IMPERIAL

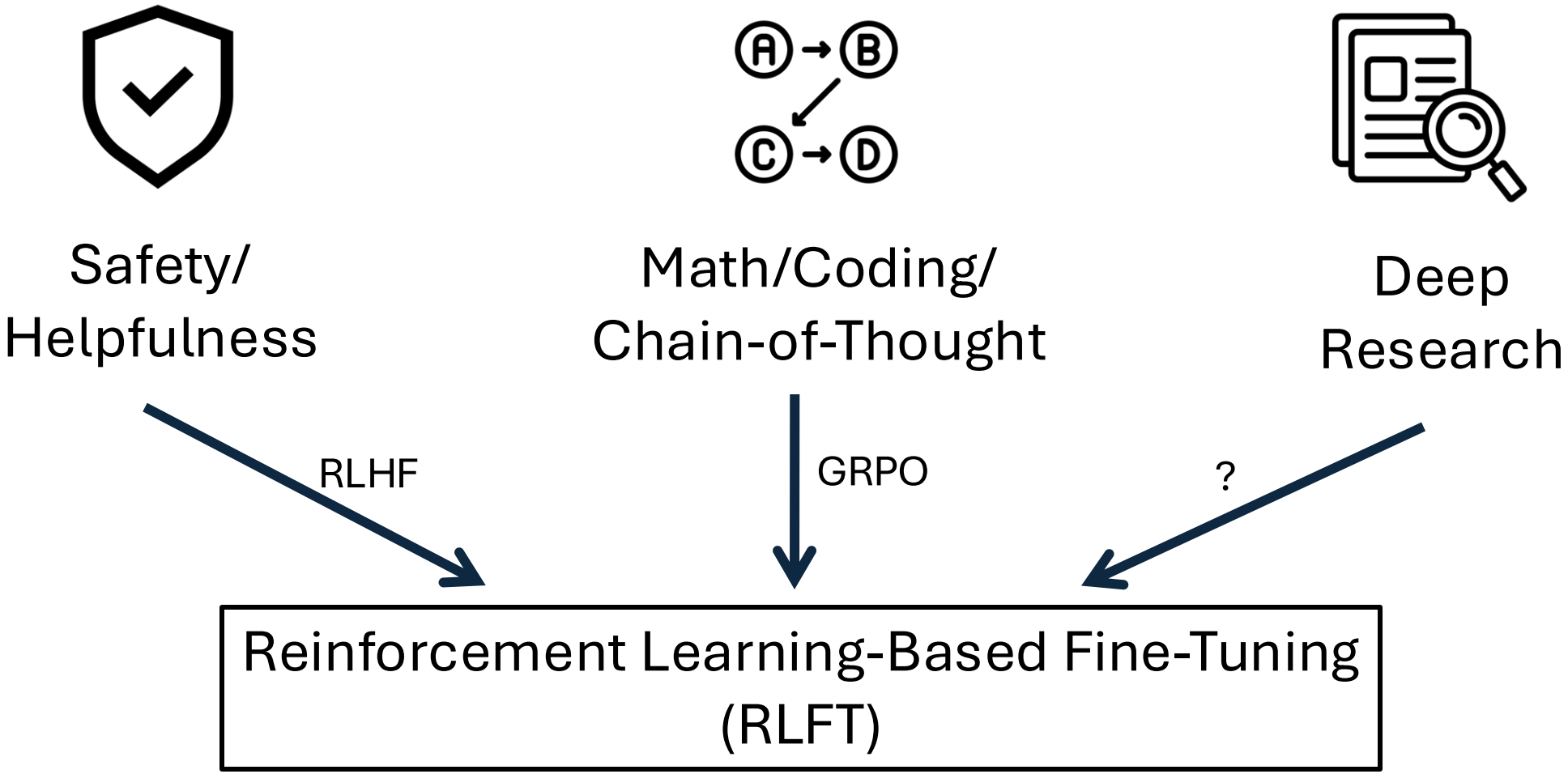
LSDS
Large-Scale Data & Systems Group

Systems Opportunities for LLM Fine-Tuning using Reinforcement Learning

Pedro F. Silvestre, Peter Pietzuch
pms20@ic.ac.uk

Imperial College London
Large Scale Data & Systems (LSDS) Group

Recent Explosion in LLM Capabilities

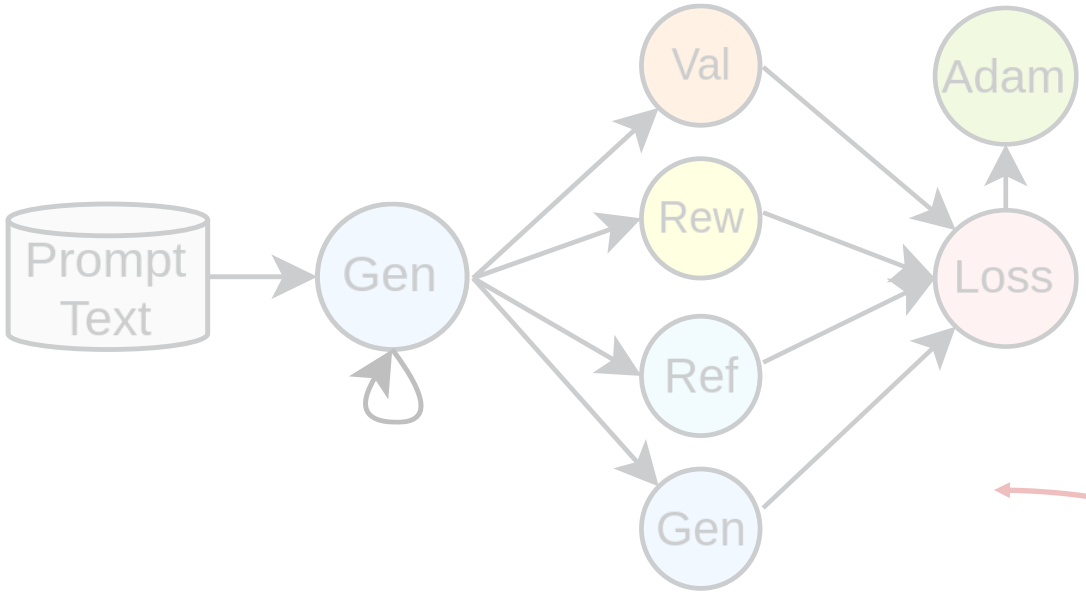


RLFT Generates Variable-Size Data

Context

Whe	re		is		Rot	ter	dam	?	The		Net	her	lan	ds	<STOP>

Variable-Size



Large-Scale

1. Sequential **Generation**

2. Parallel **Inference**

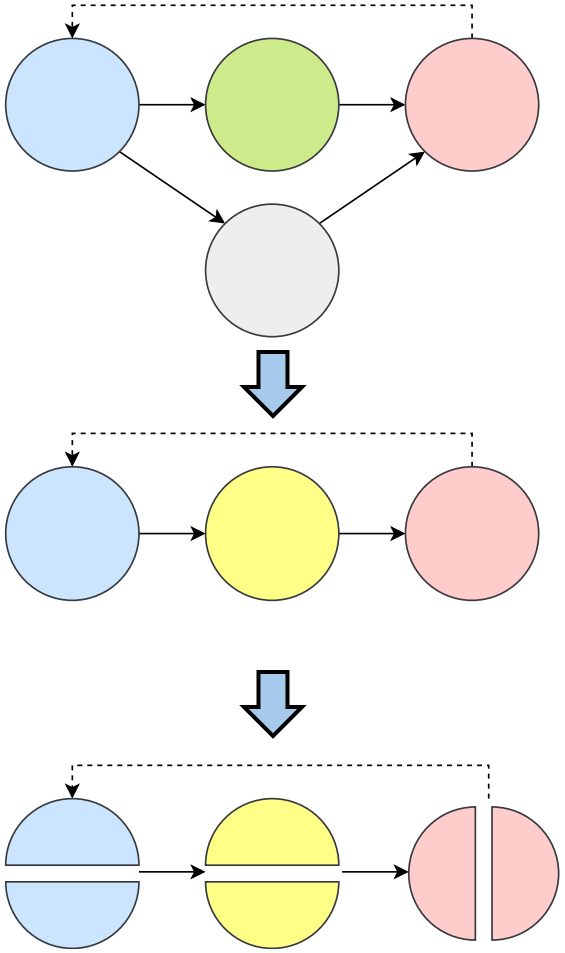
3. Backward + **Training**

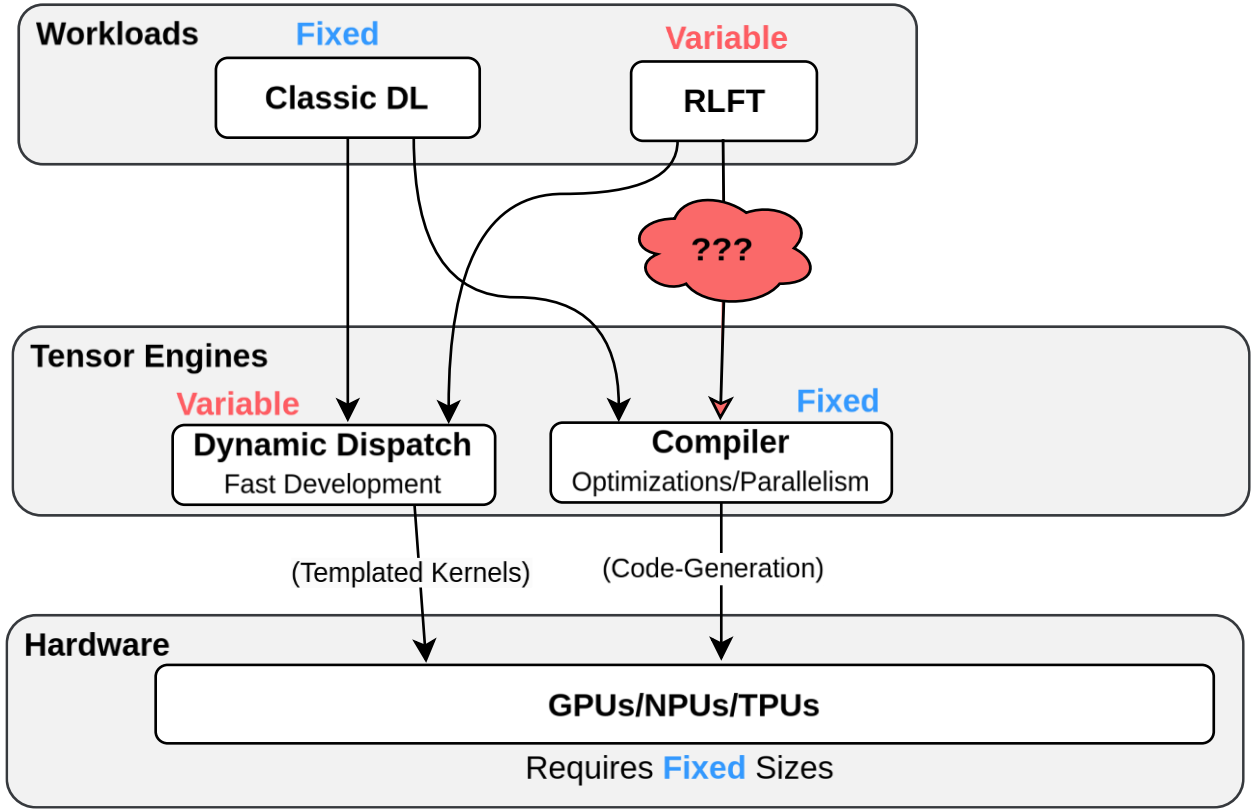
Tension Between **Large-Scale** and **Variable Size**

Computation Graph

Optimizations

Parallelization/ Distribution





Workarounds for Mapping RLFT To Compilers

Undershoot: Error

Overshoot: Overhead

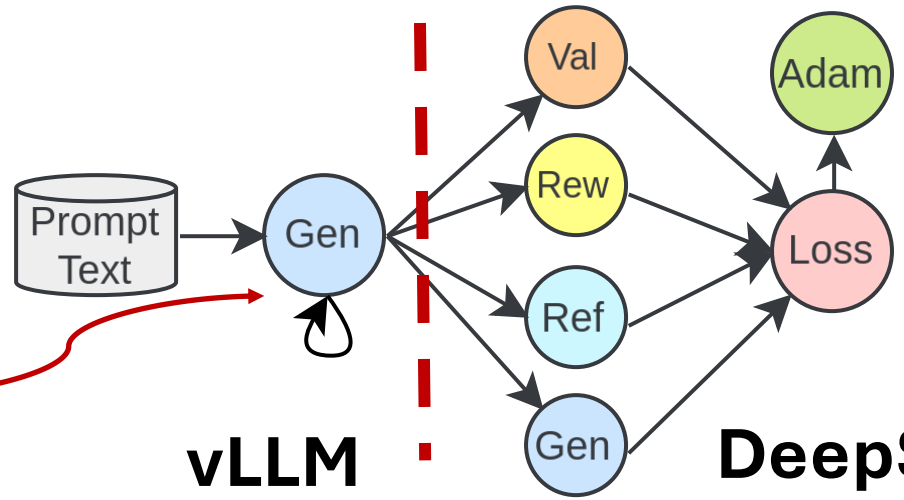
1. Pad to fixed MAX_LEN

Whe	re		is		Rot	ter	dam	?	The		Net	her	lan	ds	<STOP>	pad	pad	pad	pad	pad	pad	pad	pad

2. Mask valid steps *

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-inf	-inf	-inf	-inf	-inf	-inf	-inf	-inf

3. Graph Breaks



Pad/mask every step?

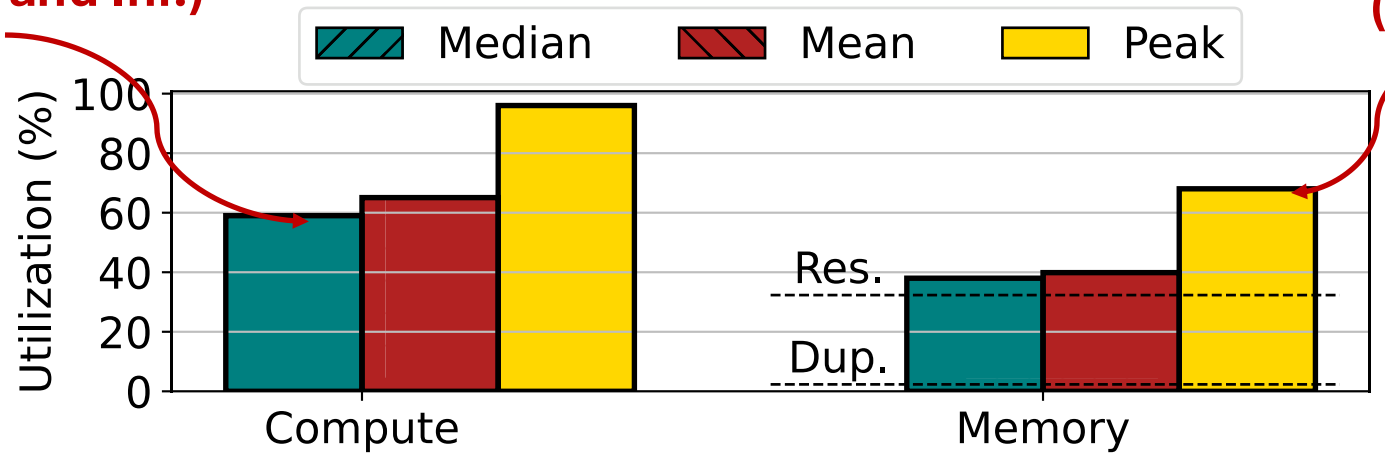
Manual Work!

Not Holistic!

TRL GRPO Benchmark on A100

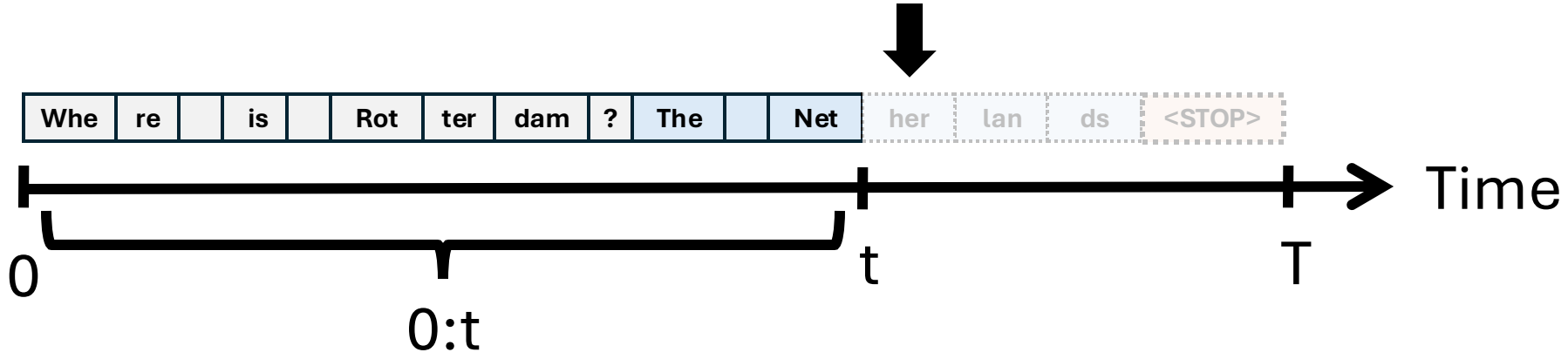
GPU underutilization
(No Overlap of Gen. and Inf.)

BS=2 exceeds memory
(Large Peak Memory)

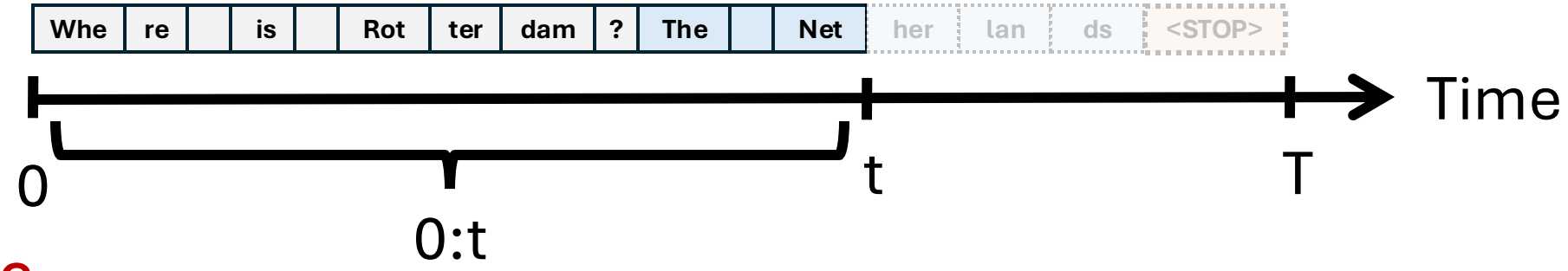


Need Better Graph Representations for RLFT!

Key Insight: Represent Symbolically

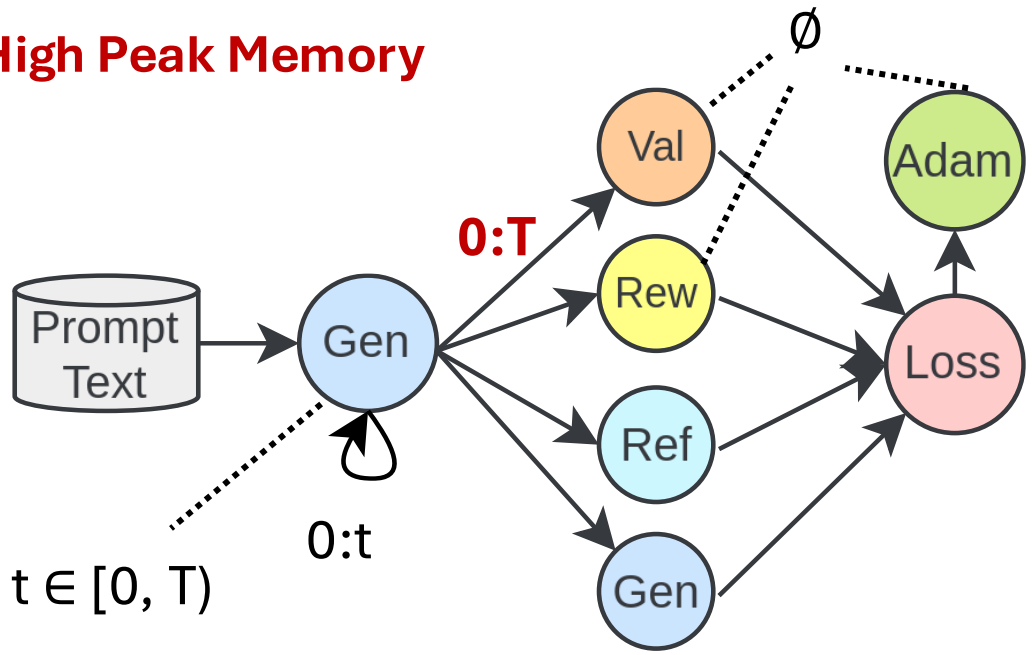


Key Insight: Represent Symbolically



No Overlap w\ Gen.

High Peak Memory



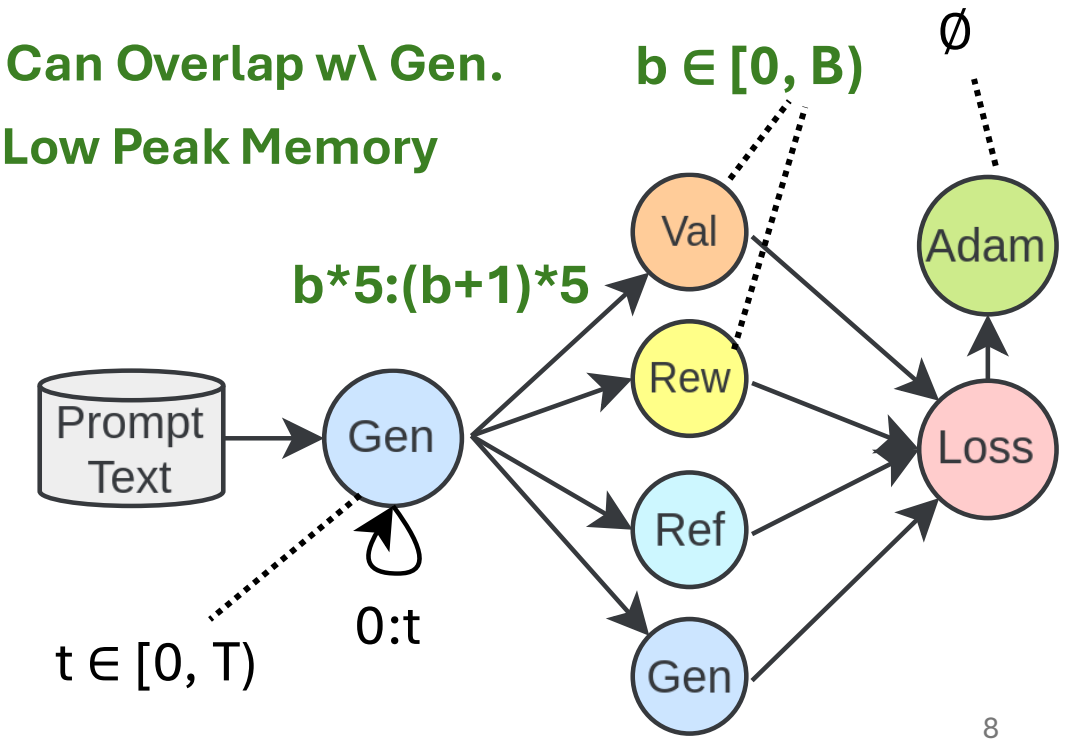
B = T/5

Can Overlap w\ Gen.

Low Peak Memory

b ∈ [0, B)

b*5:(b+1)*5



TimeRL: A Symbolic Deep Learning Compiler

Expose Symbols in **Programming Model**

(No pads/masks!)

Holistic Graph **Transformations**

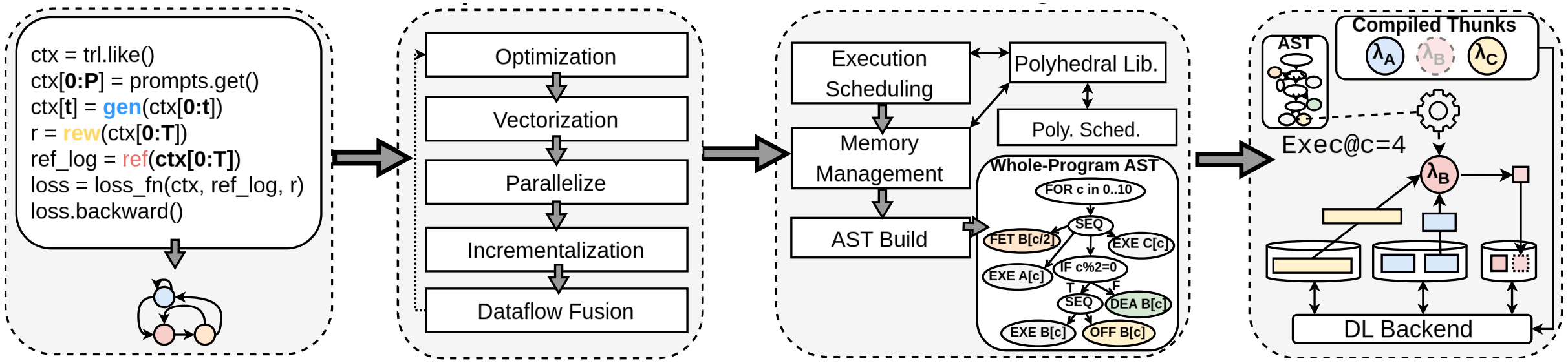
(Min. Variable-shapes)

Whole-Program **Scheduling**

(Emit a program AST)

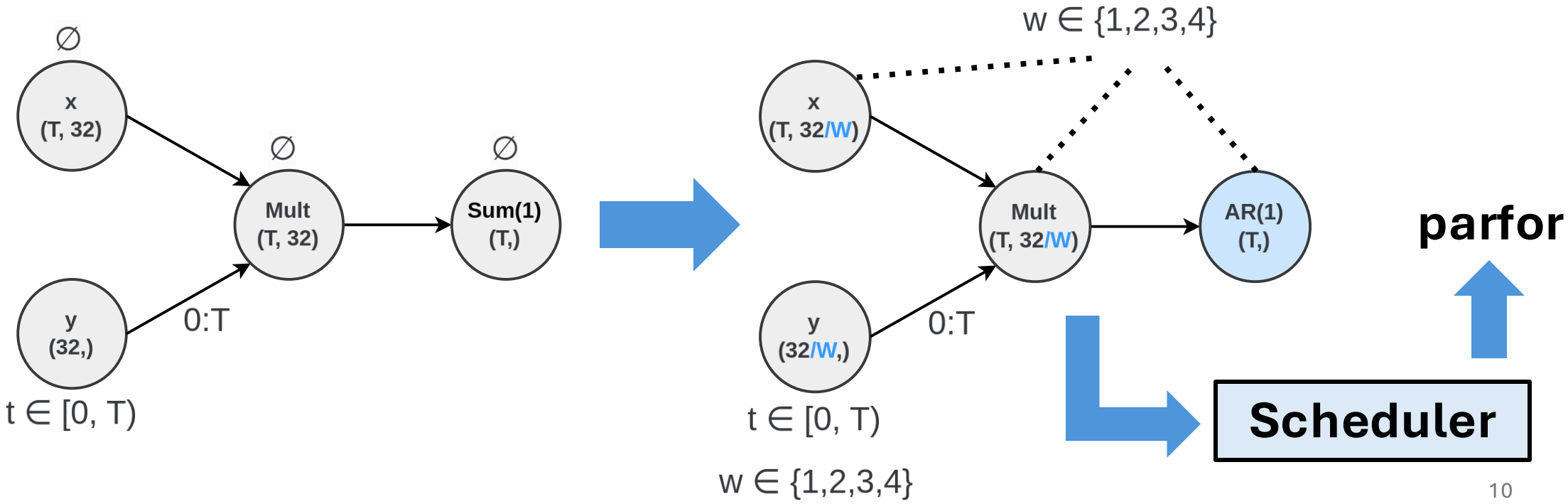
Evaluate Symbols at **Runtime**

(Specialized Storage)



How Symbols Simplify Parallelism

- Intra-Operator (TP, DP, SP):
  - **CollectiveOps**
  - **Worker** dimension
- Inter-Operator (PP, EP):
  - **P2POps**
  - **Microbatch** dimension

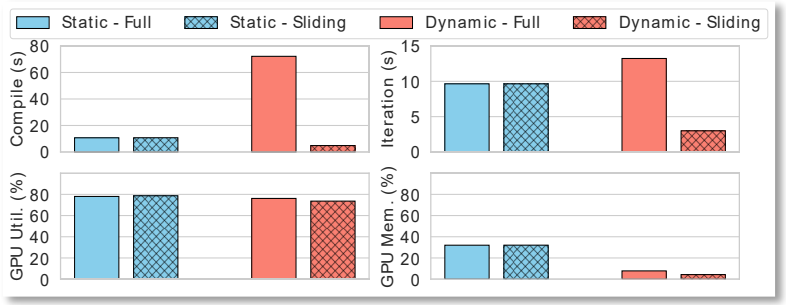


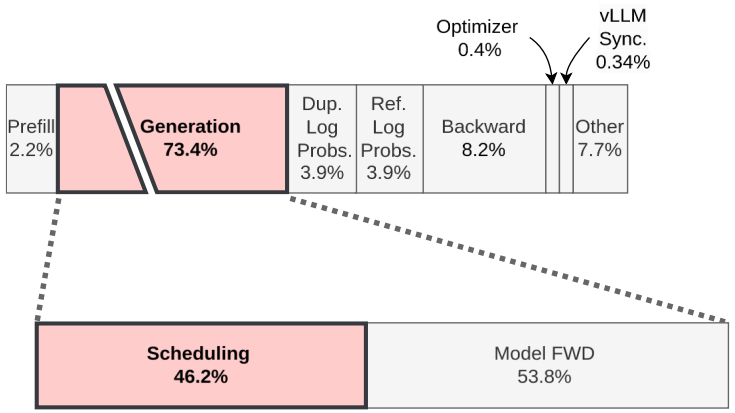
Come Find Out More!

- Challenges & Opportunities
  - Survey of existing RLFT Systems
- Evaluation of JAX for Generation
  - Overhead due to padding/masking.
  - Especially for window attention.
- Discussion on vLLM for Generation
  - Scheduling overheads!
  - Propose a method to address this

System	DL Eng.	Graph Based	Multi Algo.	Unif. Eng.	Par. Space
NeMo-Aligner[55]	PT	✗	✓	✓	3D§
OpenRLHF[26]	PT	✗	~	✗*	3D§
DS-Chat[71]	PT	✗	~	✓	3D§
TRL[65]	PT	✗	✓	✗*	3D§
FlexRLHF[69]	PT	✗	✗	✓	3D§
Puzzle[36]	PT	✗	✗	✗‡	3D§
ReaLHF[44]	PT	✓◊	✓	✓	3D
RLHFuse[79]	PT	✓◊	✓	✗†	3D
HybridFlow[56]	PT	✓◊	✓	✗*	3D

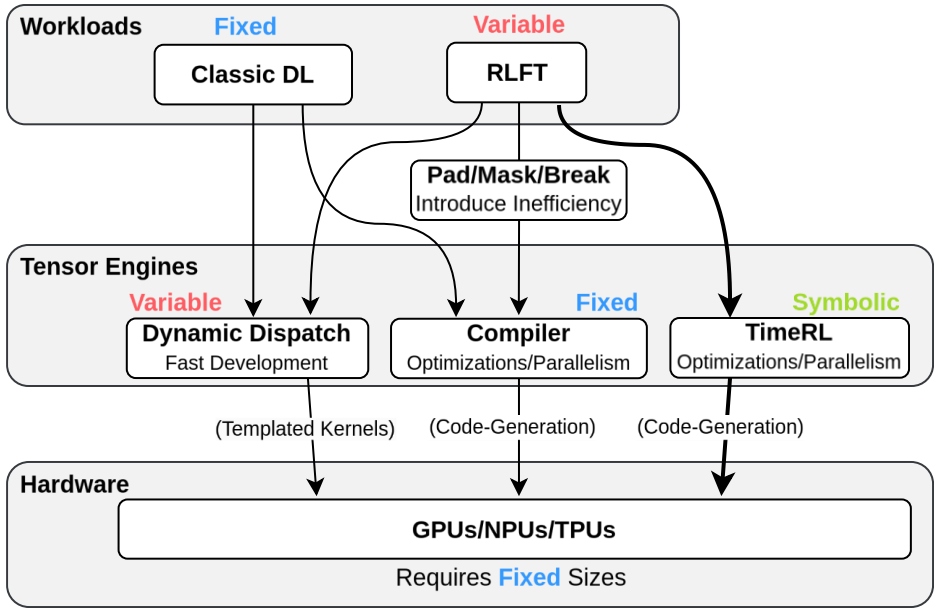
* vLLM for generation. † In-house vLLM-like engine for generation.
‡ DeepSpeed-Inference for generation. § Fixed strategy. ◊ Coarse-grained.





Conclusion

- RLFT generates **variable**-size contexts
- Existing compilers need **fixed** sizes
- Workarounds are **manual and not holistic**
- **TimeRL**: **symbols** to represent variables



TimeRL

TBD

RLFT Opportunities