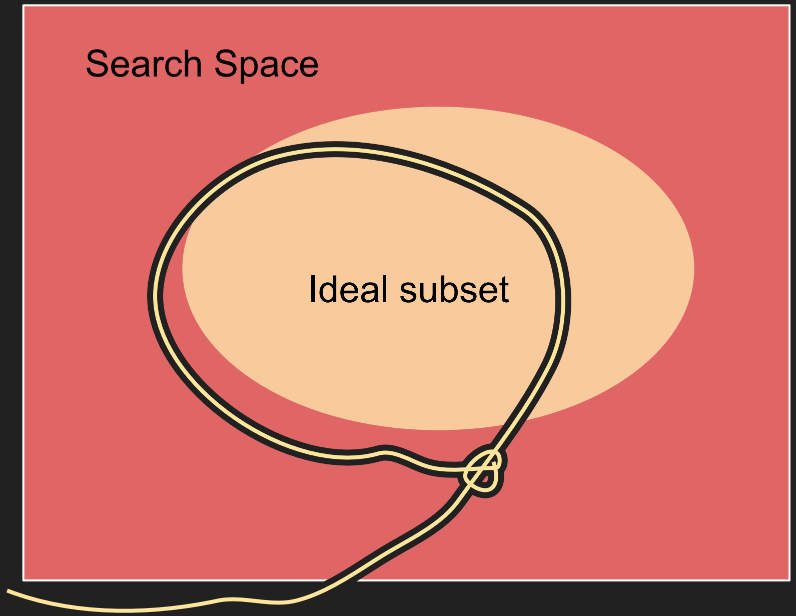
- Fit space and prior task data
  - Prompt with new task
  - Generate evaluation candidates

- Demonstrate on benchmarks
  - **64% peak** in one shot
  - **12.81✕ higher peak** (20.58✕→33.39✕) vs previous SOTA



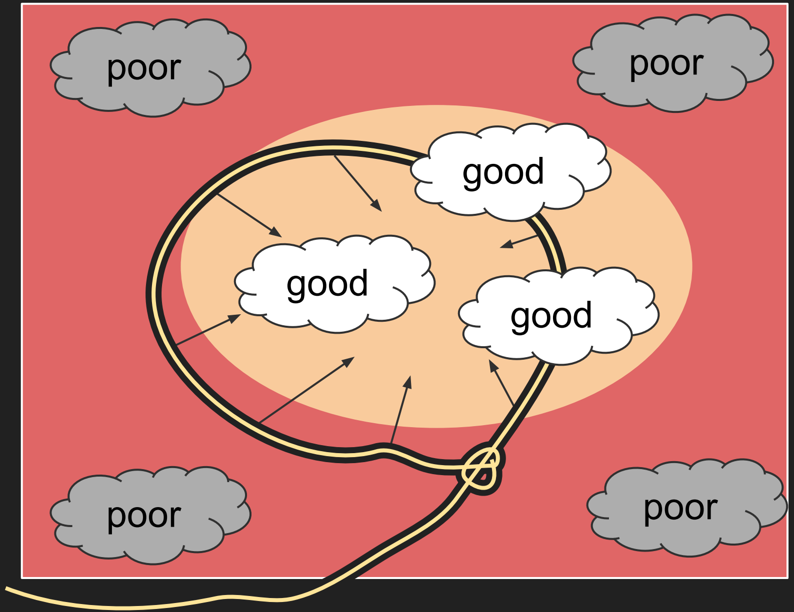
Distributions as Search

- GC lacks regression
  - No comparisons/ranking
  - Minimal data = distribution



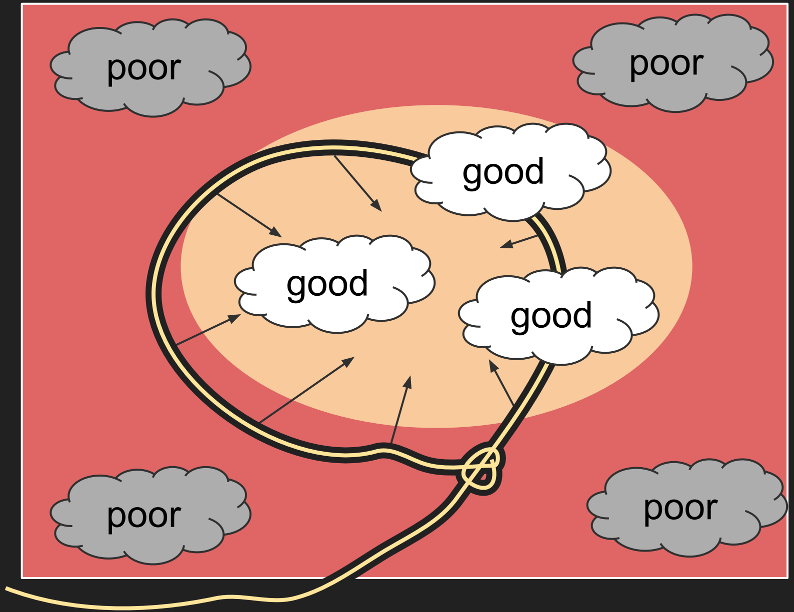
Distributions as Search

- GC lacks regression
  - No comparisons/ranking
  - Minimal data = distribution
- Provide search boundaries
  - Under-represented = Poor traits
  - Over-represented = Solved traits
  - Variance = Opportunity to explore



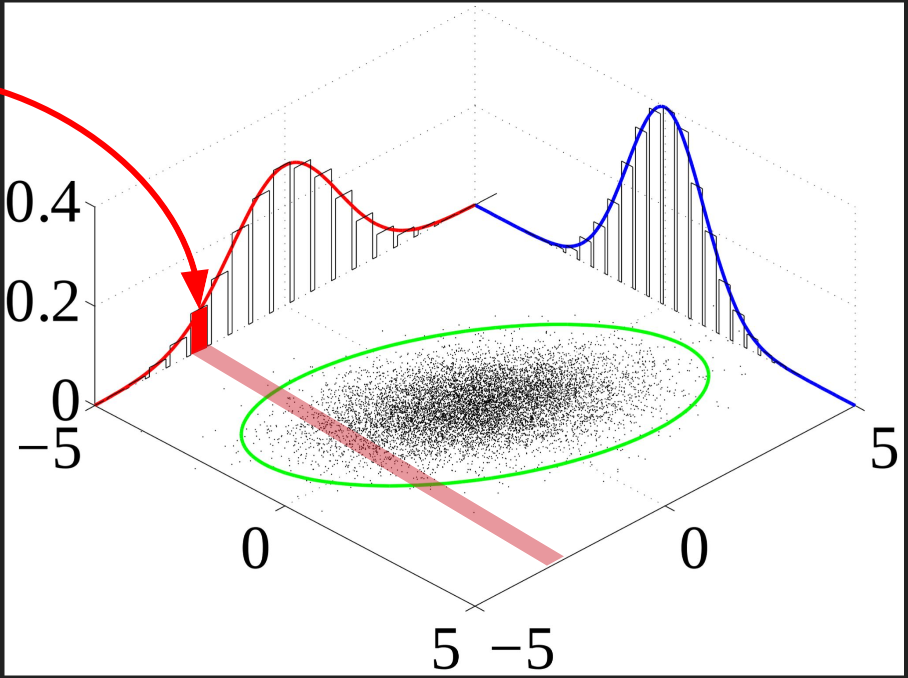
Distributions as Search

- GC lacks regression
  - No comparisons/ranking
  - Minimal data = distribution
- Provide search boundaries
  - Under-represented = Poor traits
  - Over-represented = Solved traits
  - Variance = Opportunity to explore
- Probability estimate
  - **Predict** # evaluations



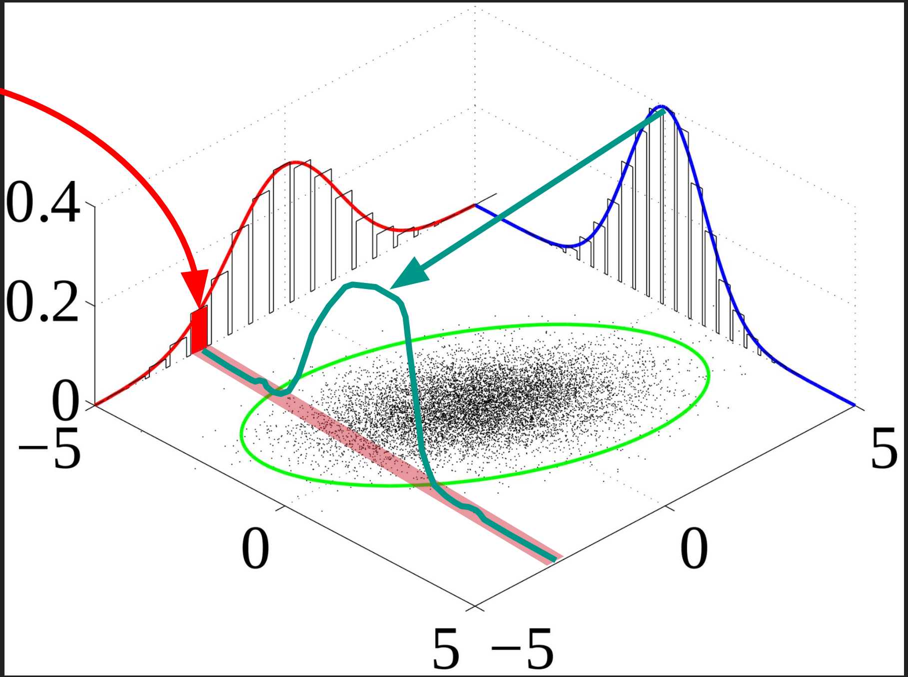
Conditional Sampling as Transfer

- Different **scales** require different solutions
  - Impose constraint



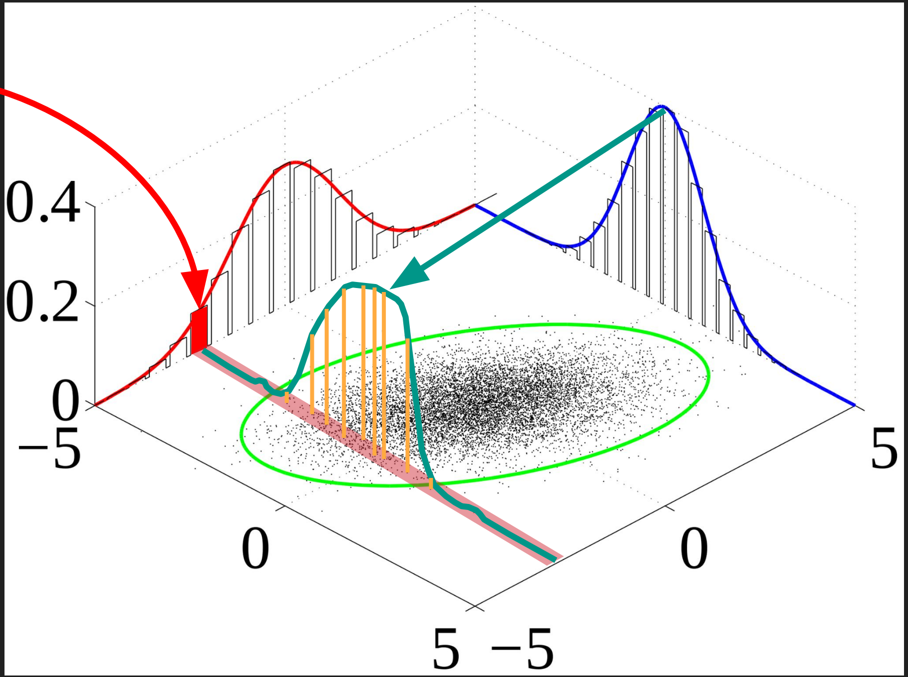
Conditional Sampling as Transfer

- Different **scales** require different solutions
  - Impose constraint
- Other **marginals** adjusted
  - All data recontextualized



Conditional Sampling as Transfer

- Different **scales** require different solutions
  - Impose constraint
- Other **marginals** adjusted
  - All data recontextualized
- **Sample** from distribution
  - No wasted samples



Immediate Performance

- Polybench Applications
  - Linear Algebra, Image Processing, Stencils, Data Mining
- 3mm XL: **12.81✕** more speedup than prior SOTA

App.	Scale	Peak Speedup (# Evaluation Discovered)				
		GC			BO	GPTune
		1^{st}	Budget	Best	Best	Best
3mm	SM	5.09	5.70 (23)	5.70 (23)	3.03 (26)	5.53 (30)
	ML	5.25	5.57 (29)	5.57 (29)	3.29 (30)	5.16 (16)
	XL	27.10	33.39 (18)	33.39 (18)	20.58 (30)	18.96 (25)

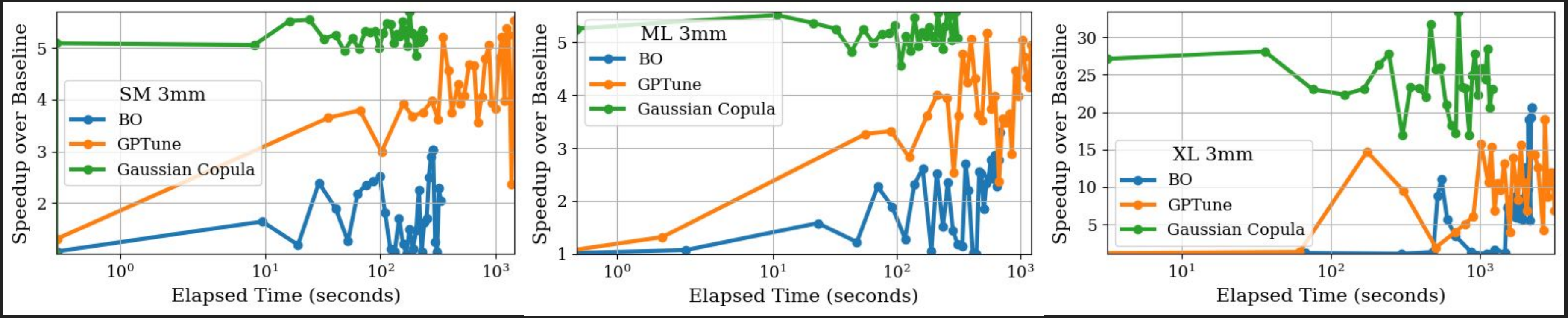
Immediate Performance

- Polybench Applications
  - Linear Algebra, Image Processing, Stencils, Data Mining
- 3mm XL: **+12.81✕** more speedup than prior SOTA
- GC exceeds prior SOTA performance
  - 1st evaluation: 44%
  - Within budget: 78%
- Worst margin of performance is **-0.24✕** speedup

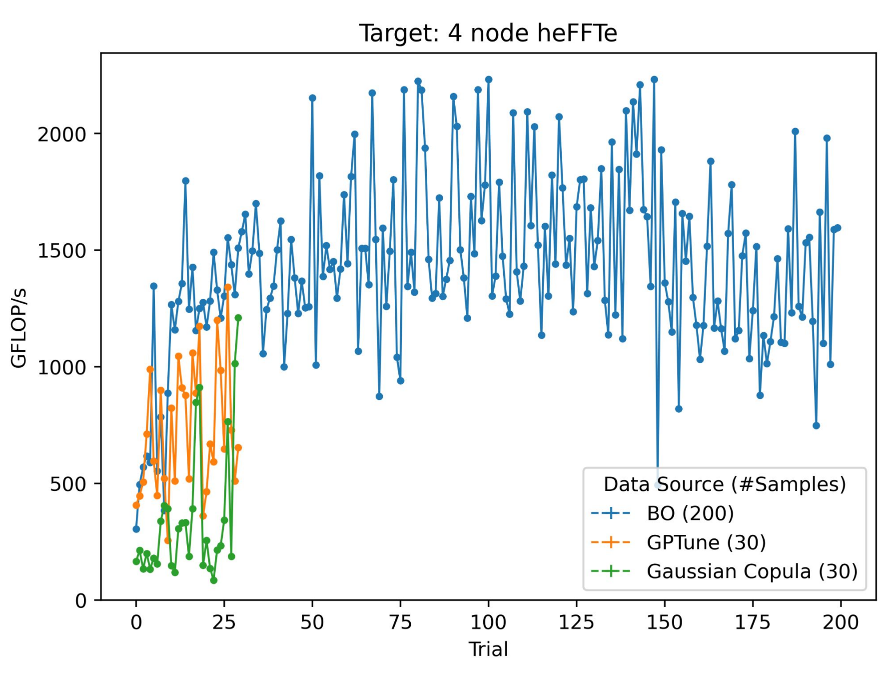
App.	Scale	Peak Speedup (# Evaluation Discovered)				
		GC			BO	GPTune
		1^{st}	Budget	Best	Best	Best
3mm	SM	5.09	5.70 (23)	5.70 (23)	3.03 (26)	5.53 (30)
	ML	5.25	5.57 (29)	5.57 (29)	3.29 (30)	5.16 (16)
	XL	27.10	33.39 (18)	33.39 (18)	20.58 (30)	18.96 (25)
Cov.	SM	21.10	21.98 (21)	21.98 (21)	21.83 (28)	13.30 (30)
	ML	4.13	4.27 (26)	4.27 (26)	3.87 (25)	4.07 (30)
	XL	23.04	23.96 (2)	23.96 (2)	8.43 (12)	17.88 (9)
Floyd-W.	SM	1.01	1.02 (17)	1.02 (17)	1.02 (20)	1.01 (26)
	ML	1.02	1.02 (1)	1.02 (1)	1.01 (25)	1.01 (3)
	XL	0.99	1.00 (29)	1.00 (29)	1.01 (16)	1.01 (20)
Heat3d	SM	1.83	2.03 (5)	2.06 (18)	2.21 (15)	2.30 (28)
	ML	1.89	1.89 (1)	2.06 (10)	2.12 (25)	1.80 (6)
	XL	1.50	2.92 (2)	3.09 (18)	2.16 (13)	2.75 (29)
LU	SM	1.16	1.18 (25)	1.18 (25)	1.12 (30)	1.11 (19)
	ML	1.15	1.20 (24)	1.20 (24)	1.17 (26)	1.19 (5)
	XL	1.00	1.00 (3)	1.00 (3)	0.98 (13)	1.00 (29)
Syr2k	SM	2.06	2.90 (2)	3.32 (18)	2.34 (12)	2.41 (11)
	ML	0.80	1.17 (2)	1.22 (16)	0.93 (29)	0.85 (30)
	XL	0.95	1.09 (2)	1.09 (2)	0.42 (23)	0.85 (26)

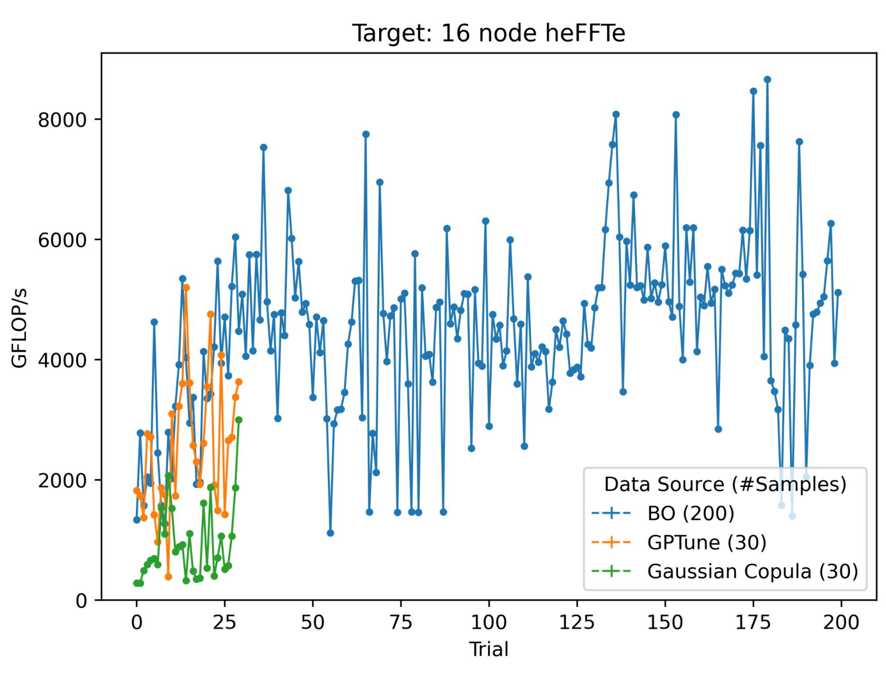
Consistently Better

- GC selects better configuration than prior work almost every single evaluation



Limited By Complexity

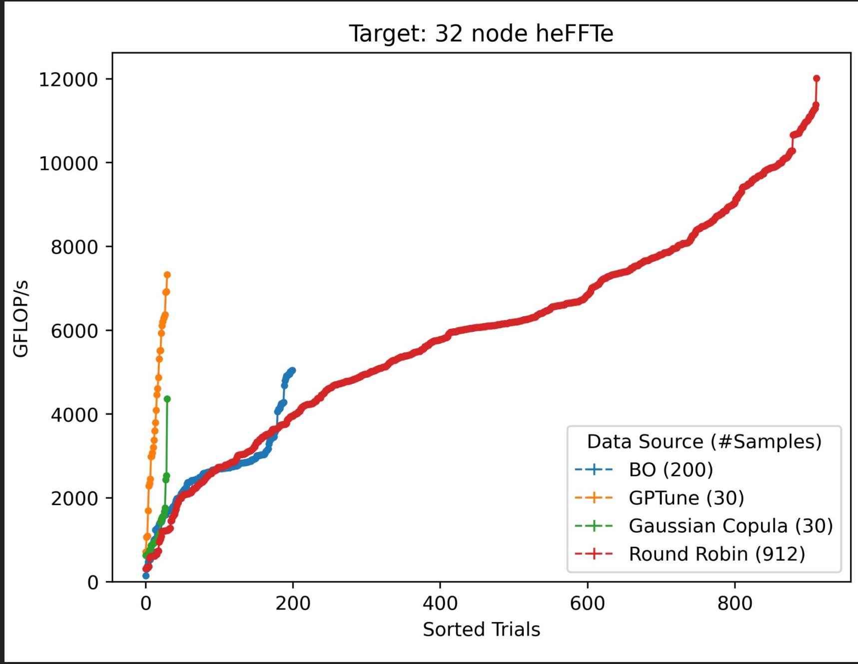




- heFFTe: Exascale benchmark application
- Most learning must be performed on-task

Explaining Failure

- Knowledge has low utility
  - Tasks change too much
  - Dependent & complex variable relationships
- GCTLA's budget and correlation **warn**!
  - No budget
  - No correlation despite expectation
- Warning, not solution
  - More training data?
  - Train from scratch?



Outline

- Background and Overview
- Systemic Approach
  - Hardware
  - Software
  - Tuning
- **Concluding Discussion**
- Q&A

Summary and Broader Impacts

- Exciting hardware
  - Improve thermal management within HPC
  - Develop designs together
- Effective software
  - Leverage deep understanding
  - Performance scaling to hardware
- Practical tuning
  - Novel technique effective with minimal data
  - Broader utilization, greater clarity

Future Works (Science is never “done” ™)

- Hardware
  - Full accounting of water and energy
  - Effectiveness of forced induction
  - Firmware adaptation for better demand response
- Software
  - Locality between applications and BLAS
  - Determining maximum extent of reuse
  - New application domains
- Tuning
  - Other probability models
  - Supporting model paradigms
  - Multi-scale tuning remains challenging!

Funding Acknowledgements

Portions of this material were supported by the National Science Foundation under Grant Nos. MRI# 2024205, MRI# 1725573, and CRI# 2010270; as well as Grants CNS-CCF-1942182, CNS-1551262, and CCF-1551511.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

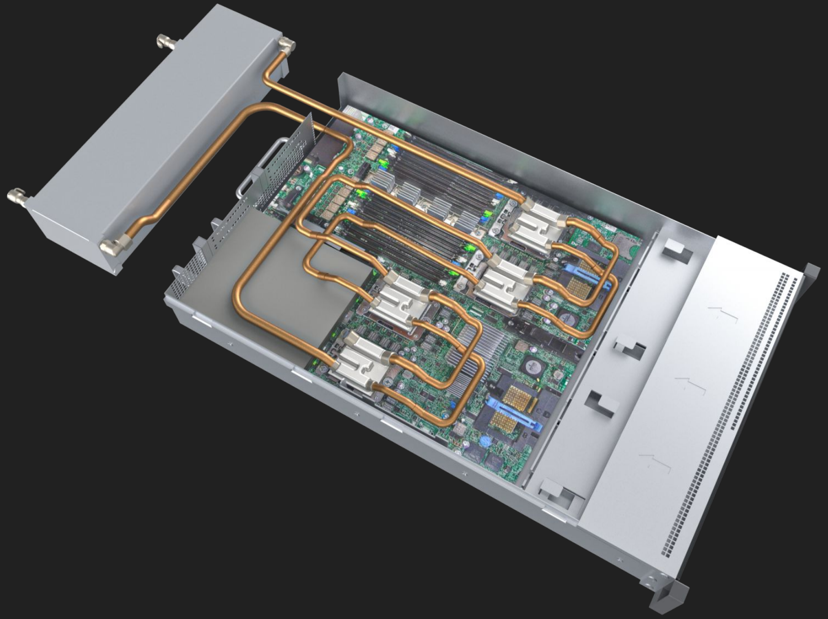
Clemson University is acknowledged for generous allotment of compute time on the Palmetto Cluster.

Portions of this material were supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

Backup Slides!

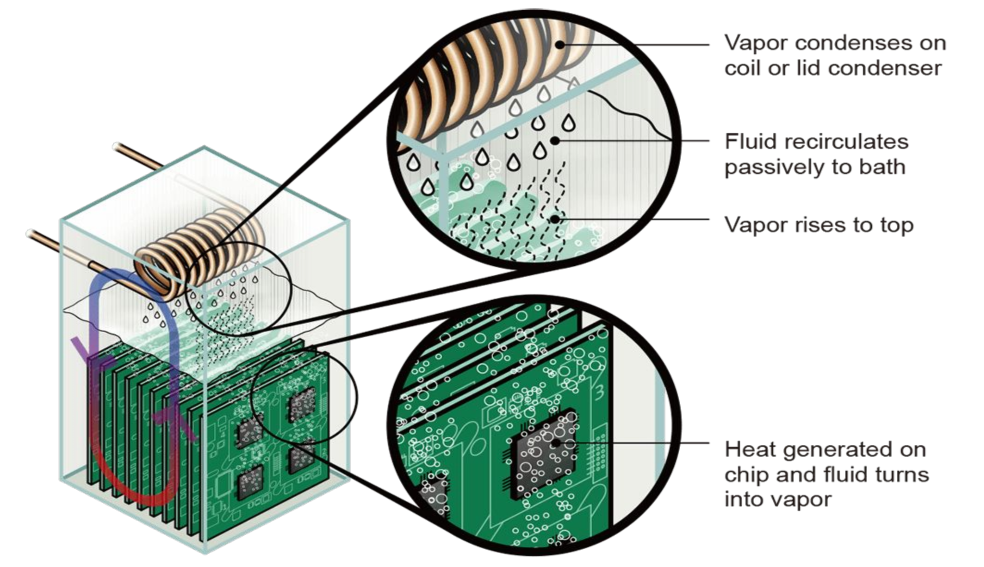
Direct Liquid Cooling

- Circulate liquid via cold plates
  - Microchannels
- Simple in-rack setup
  - More efficient with hotter fluid
- Fluid completely contained
  - Safety for components
  - Minor effectiveness loss
  - Low maintenance overhead



Dual Phase Immersion Cooling

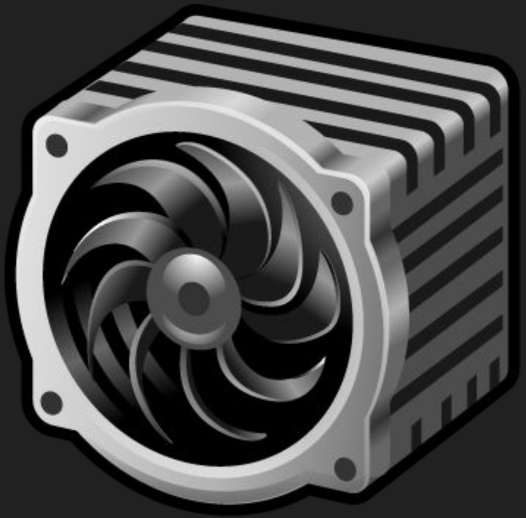
- Physics-driven heat transfer
- No moving parts
- Gas complicates maintenance & safety
- Fluids not eco friendly
- Nonzero risk of combustion!



The Passive 2-Phase Immersion Cooling Cycle

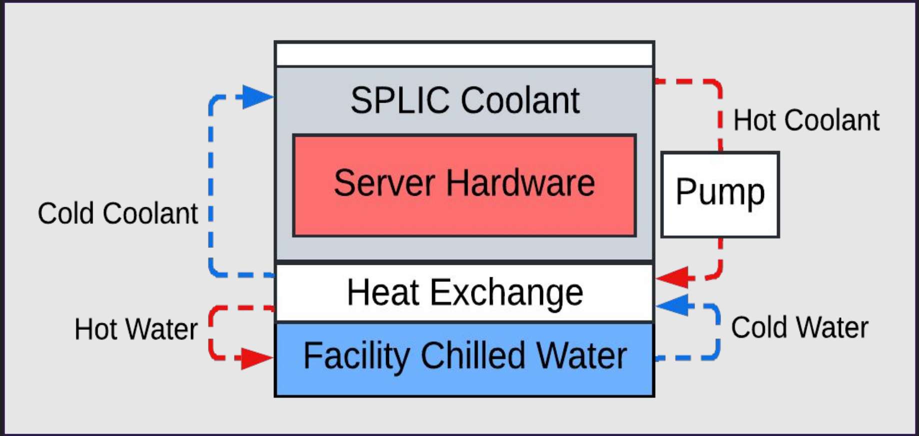
Issues of energy and efficiency

- LBNL: ~2% US electricity in data centers
  - International Energy Agency: Europe ~1.5%
- McKinsey & Company: 40% data center energy for cooling
  - +10% year-over-year
- Air-based cooling won’t grow
  - Current GPUs throttle
  - kW-scale TDPs melt
- Water conducts 30x more heat per unit volume



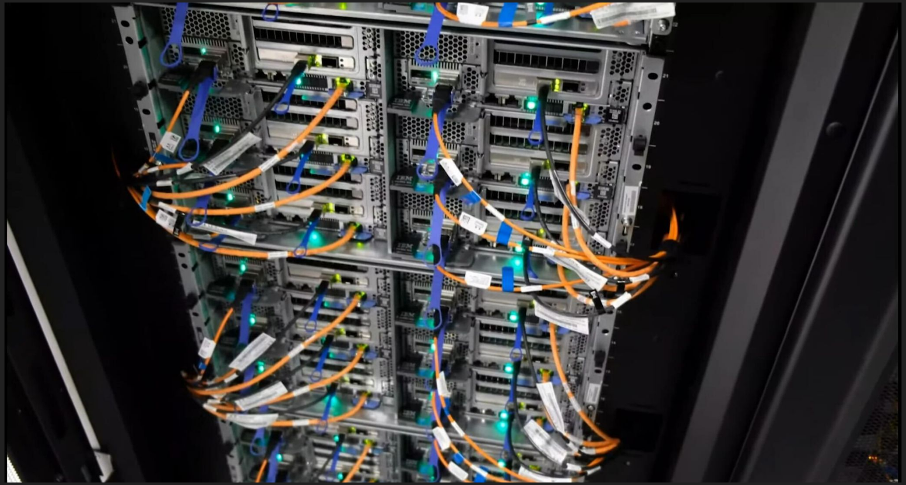
Hardware Preparation

- Remove
  - Thermal paste
  - Fans from power supply, motherboard
- Add
  - Indium foil
- Be careful!
  - Ethernet & power cables



Air-Cooled Replica Server

- Immersion can be a one-way trip
  - Some hardware is immersion-only
- Near-identical servers in air-cooled rack/cabinet
  - Fans in chassis
  - Rear door heat exchange



SPLIC and Air-cooled Evaluation Platforms

- Right: SPLIC
- Bottom: Air-cooled

Nodes	Kind	Hardware	Specs
Deepgreen	Mother-board	S8021GM2NR-2T	Five (5) PCIe 3.0 x16 slots
	CPU	AMD EPYC 7551P	2.00 GHz 32 cores
	GPU	Two (2) NVIDIA Titan V	5120 cores 12 GB HBM2
n01, n02	Mother-board	X9DRG-QF	Four (4) PCIe 3.0 x16 slots
	CPU	Intel(R) Xeon(R) CPU E5-2670 v3	2.30 GHz 64 cores

Nodes	Kind	Hardware	Specs
Palmetto S12	Mother-board	DELL 0D9WDC	Four (4) PCIe 3.0 x16 slots
	CPU	Intel(R) Xeon(R) E5-2680 v4	2.40 GHz 28 cores
	GPU	Two (2) NVIDIA P100	3584 cores 12 GB HBM2

Thermal Deltas by Hardware Component

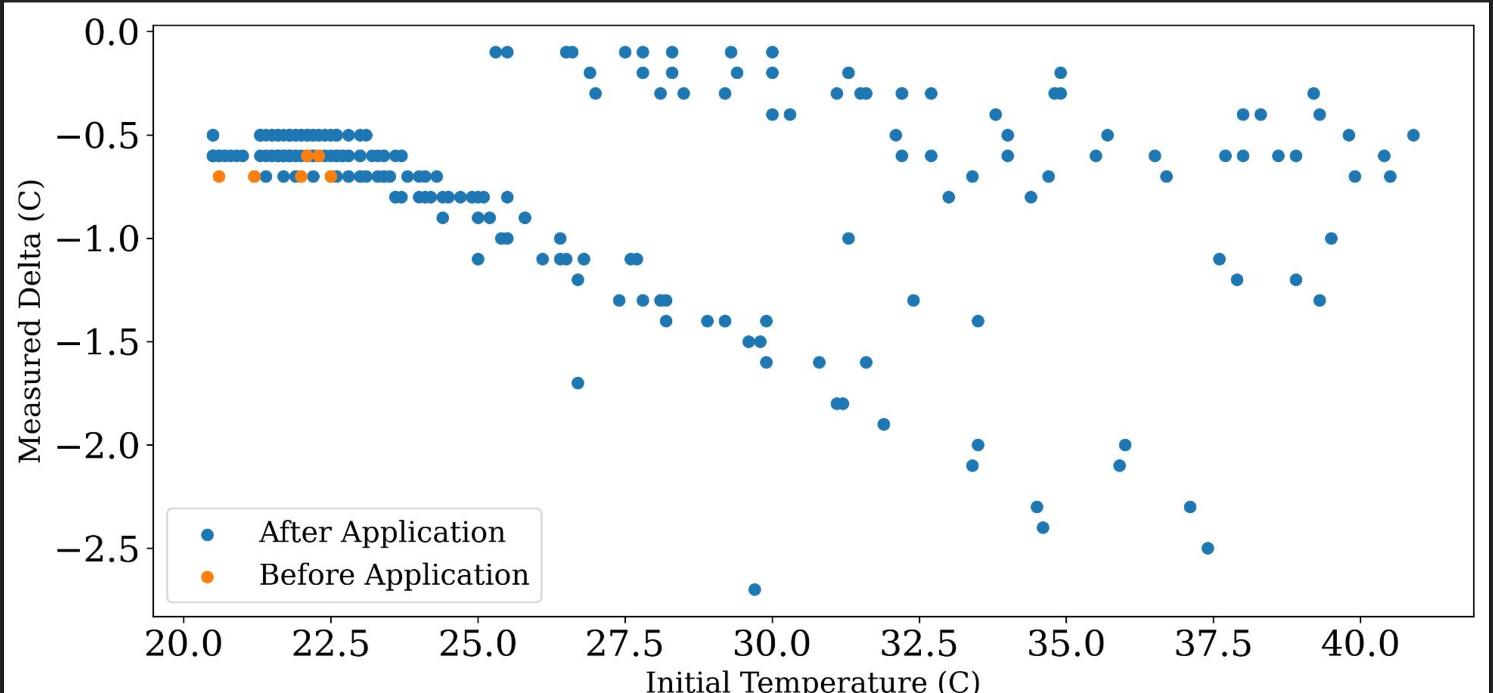
- Compute drives heat
  - Only GPU risks thermal
- Air is somewhat uniform
  - CPUs heat up 1.3X faster than SPLIC
  - GPUs heat up 1.2X slower than SPLIC

Hardware Component	Idle Temperature Minimum/Mean /Maximum (°C)	Interval (s) to Increase Temp by 1°C	Maximum Observed Operating Temperature (°C)
node0091 CPU	27.06 / 27.32 / 28.10	475.86	54.00
node0091 GPU	26.00 / 26.49 / 27.00	309.89	66.00
node0048 CPU	25.32 / 25.66 / 26.26	1325.07	45.13
node0048 GPU (idle)	25.00 / 25.50 / 26.00	315.08	27.00

Hardware Component	Idle Temp. Min/Mean /Max (°C)	Interval (s) to Increase Temp by 1°C	Op. Temp. Observed Max (°C)
SPLIC Coolant	19.80 / 21.41 / 22.50	970.99	47.10
deepgreen CPU	14.75 / 17.51 / 19.66	702.20	55.62
deepgreen GPU	24.25 / 26.54 / 28.00	266.83	96.00
deepgreen NVMe	14.00 / 17.91 / 20.00	1068.48	44.00
n01 CPU	17.44 / 21.30 / 24.04	607.33	66.00
n02 CPU	17.67 / 20.81 / 24.26	560.35	69.00

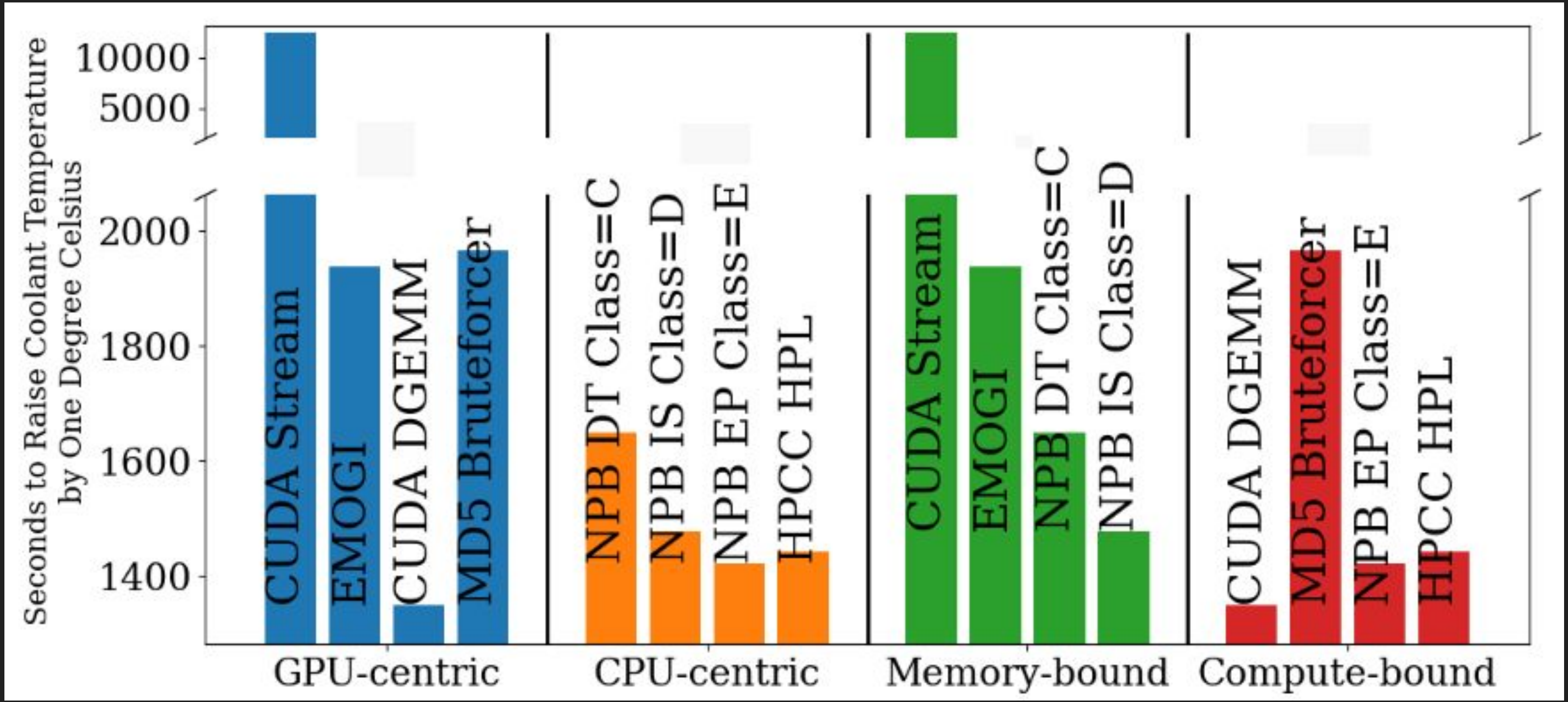
Inconsistent, Improving Efficiency

- Pumps reduce hotter coolant temperature more
- Many **ineffective** periods

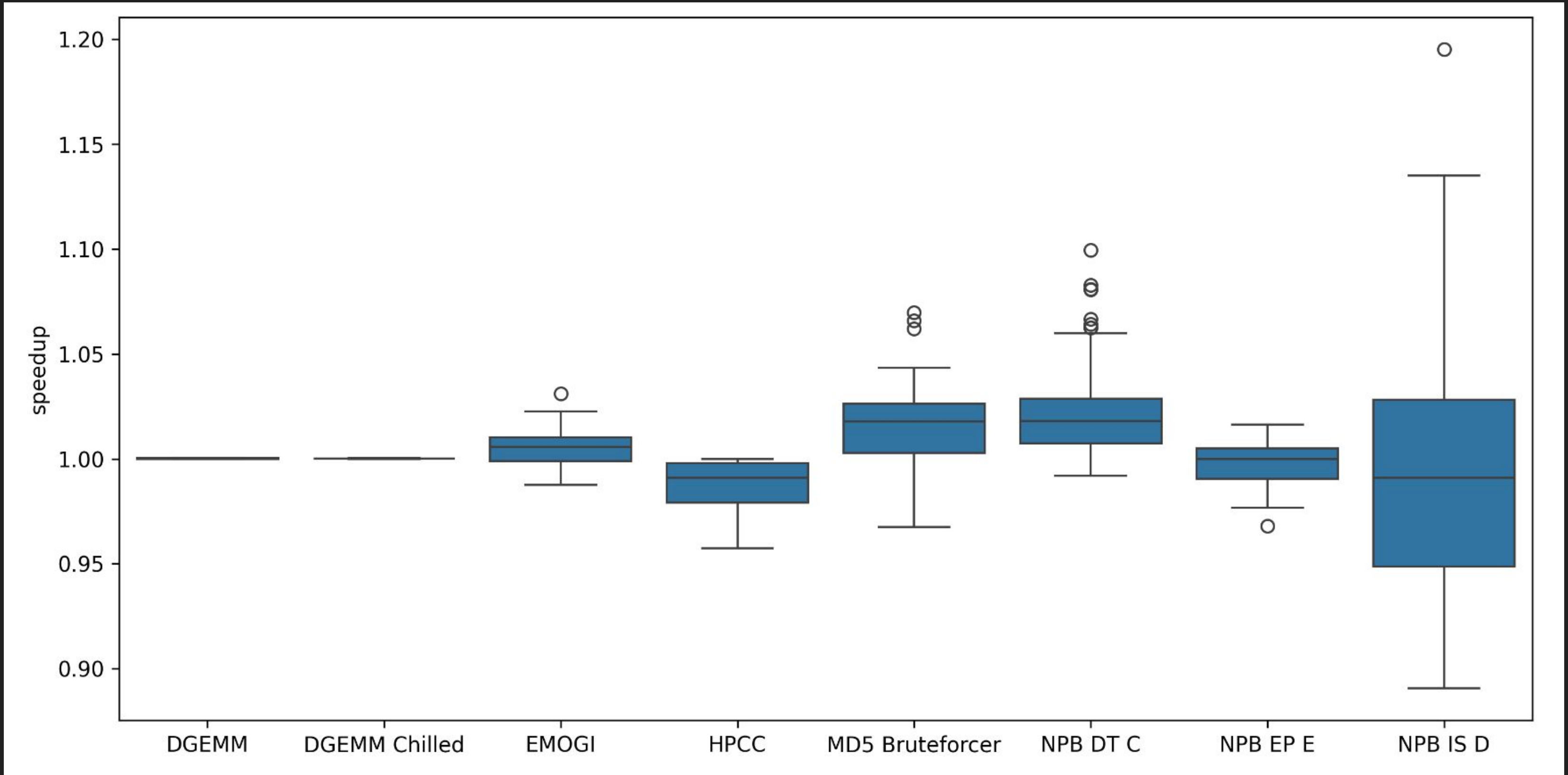


Compute Used > Device Used

- Lower values = faster heat accumulation



Little Performance Deviations



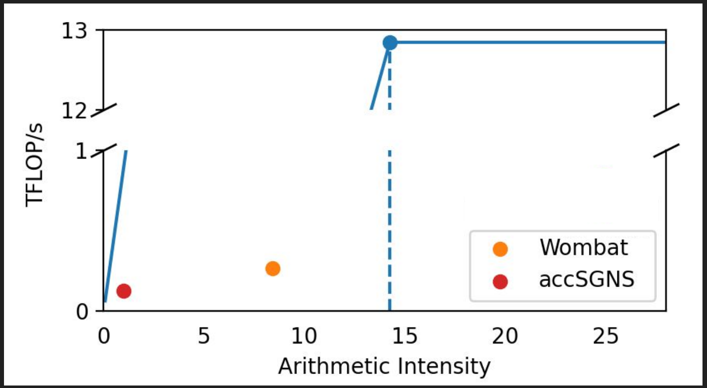
Energy Efficiency Not Guaranteed

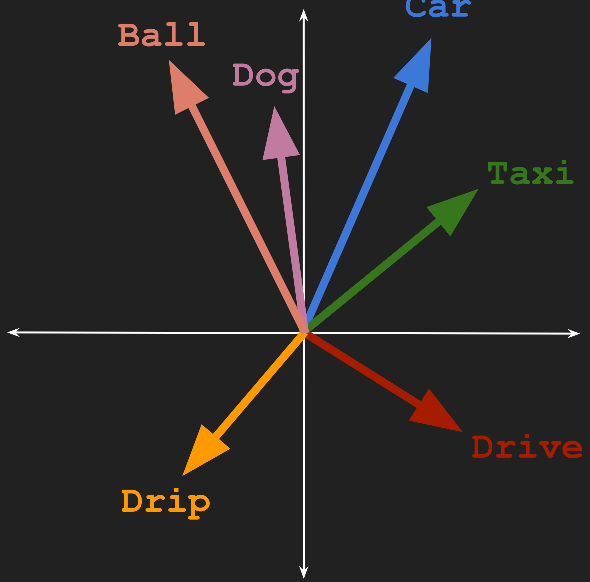
- Passive cycles **waste power**
  - +15 kJ CUDA Stream
  - +19 kJ HPL
- Peak dissipation **more power efficient**
  - -4 kJ CUDA DGEMM
  - -7 kJ NPB EP
- Known theoretically [1]
  - +1.6x heat capacity, +6.5x thermal conductivity
  - +100x viscosity, +693x denser
- Tuning **required**

[1] Jimil M. Shah et al. “Evaluating the Reliability of Passive Server Components for Single-Phase Immersion Cooling” ASME Journal of Electronic Packaging Volume 144 Issue 2

Why Optimize Word2Vec?

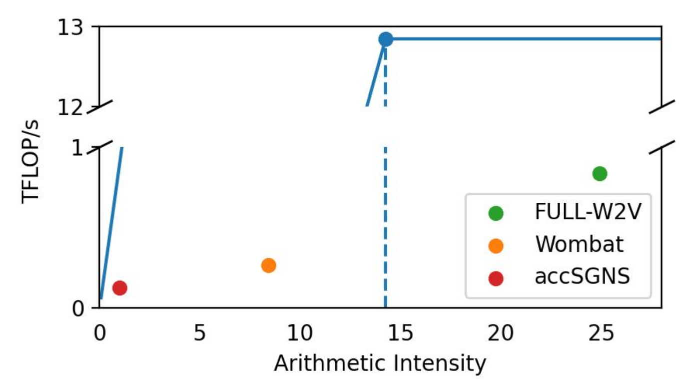
- Repeated usage
  - Larger dataset → richer model
  - Pretraining not always enough
  - Language evolution: more training
- Ripe for GPU acceleration
  - Embarrassingly parallel
  - Simple network architecture
  - … CPUs were faster!!





Suboptimal GPU Performance

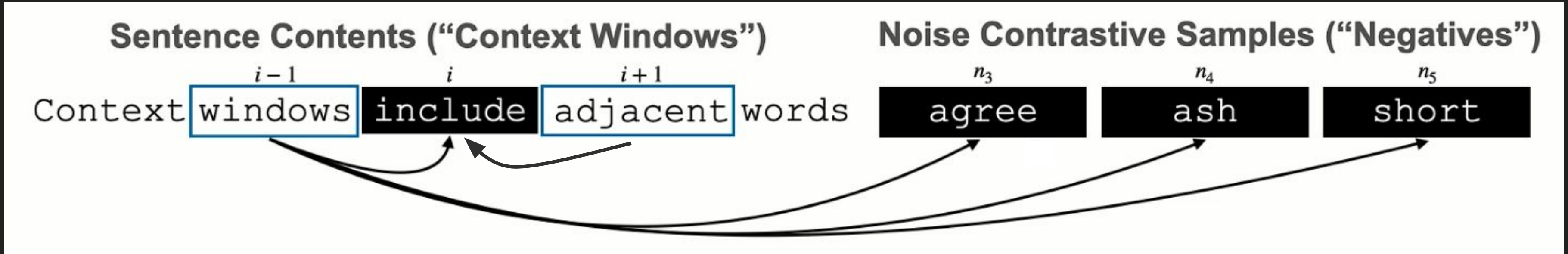
- Optimizations for K40 GPUs (2013) failed to continue scaling performance
- CPUs maintained general performance advantage



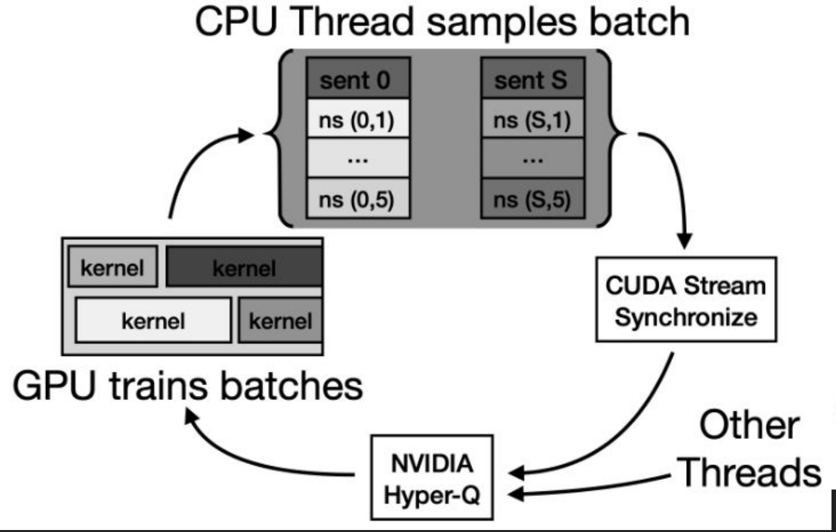
Hardware Platform	pWord2Vec (CPU) Millions Words/Sec	Wombat (GPU) Millions Words/Sec
Broadwell CPU, P100 GPU (2016)	**10.36**	2.86
Haswell CPU, TitanXP GPU (2017)	8.4	3.3
Skylake CPU, V100 GPU (2018)	9.32	10.33

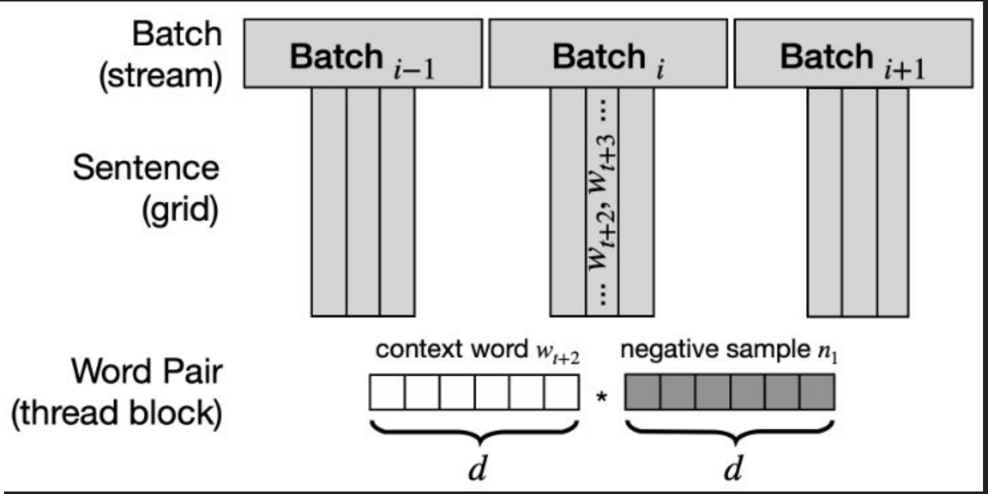
Bird’s Eye View of Algorithm

- Text is decomposed into “sentences”
  - Embarrassingly parallel
- Sentences are composed of context windows
  - Words close enough for association
  - Serial order necessary for convergence and correctness
- Noise-contrastive samples (negatives) for each context window
  - Positively correlate words within the window
  - Negatively correlate spurious selection of words

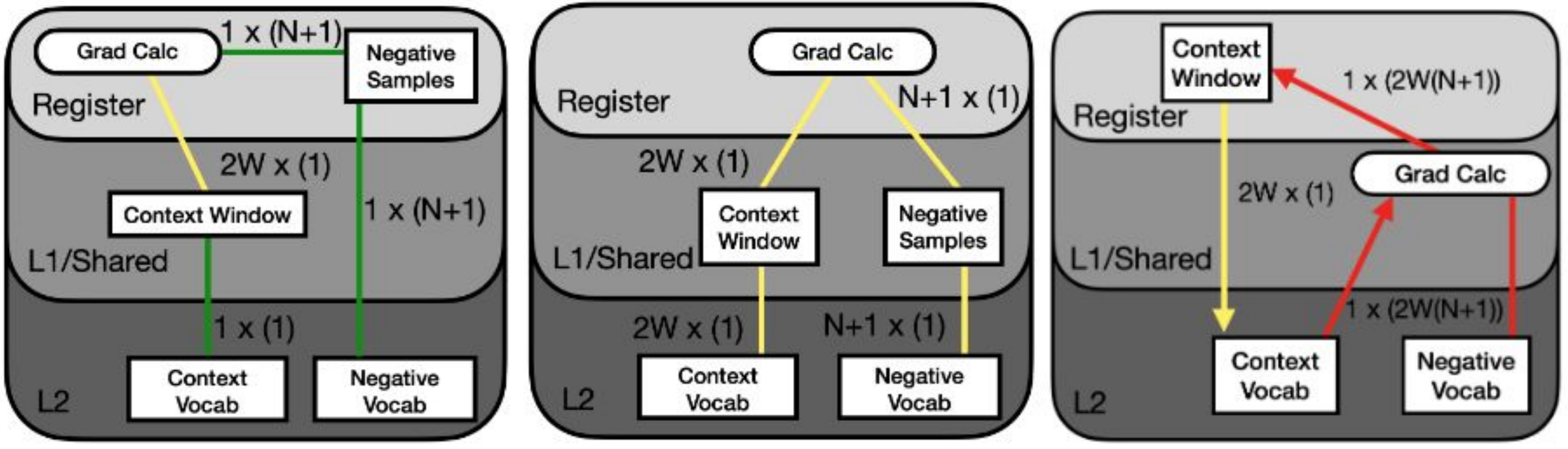


FULL-W2V Coordination and Decomposition





Comparing Memory Demand



(a) FULL-W2V (b) Wombat (c) accSGNS

In Depth Word2Vec Memory Demand

- Gigabytes-per-epoch
- Primarily reduce L1 demand
  - Reuse in shared not reported by tool (all implementations)

Implementation	L1/TEX	L2	DRAM	Sum
FULL-W2V	**94.760**	**88.723**	**41.851**	**225.334**
FULL-Register	885.065	781.576	66.555	1,733.196
accSGNS	1,134.448	493.614	226.578	1,854.640
Wombat	2,303.525	1,432.774	45.799	3,782.098

Word2Vec Evaluation Platforms

- Span multiple CPU and GPU generations

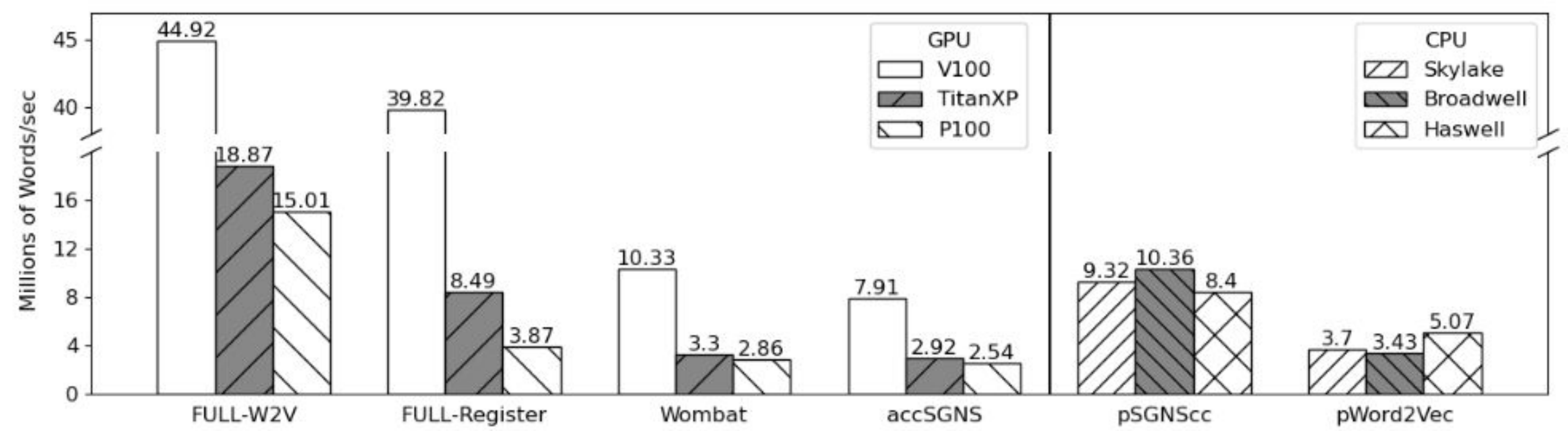
Hardware	GPU Specs	CPU Specs
GPU: V100 Gen-6 Volta CPU: Xeon Gold 6148 Gen-6 Skylake	80 SMs 14 TFLOP/s 16 GB HBM2 900 GB/s 4 Warp Schedulers	2 20-core CPUs 2.40 GHz 27.5 MB L3
GPU: Titan XP Gen-5 Pascal CPU: Xeon E5-2670 v3 Gen-4 Haswell	60 SMs 12.15 TFLOP/s 12 GB GDDR5x 548 GB/s 2 Warp Schedulers	2 12-core CPUs 2.30 GHz 30 MB L3
GPU: P100 Gen-5 Pascal CPU: Xeon E5-2680 v4 Gen-5 Broadwell	56 SMs 9.3 TFLOP/s 12 GB HBM2 549 GB/s 2 Warp Schedulers	2 14-core CPUs 2.40 GHz 35 MB L3

Word2Vec Datasets and Evaluations

- Text8 benchmarks throughput over 20 epochs
- One Billion Words benchmarks quality after 5 epochs
- Quality measured by
  - Spearman's rank correlation coefficient WS-353 and SimLex-999
  - Hyperwords analogy scores

Corpus	Vocabulary Size	Words/Epoch	Sentences
Text8	71,291	16,718,845	17,006
One Billion Words	555,514	804,269,957	30,607,795

Throughput on One Billion Words Corpus



Why Word2Vec Issues Better

- Better warp availability
  - Max active warps is 16 on both XP and V100

	XP Architecture				V100 Architecture			
	Wombat	accSGNS	FULL-Register	FULL-W2V	Wombat	accSGNS	FULL-Register	FULL-W2V
Max Warps	11.03	12	16	13	11.03	12	16	9
Active Warps	4.59	11.08	15.86	9.59	4.66	9.41	14.92	8.99
Eligible Warps	0.16	1.33	0.42	0.99	0.18	1.09	1.86	1.90

- Higher IPC / fewer stalls
  - Only comparable between HIGHLY similar implementations

	XP Architecture		V100 Architecture	
	FULL-Register	FULL-W2V	FULL-Register	FULL-W2V
IPC	1.19	2.78	2.38	3.22
Long Scoreboard	38.66	1.25	11.00	0.97
Short Scoreboard	4.49	3.43	4.19	2.95
Arithmetic	0.16	0.18	1.14	0.66
Overhead	13.05	10.48	7.93	6.35

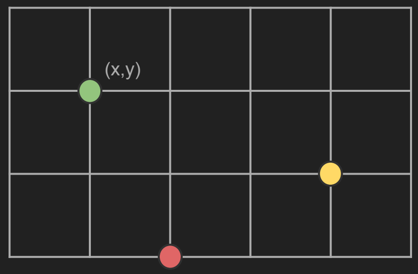
Minimal Effects on Embedding Quality

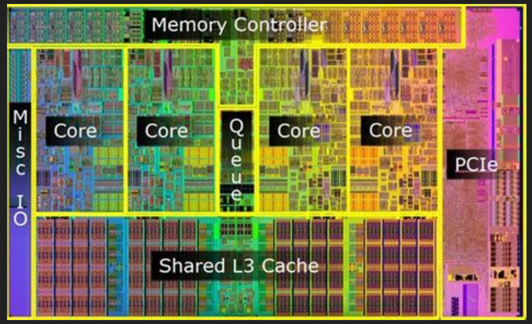
- One Billion Words corpus
- Average over 5 repeated trials

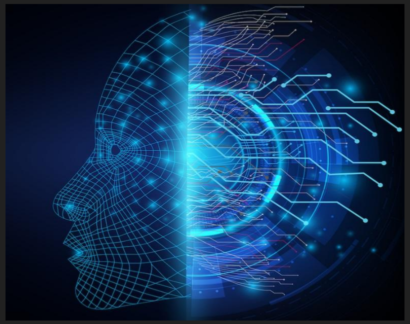
	WS-353	SimLex-999	COS-ADD	COS-MUL
pWord2Vec	0.6070	0.3499	29.895%	29.166%
Wombat	0.5952	0.3596	29.661%	28.988%
FULL-W2V	0.5923	0.3582	29.775%	29.386%

Existing Approaches

- Grid search: simple
  - Fast and imprecise
  - Precision costs speed
- System modeling: generalize
  - Costly one-time setup
  - Inflexible
- Application-based ML: specialize
  - Need lots of data
  - High quality results
- Transfer learning: re-use knowledge
  - Reduce needed data







Practical Example of Tuning Costs

	Input Scale		
	Small	Medium	Large
Input Scale Characteristics			
Array Dimensions	≤ 80	≤ 220	≤ 1200
Naive Tera-Ops	0.037	4.75	2924.24
Worst Runtime (s)	0.00017	0.1096	9.8631
Best Configuration Values			
Packed Arrays	A,E,F	F	A,B,E
Loop Interchanges	N/A	N/A	Outer Exchange
Tile Sizes	16, 2048, 4	96, 16, 4	4, 2048, 4
Speedup Over Default	1.13×	14.94×	50.50×

Rejection Sampling Costly

- Other generative techniques lack conditional sampling at considerable cost to efficiency

Method	Time (s)	Sample Acceptance Rate	Reject Reason (%)	
			Repeated	Ill-Conditioned
Random	0.24	90.8%	9.2%	0%
GC	0.52	37.87%	62.13%	0%
CTGAN	1.28	4.8%	7.33%	87.87%
TVAE	80.77	0.05%	2.25%	97.70%

Exploratory surrogate models

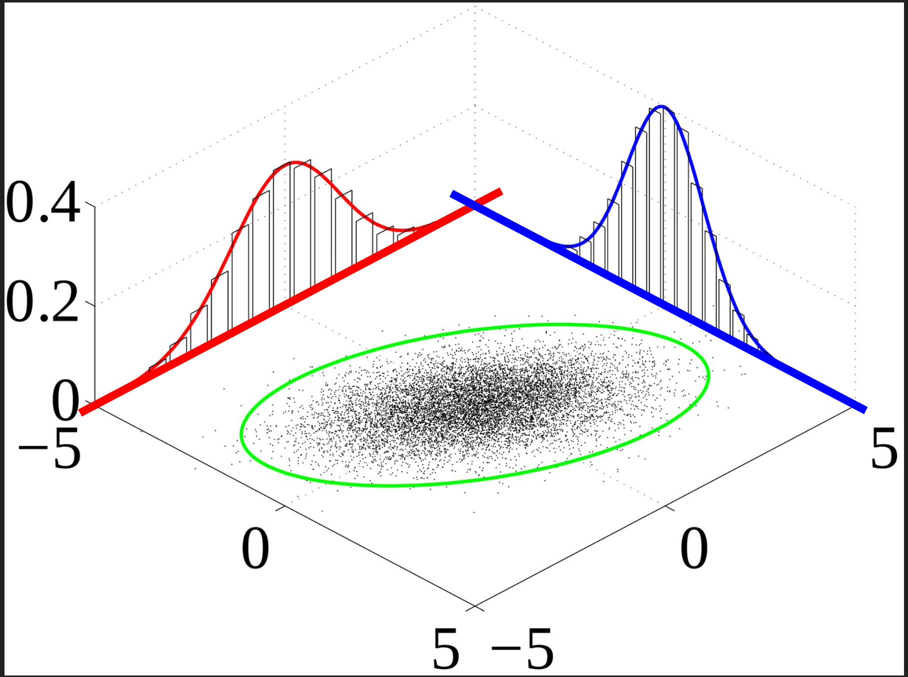
- BLISS, YTOPT, GPTune
- Surrogate represents known information and trend
  - Uncertainty measure
  - Iteratively explore & exploit
- Powerful efficiency, but restart from scratch
  - Need to leverage known data

Shortcomings of Prior TL

- Regression requires new data for calibration
  - Expensive restart
  - Random collection
- Machine-learning scales to **BIG DATA**
  - Desirable to use minimal data
  - Long-term convergence too slow
  - Limited improvement
- Primary gap:
  - Simple
  - Aggressive
  - Transferrable

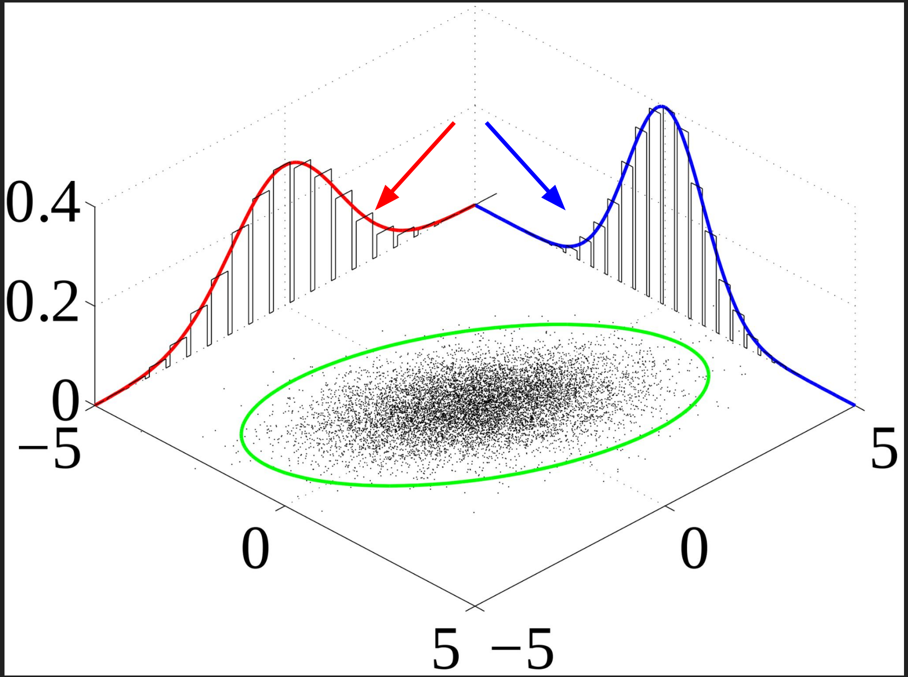
GC Model

- Multivariate probability distribution



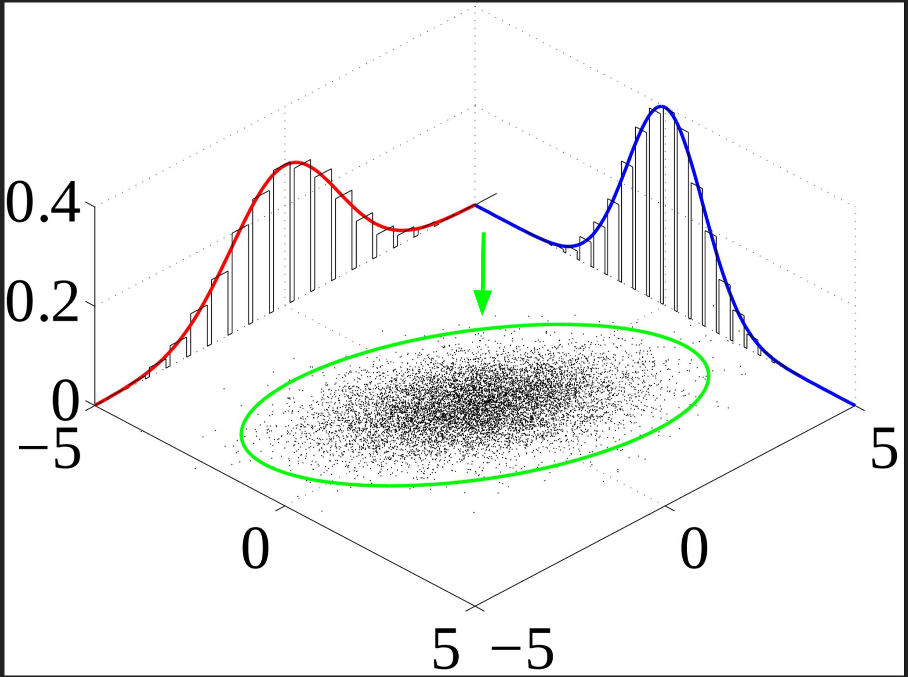
GC Model

- Multivariate probability distribution
- Components
  - Disjoint marginal per variable



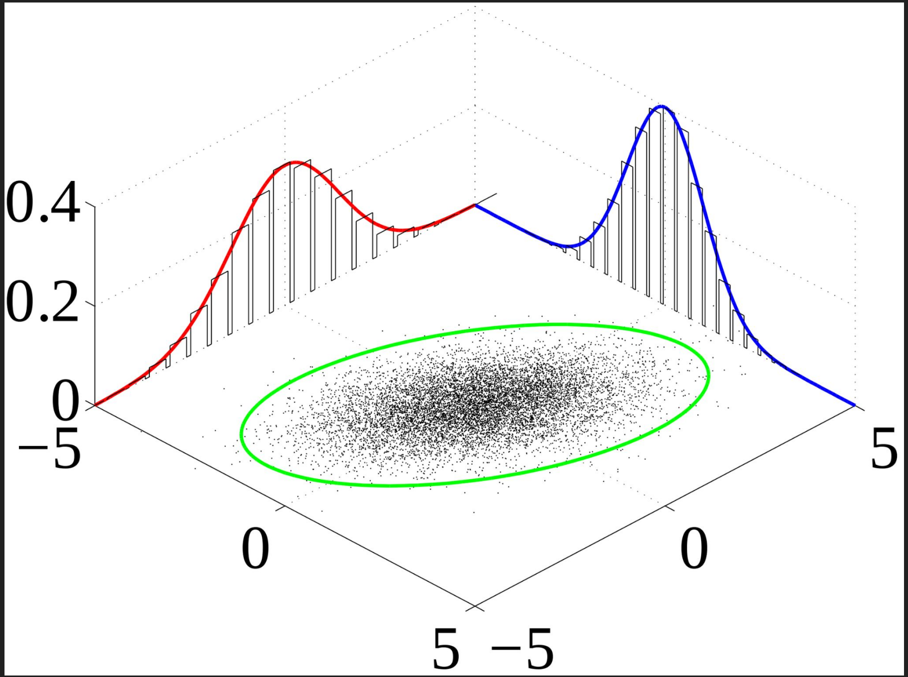
GC Model

- Multivariate probability distribution
- Components
  - Disjoint marginal per variable
  - Correlations as joint distribution



GC Model

- Multivariate probability distribution
- Components
  - Disjoint marginal per variable
  - Correlations as joint distribution
- Capabilities
  - Samples ↔ Distributions
  - Conditional sampling



“Good” Distribution from Filtered Data

- Need **limited** tuning space coverage
  - |generable| / |space|
  - **Reduce**, not eliminate

Filtering Quantile (%)	Tuning Space Coverage
100	1.00
90	1.00
80	1.00
70	1.00
60	0.91
50	0.91
40	0.91
30	0.82
20	0.07
10	0.06

“Good” Distribution from Filtered Data

- Need **limited** tuning space coverage
  - |generable| / |space|
  - **Reduce**, not eliminate
- Specificity matches optimal area
  - Minimize divergence
    - Top-10% optimal configs
    - Top-X% training data
  - Lower divergence = better match

Filtering Quantile (%)	Tuning Space Coverage	KL Divergence
100	1.00	0.1878
90	1.00	0.1713
80	1.00	0.1609
70	1.00	0.1525
60	0.91	0.1409
50	0.91	0.1212
40	0.91	0.1333
30	0.82	0.1713
20	0.07	0.2766
10	0.06	0.3079

Filtering: Out with the Bad

- Filter source data via observed quantiles
  - Remove poor features

Filtering Quantile (%)	Tuning Space Coverage	KL Divergence
100	1.00	0.1878
90	1.00	0.1713
80	1.00	0.1609
70	1.00	0.1525
60	0.91	0.1409
50	0.91	0.1212
40	0.91	0.1333
30	0.82	0.1713
20	0.07	0.2766
10	0.06	0.3079

Filtering: Preserve Sufficient Coverage

- Filter source data via observed quantiles
  - Remove poor features
- Careful! Do not filter too much!
  - Keep top 15+%

Filtering Quantile (%)	Tuning Space Coverage	KL Divergence
100	1.00	0.1878
90	1.00	0.1713
80	1.00	0.1609
70	1.00	0.1525
60	0.91	0.1409
50	0.91	0.1212
40	0.91	0.1333
30	0.82	0.1713
20	0.07	0.2766
10	0.06	0.3079

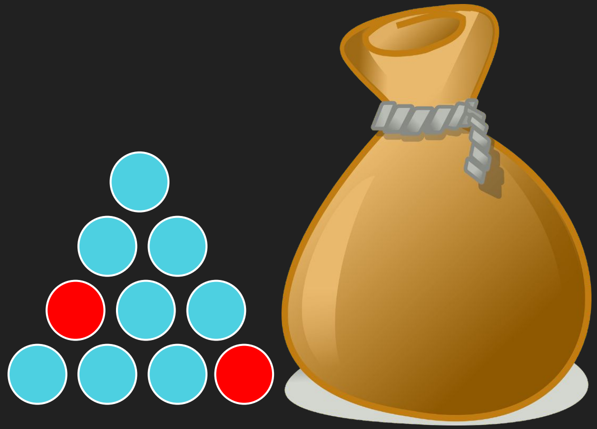
Filtering: Empirical Ideal

- Filter source data via observed quantiles
  - Remove poor features
- Careful! Do not filter too much!
  - Keep top 15+%
- Suggest top 30%
  - Sufficient but minimized space coverage
  - Divergence not increasing too much

Filtering Quantile (%)	Tuning Space Coverage	KL Divergence
100	1.00	0.1878
90	1.00	0.1713
80	1.00	0.1609
70	1.00	0.1525
60	0.91	0.1409
50	0.91	0.1212
40	0.91	0.1333
30	0.82	0.1713
20	0.07	0.2766
10	0.06	0.3079

Budget Estimation: Probability of Success

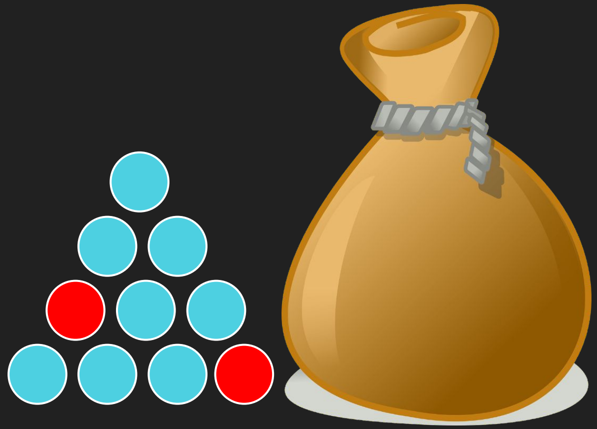
- Hypergeometric sampling (blind marble picking):
  - |C| configurations (marbles)
    - |I| near-optimal (red marbles)
  - Up to k samples



$$P(\#Optimal \geq 1) = \sum_{i=1}^{k} \frac{\binom{|I|}{i}\binom{|C|-|I|}{k-i}}{\binom{|C|}{k}}$$

Budget Estimation: Probability of Success

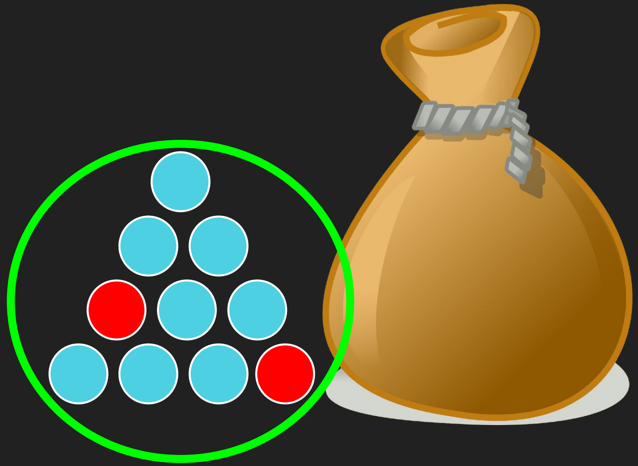
- Hypergeometric sampling (blind marble picking):
  - $|C|$ configurations (marbles)
    - $|I|$ near-optimal (red marbles)
  - Up to k samples



$$P(\#Optimal \geq 1) = \sum_{i=1}^{k} \frac{\binom{|I|}{i}\binom{|C|-|I|}{k-i}}{\binom{|C|}{k}}$$

Budget Estimation: Probability of Success

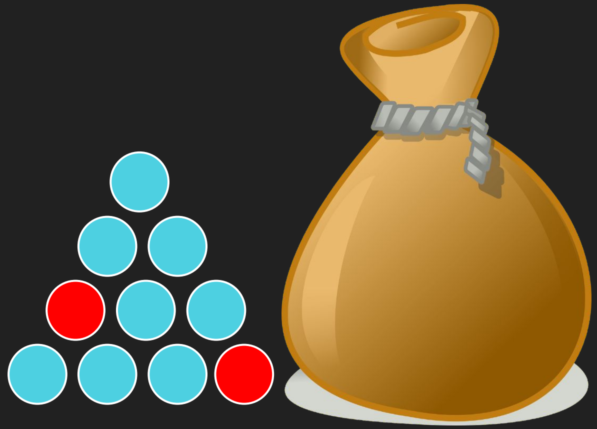
- Hypergeometric sampling (blind marble picking):
  - $|C|$ configurations (marbles)
    - $|I|$ near-optimal (red marbles)
  - Up to k samples



$$P(\#Optimal \geq 1) = \sum_{i=1}^{k} \frac{\binom{|I|}{i}\binom{|C|-|I|}{k-i}}{\binom{|C|}{k}}$$

Budget Estimation: Probability of Success

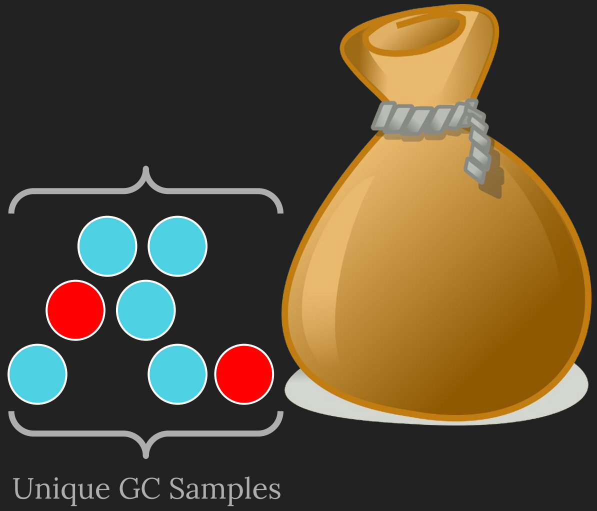
- Hypergeometric sampling (blind marble picking):
  - $|C|$ configurations (marbles)
    - $|I|$ near-optimal (red marbles)
  - Up to k samples
- Incomplete coverage from GC
  - Remove marbles before sampling!



$$P(\#Optimal \geq 1) = \sum_{i=1}^{k} \frac{\binom{|I|}{i}\binom{|C|-|I|}{k-i}}{\binom{|C|}{k}}$$

Budget Estimation: Probability of Success

- Hypergeometric sampling (blind marble picking):
  - |C| configurations (marbles)
    - |I| near-optimal (red marbles)
  - Up to k samples
- Incomplete coverage from GC
  - Remove marbles before sampling!
- Probability estimation
  - Unique GC samples are proxy for |C|
  - Estimate reduction in |I|



$$P(\#Optimal \geq 1) = \sum_{i=1}^{k} \frac{\binom{|I|}{i}\binom{|C|-|I|}{k-i}}{\binom{|C|}{k}}$$

Experiment Design

- Evaluation Platform: ANL LCRC Swing Cluster
  - 2✕ AMD EPYC 7742 (64-core; 128-logical)
  - 1✕ 40 GB NVIDIA A100
  - Clang with Polly LLVM loop optimizer
- Each application source sizes:
  - Bayesian Optimization with Random Forest
  - 200✕ each for Small, Medium, Large
- Each application target sizes:
  - 30✕ each for Small-Medium, Medium-Large, Extra-Large

Benchmark	#Params	# Configurations
3mm	10	376,320
Covariance	5	5,324
Floyd–Warshall	5	5,324
Heat3d	6	10,648
LU	5	5,324
Syr2k	6	10,648
AMG	9	1,180,980
RSBench	9	5,196,312
XSBench	8	577,368
SW4Lite	8	4,752

Tuning Spaces and GC Coverage

- Coverage represents reasonable # unique samples attainable
- Budget based on hypergeometric sampling with 5% regret and 95% confidence in 1+ samples in top-10%

Benchmark	#Params	# Configurations	GC Coverage	GC Budget
3mm	10	376,320	$\approx$ 2,500	–
Covariance	5	5,324	$\approx$ 110	–
Floyd–Warshall	5	5,324	$\approx$ 1,800	15
Heat3d	6	10,648	$\approx$ 1,600	8
LU	5	5,324	$\approx$ 210	–
Syr2k	6	10,648	$\approx$ 800	3
AMG	9	1,180,980	$\approx$ 108,500	5
RSBench	9	5,196,312	$\approx$ 316,800	3
XSBench	8	577,368	$\approx$ 77,500	7
SW4Lite	8	4,752	$\approx$ 1,800	15

Compared Approaches

- Baseline
  - Parameters derived from original source
  - Reference for speedup
- Bayesian Optimization (BO)
  - From scratch without TL; same settings as training dataset
- All TL use the same prior dataset from BO
  - GPTune DTLA
    - SOTA TL autotuner using Gaussian Processes
  - GC-TLA (ours)
    - Fit to top-30% source data; conditionally sample for TL

Tuned Parameters for Input-Scaling GC

- Polybench

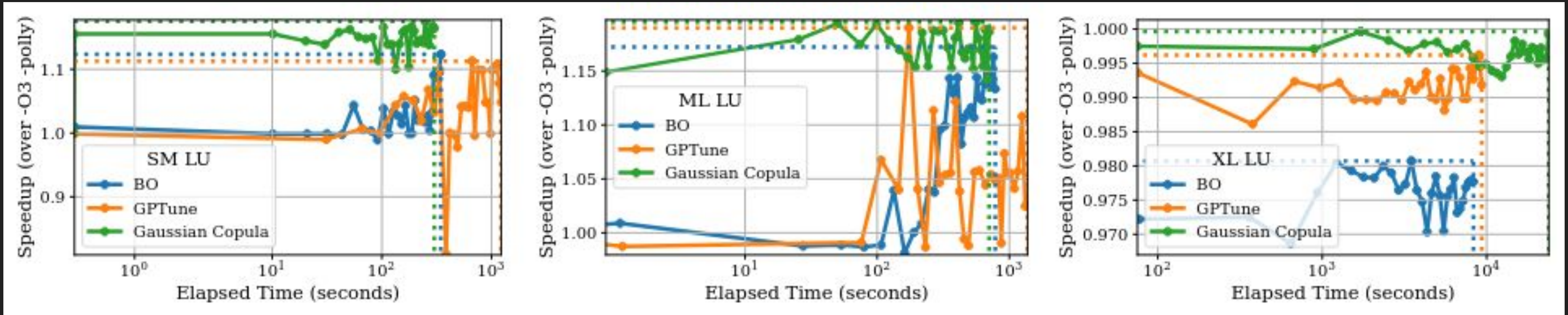
Parameter	3MM	Covariance	Floyd-Warshall	Heat3d	LU	Syr2k
Tile Sizes	[4-2048], [4-2048], [4-2048]	[4-128], [4-2048], [4-256]	[4-128], [4-2048], [4-256]	[4-128], [4-2048], [4-256]	[4-128], [4-2048], [4-256]	[4-128], [4-2048], [4-256]
Loop Interchange	[Yes, N/A]	[Yes, N/A]	[Yes, N/A]	[Yes, N/A]	[Yes, N/A]	[Yes, N/A]
Array Packing	[Yes, N/A] × 6	[Yes, N/A]	[Yes, N/A]	[Yes, N/A] × 2	[Yes, N/A]	[Yes, N/A] × 2

- ECP

Parameter	AMG	RSBench	XSBench	SW4Lite
Tile Sizes	[10-200], [2-256], [2-256], [10-200]	[2-256], [2-256]	[2-256], [2-256]	—
Optional Parameters	Parallel For	Parallel For	Parallel For	Parallel For, Nowait, MPI_Barrier
Parallel For Schedule	—	[100-2000], [10-200]	[10-160]	[dynamic, static]
Unrolling Options	[unroll, N/A]	[unroll, N/A]	[unroll, N/A]	[unroll (6) [1], unroll, no-unroll]
# Threads	[4-8]	[2-256]	[2-256]	[2-256]
KMP Affinity	[compact, scatter, balanced]	[compact, scatter, balanced, none, disabled, explicit]	[compact, scatter, balanced, none, disabled, explicit]	—
OMP Proc Bind	—	—	—	[close, spread, master]
OMP Places	[core, threads, sockets]	[core, threads, sockets]	[core, threads, sockets]	[core, threads, sockets]

Some Polybench Spaces are Difficult

- GC still comparatively excels



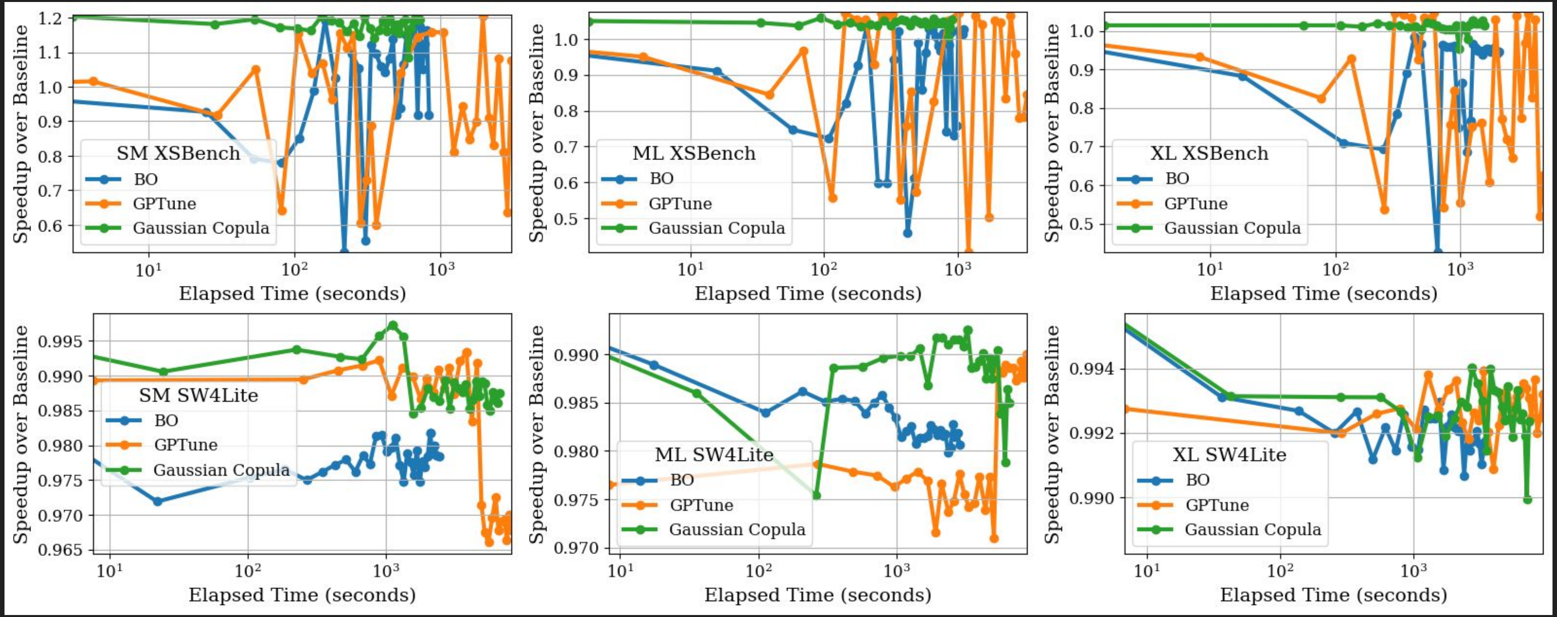
ECP Demonstrates Sophistication

- Speedup is difficult!!
- GC's best results achieved on-budget
- GC continues to succeed with complex spaces
- Worst margin of performance is -0.02✕ speedup

App.	Scale	Peak Speedup (# Evaluation Discovered)				
		GC			BO	GPTune
		1^{st}	Budget	Best	Best	Best
AMG	SM	0.87	0.91 (3)	0.91 (3)	0.92 (19)	0.90 (19)
	ML	0.93	0.93 (1)	0.93 (1)	0.93 (20)	0.87 (3)
	XL	0.95	0.95 (5)	0.98 (23)	0.97 (27)	0.93 (25)
RSBench	SM	1.40	1.40 (3)	1.40 (8)	1.25 (29)	1.13 (22)
	ML	1.02	1.04 (2)	1.04 (15)	0.97 (22)	1.04 (27)
	XL	1.00	1.00 (1)	1.01 (10)	0.97 (14)	1.02 (18)
XSBench	SM	1.20	1.20 (7)	1.21 (28)	1.17 (24)	1.21 (24)
	ML	1.05	1.06 (4)	1.06 (4)	1.04 (6)	1.07 (5)
	XL	1.01	1.02 (5)	1.03 (24)	0.99 (6)	1.05 (5)
SW4Lite	SM	0.99	1.00 (6)	1.00 (6)	0.98 (26)	0.99 (17)
	ML	0.99	0.99 (10)	0.99 (16)	0.99 (3)	0.99 (30)
	XL	0.99	0.99 (12)	0.99 (12)	0.99 (1)	0.99 (14)

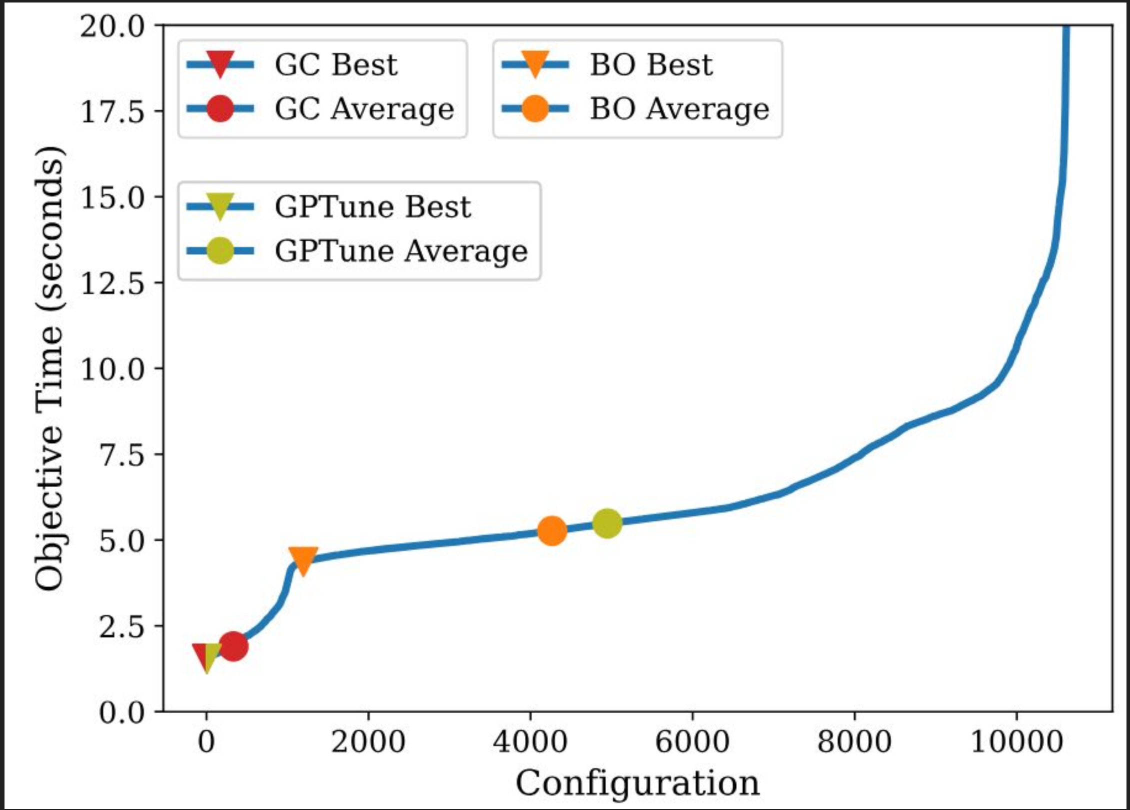
Continued Success with Greater Complexity

- Better budget result in less time than prior work



Exhaustive Quality on Syr2k XL Benchmark

- Higher average quality than GPTune
- Same high-quality results
- Immediate access



Extending to Multi-Scale Tuning

- Usually larger problems are handed to larger-scale systems
  - Weak scaling
- Greater complexity in tuning
  - Search space **grows** as hardware changes
  - More performance inflection points
  - Higher co-dependency between parameters

Multi-Scale Benchmark

- Highly Efficient Fast Fourier Transform for Exascale
  - Leverage many GPUs
  - Network bottleneck
  - Performance tuning **required**
- Desire maximum **throughput**
- Scale 2→32 Nodes (4 GPUs/Node)
  - 9.6k → 124.4k configurations
  - Weak-scaled FFT volume
  - Learn hardware & task size **simultaneously**

Multi-Scale Tuning Tasks

- Performance range in GFLOP/s
  - Spans all evaluations from all techniques

# Nodes	Total GPUs	# MPI Ranks	FFT Dimensions			Worst Known Performance	Best Known Performance
			X	Y	Z		
2	8	8	256	256	256	53.7	1,308
4	16	16	256	256	512	86.0	2,233
8	32	32	256	512	512	160.5	4,170
16	64	64	512	512	512	174.6	8,664
32	128	128	512	512	1024	146.9	12,018

heFFTe Tuning Space

- Precision
  - Single, double
- Reordering
  - Enabled, disabled
- MPI Communcation Strategy
  - All-to-All, All-to-All-v, Peer-to-Peer, Peer-to-Peer with Pipelining
- Reshaping
  - Pencils, slabs
- Conversion
  - Complex / Real
- **MPI Topology**
  - Hardware dependent, but VIRTUAL implementation

Full Multi-Scale Results

- GCTLA typically exceeded by GCTLA/BO random
    - GPTune's best also frequently random

Target Task	GCTLA Peak Relative to BO Peak	BO Trials to Exceed Best GCTLA	GPTune Peak Relative to BO Peak	BO Trials to Exceed Best GPTune
2 Nodes	39.7%	1	94.4%	32
4 Nodes	54.2%	5	60.1%	5
8 Nodes	38.6%	1	66.0%	14
16 Nodes	34.6%	5	60.0%	13
32 Nodes	86.6%	59	145.3%	N/A