Optimizing RLHF Training for Large Language Models with Stage Fusion

Yinmin Zhong[1] Zili Zhang[1] Bingyang Wu[1] Shengyu Liu[1] Yukun Chen[2] Changyi Wan[2] Hanpeng Hu[2] Lei Xia[2] Ranchen Ming[2] Yibo Zhu[2] Xin Jin[1]

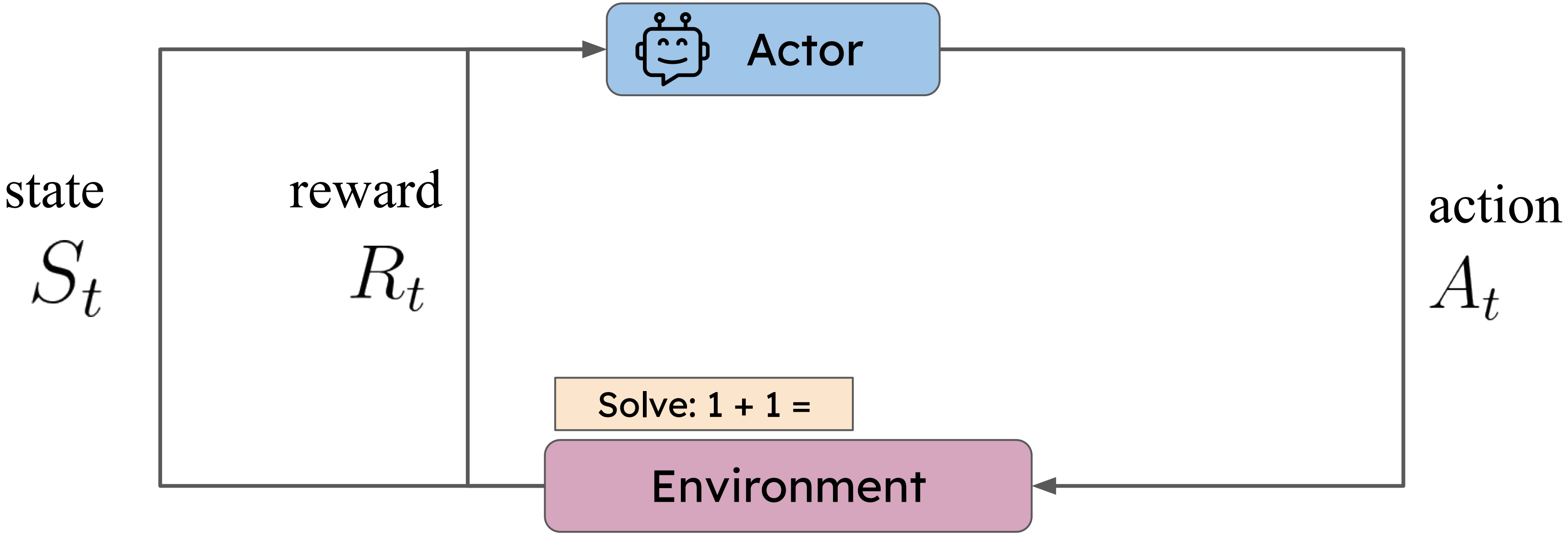
[1]*Peking University* [2]*StepFun*

What is RLHF?

RLHF = **R**einforcement **L**earning with **H**uman **F**eedback

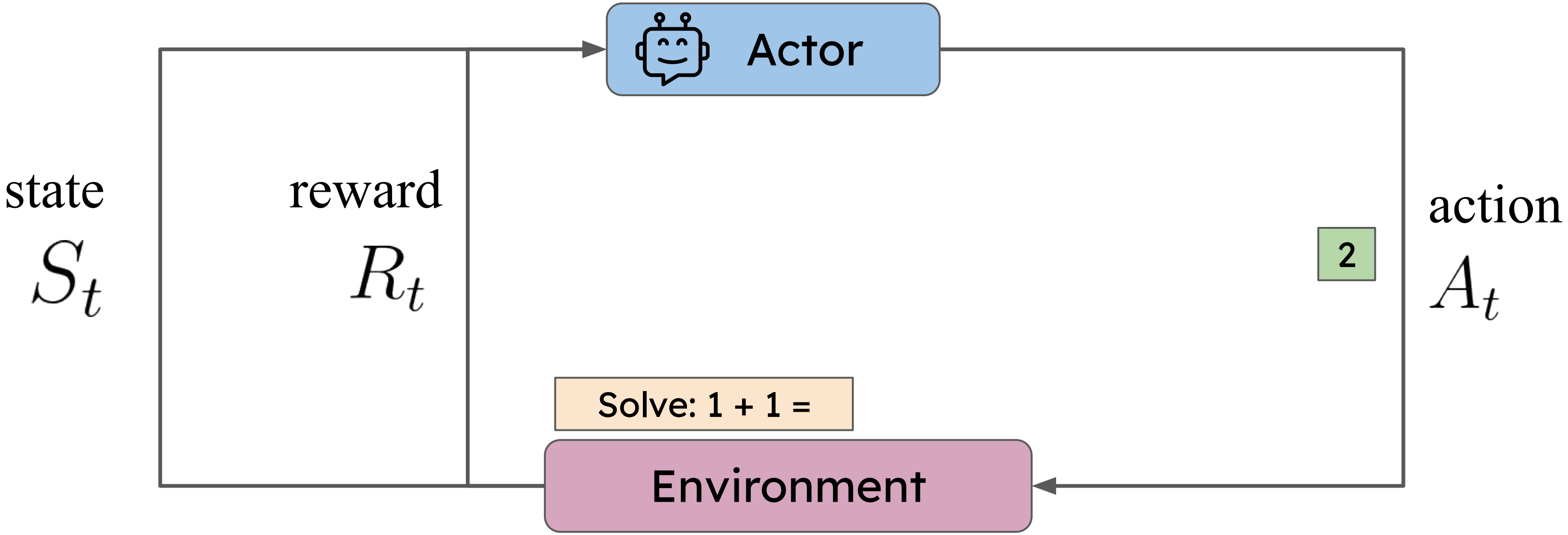
What is RLHF?

RLHF = **Reinforcement Learning** with **H**uman **F**eedback



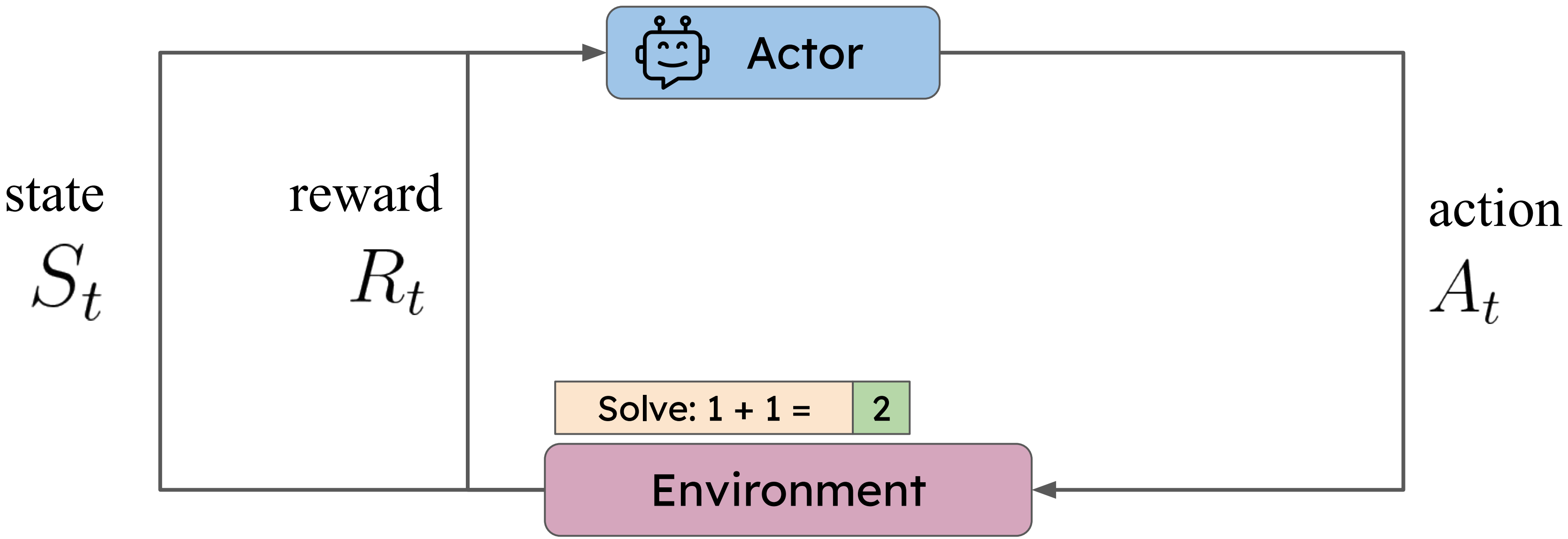
What is RLHF?

RLHF = **Reinforcement Learning** with **H**uman **F**eedback



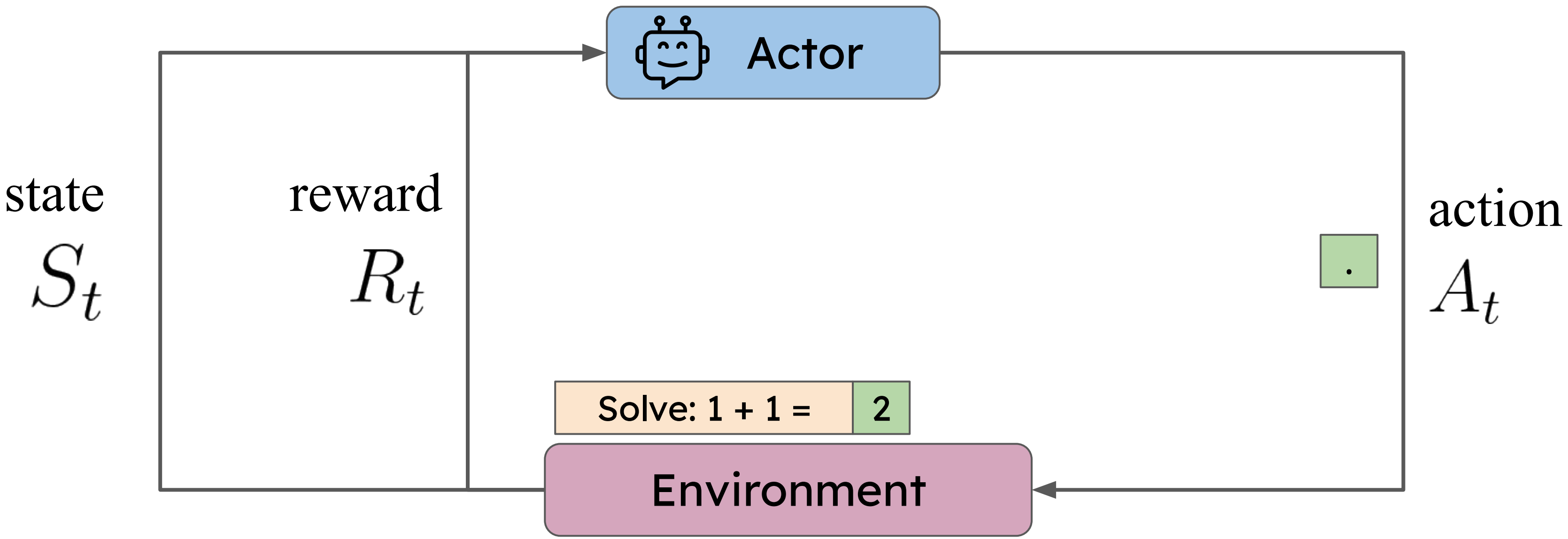
What is RLHF?

RLHF = **Reinforcement Learning** with **H**uman **F**eedback



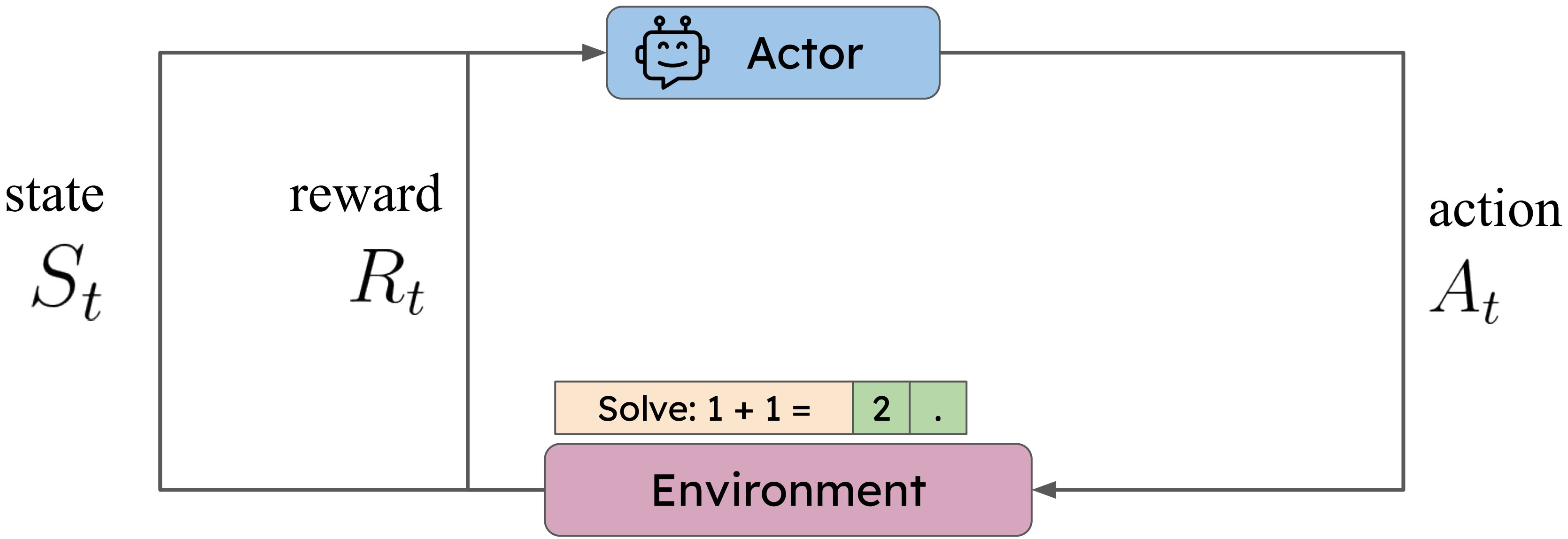
What is RLHF?

RLHF = **R**einforcement **L**earning with **H**uman **F**eedback



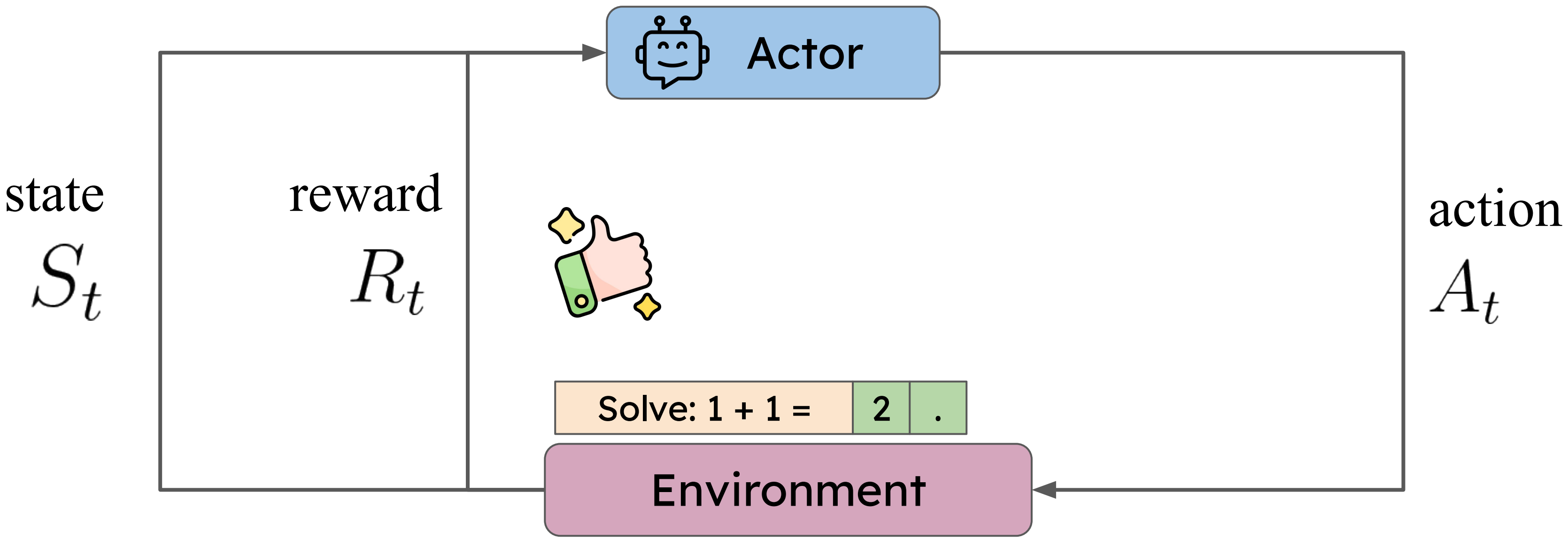
What is RLHF?

RLHF = **R**einforcement **L**earning with **H**uman **F**eedback



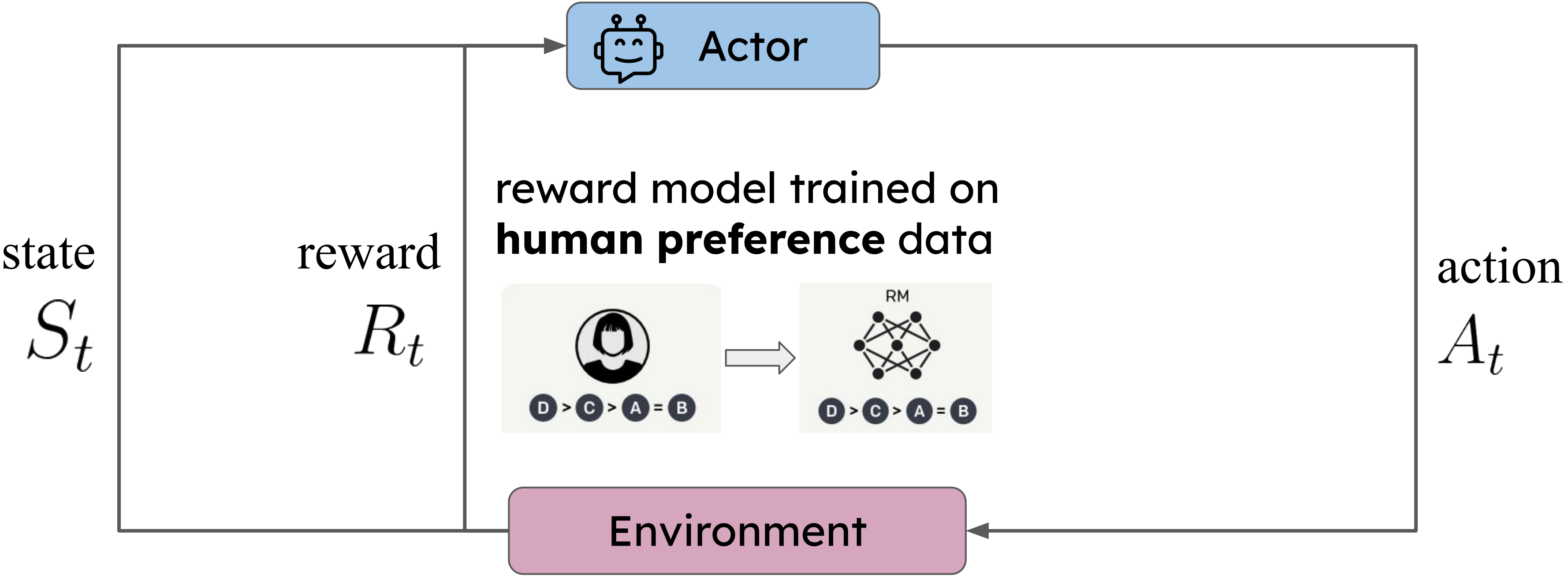
What is RLHF?

RLHF = **Reinforcement Learning** with **H**uman **F**eedback



What is RLHF?

RLHF = **R**einforcement **L**earning with **H**uman **F**eedback



Source: Yuntao Bai, et al. "Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback." .

Why use RLHF?

- Human preference
- Safety and robustness
- Hallucination alleviation
- Reasoning ability
- Complex tasks
- … … … … … …

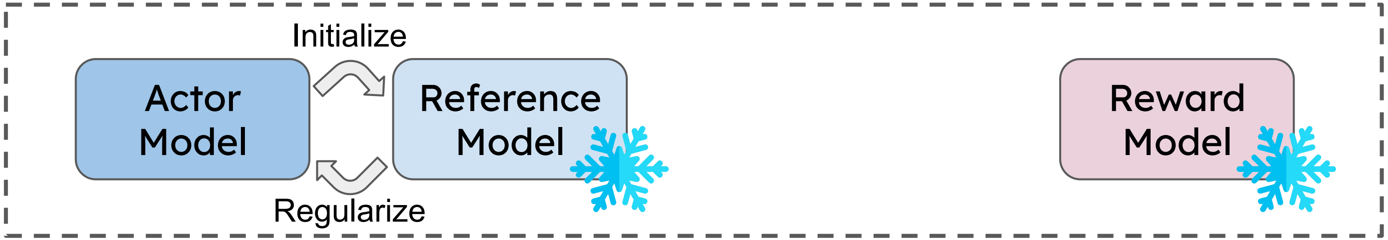
GPT-4 → ChatGPT

How to do RLHF?

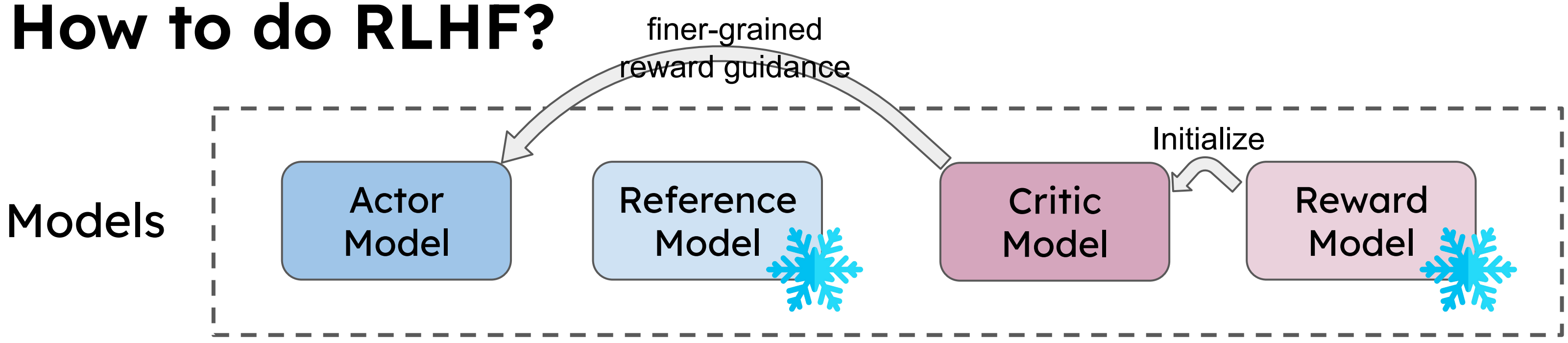
Models

How to do RLHF?

Models

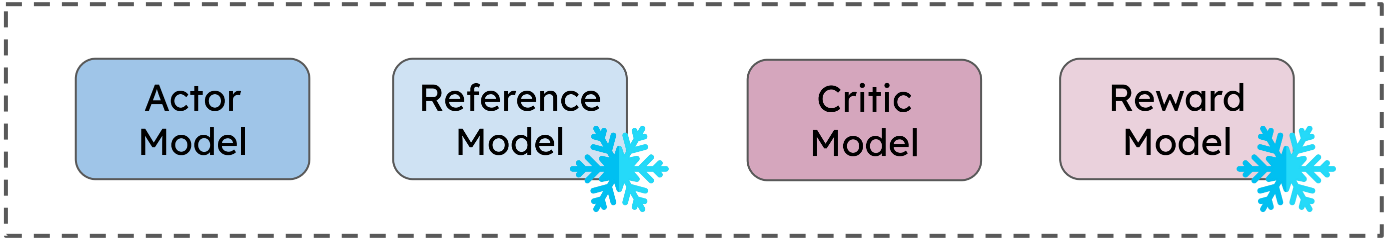


How to do RLHF?



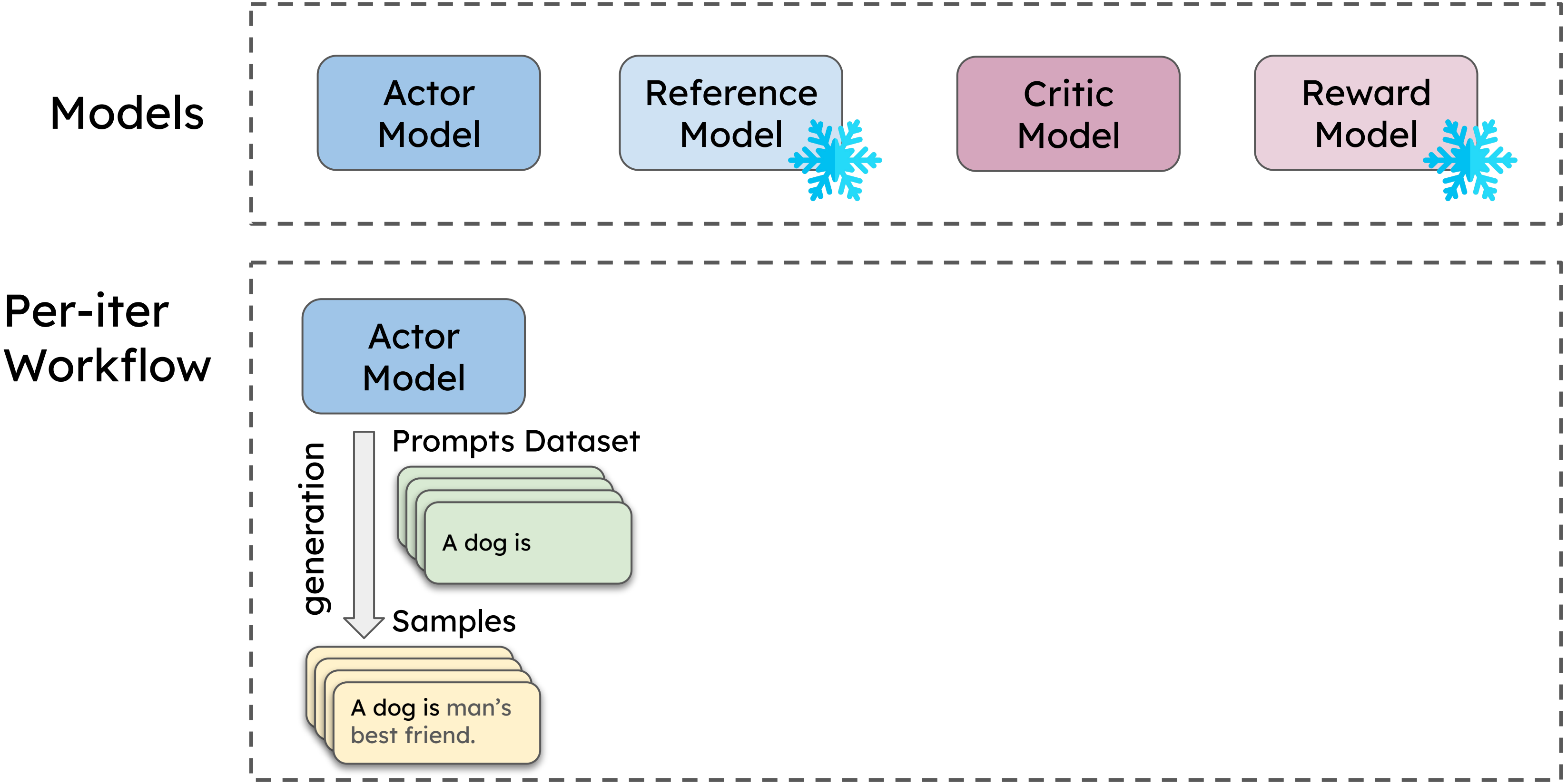
How to do RLHF?

Models

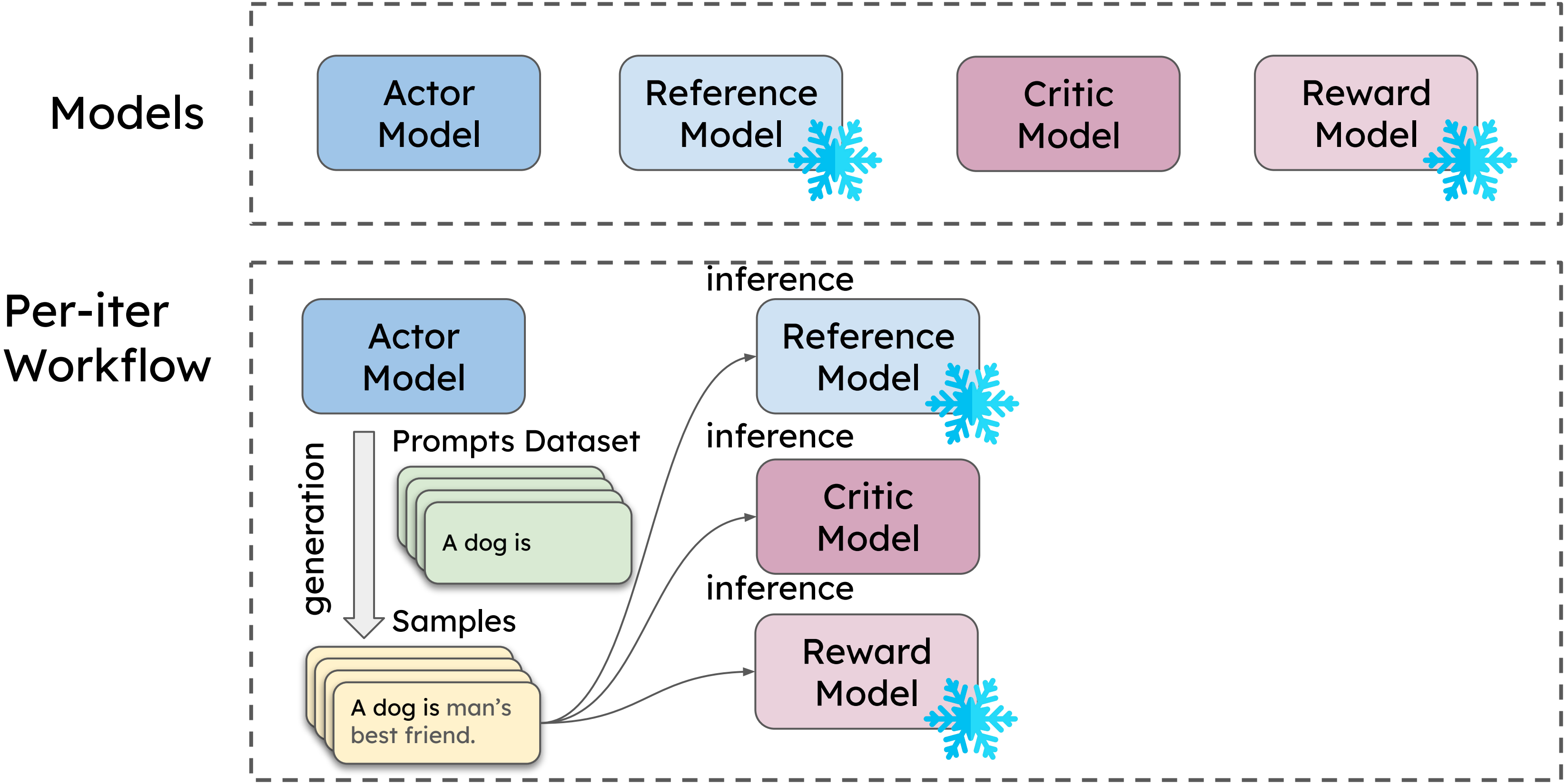


Per-iter Workflow

How to do RLHF?

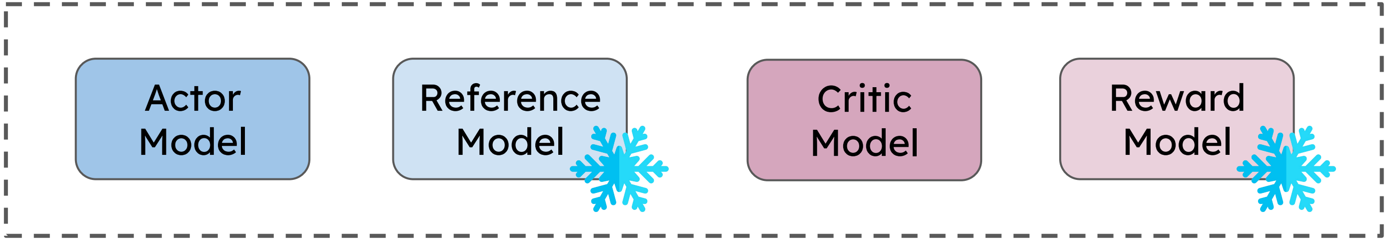


How to do RLHF?

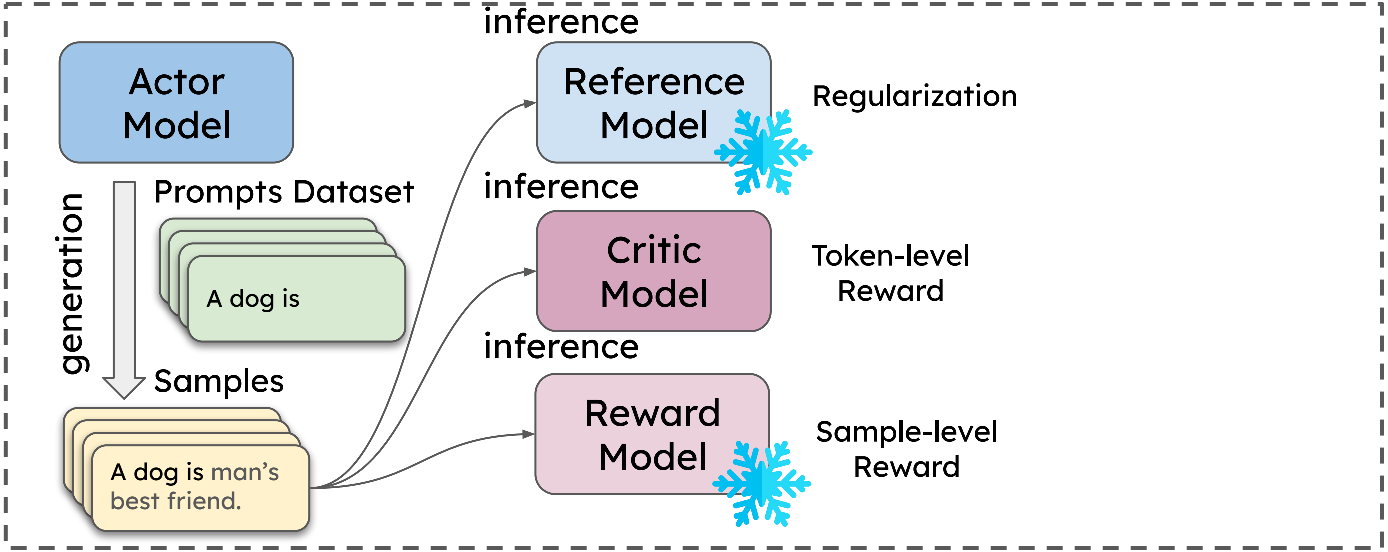


How to do RLHF?

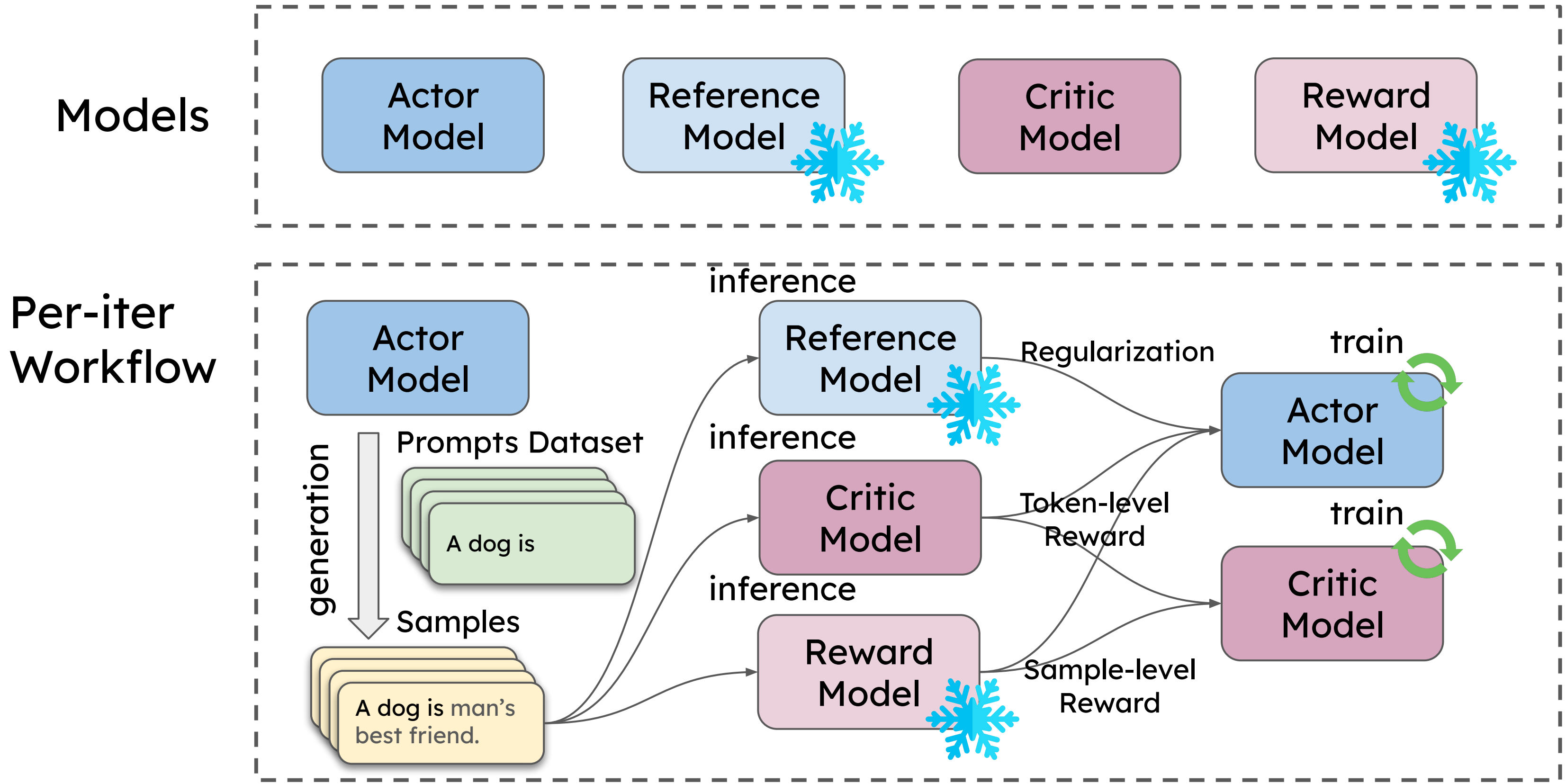
Models



Per-iter Workflow



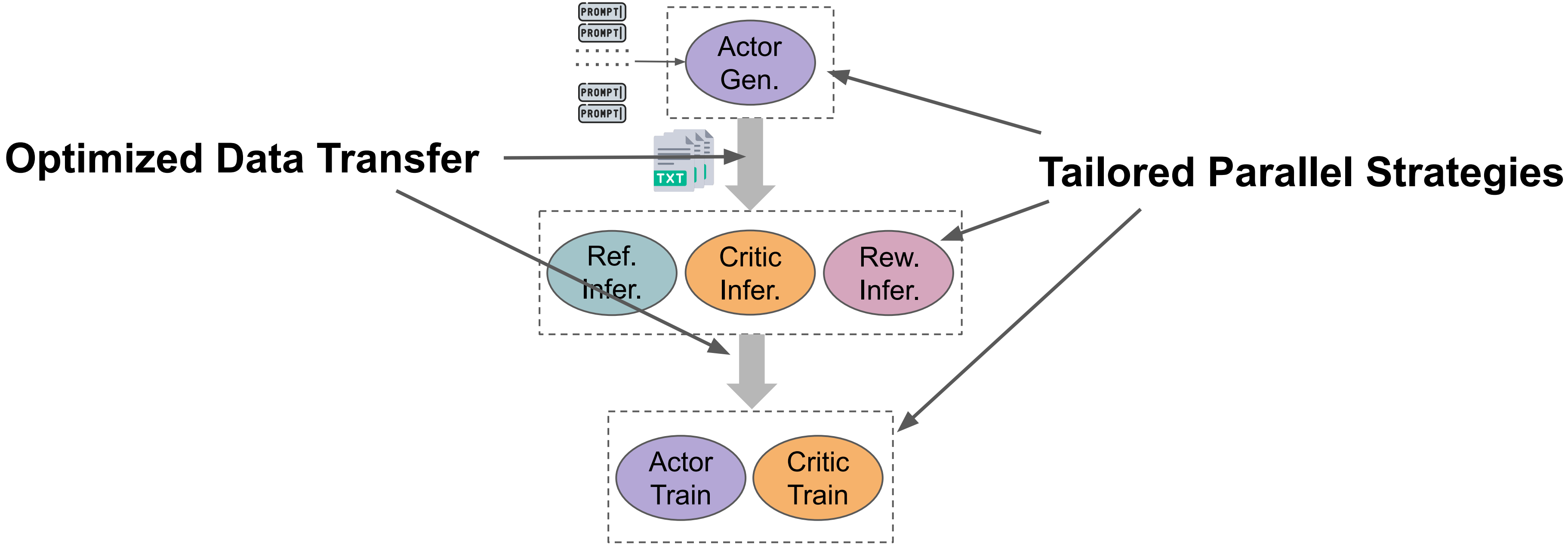
How to do RLHF?



Summary

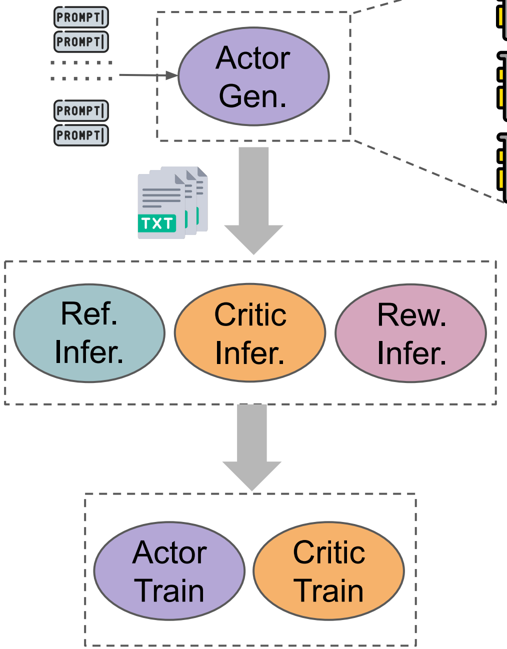
- Four unique models
  - each can have billions of parameters
  - huge computation and memory requirement
- Three distinct stages
  - generation => inference => training
  - different computation pattern
- Six individual tasks
  - complex workflow
  - data, weights, and communication orchestration

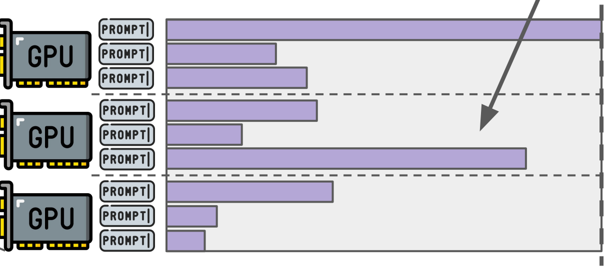
Optimizations in existing systems

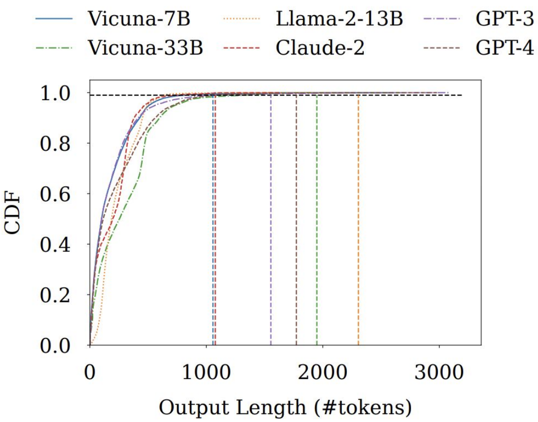


Problems in existing systems

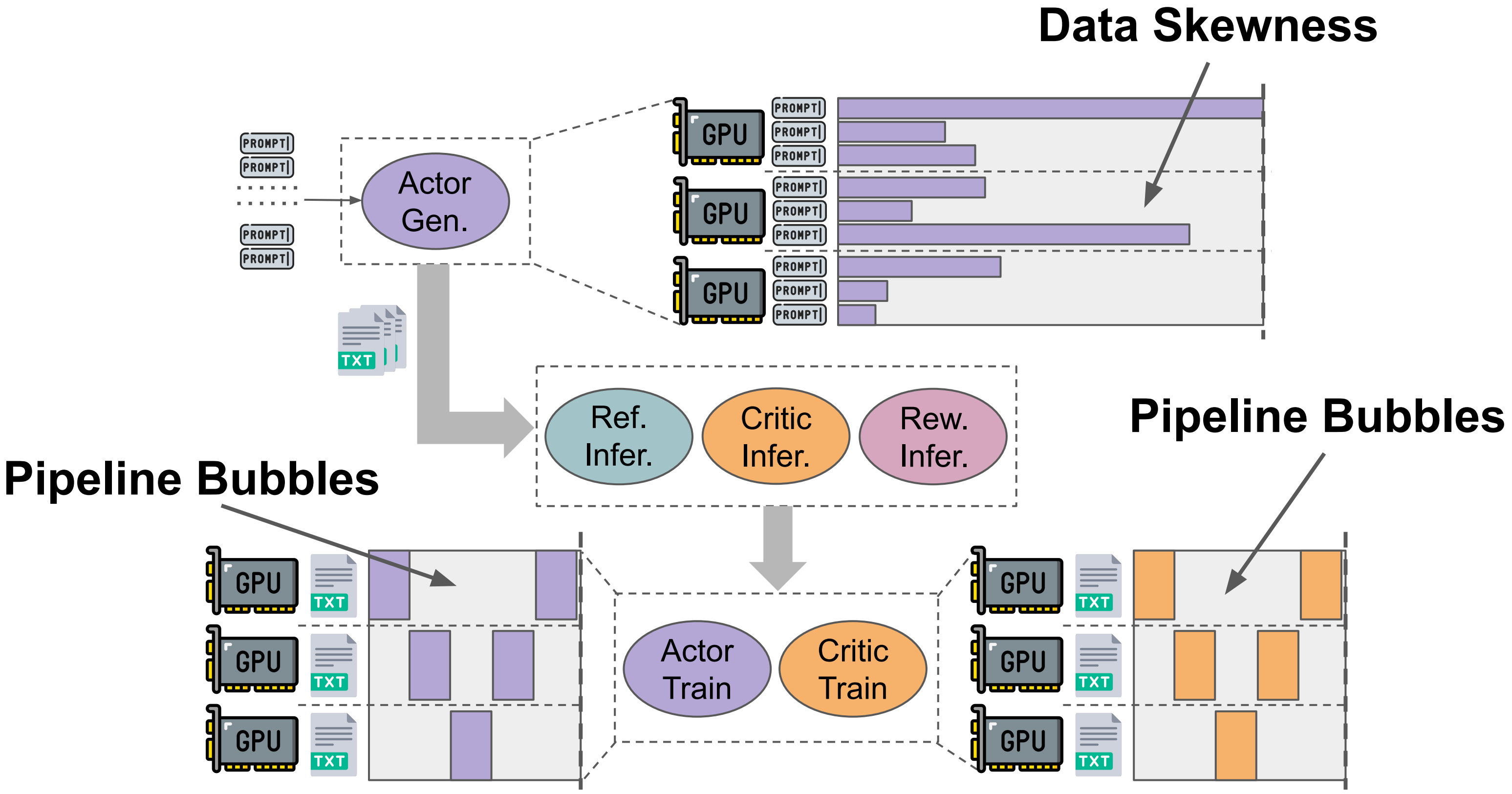
Data Skewness







Problems in existing systems



The fundamental problem

Existing RLHF training systems view **each task** as the **smallest execution unit**, failing to delve into the **inherent characteristics and structure** inside the tasks.

The fundamental problem

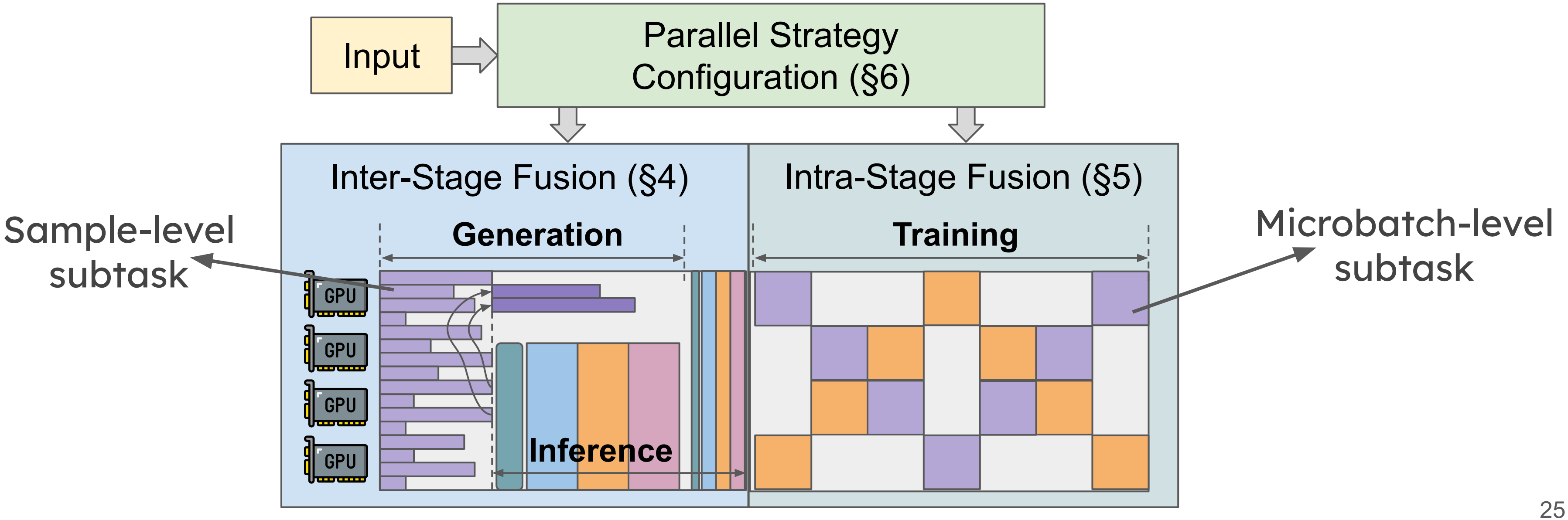
Existing RLHF training systems view **each task** as the **smallest execution unit**, failing to delve into the **inherent characteristics and structure** inside the tasks.

Opportunities

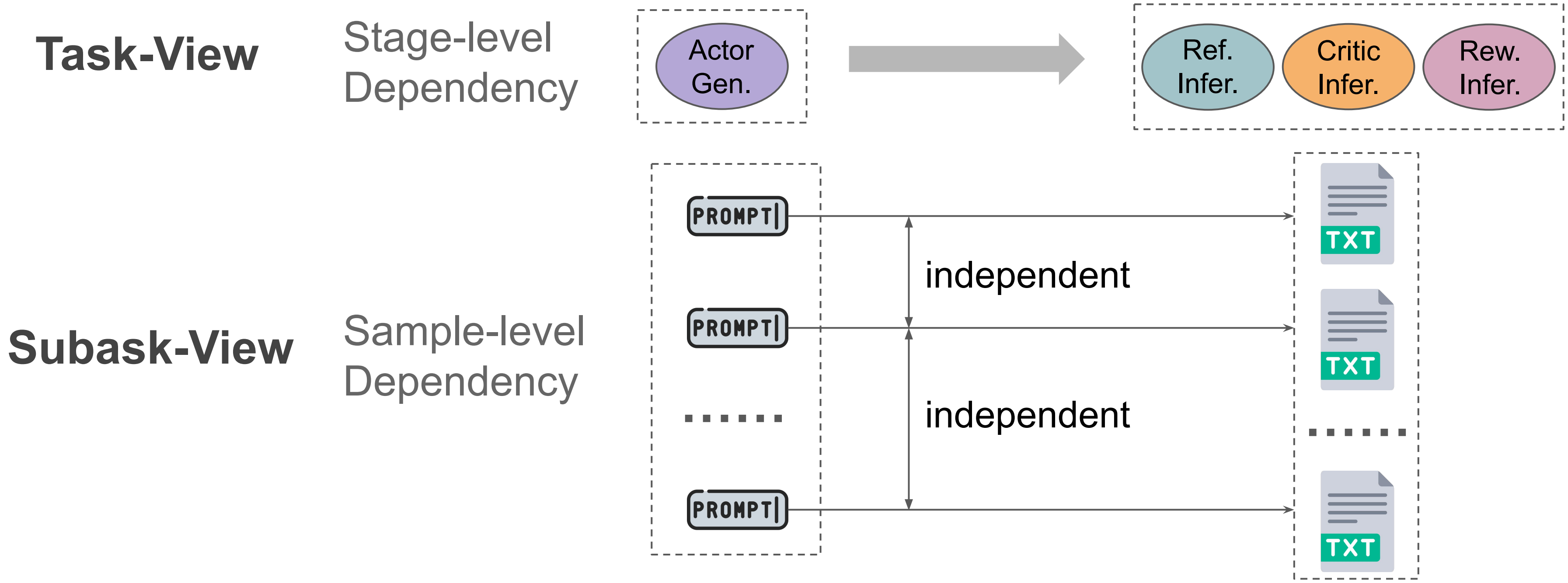
- Generation Stage: each **sample** can be viewed as a subtask
- Training Stage: each **micro-batch** can be viewed as a subtask

RLHFuse Overview

RLHFuse breaks the traditional view of RLHF workflow as a composition of individual tasks, splitting each task into **finer-grained subtasks**, and performs **stage fusion** to improve GPU utilization.

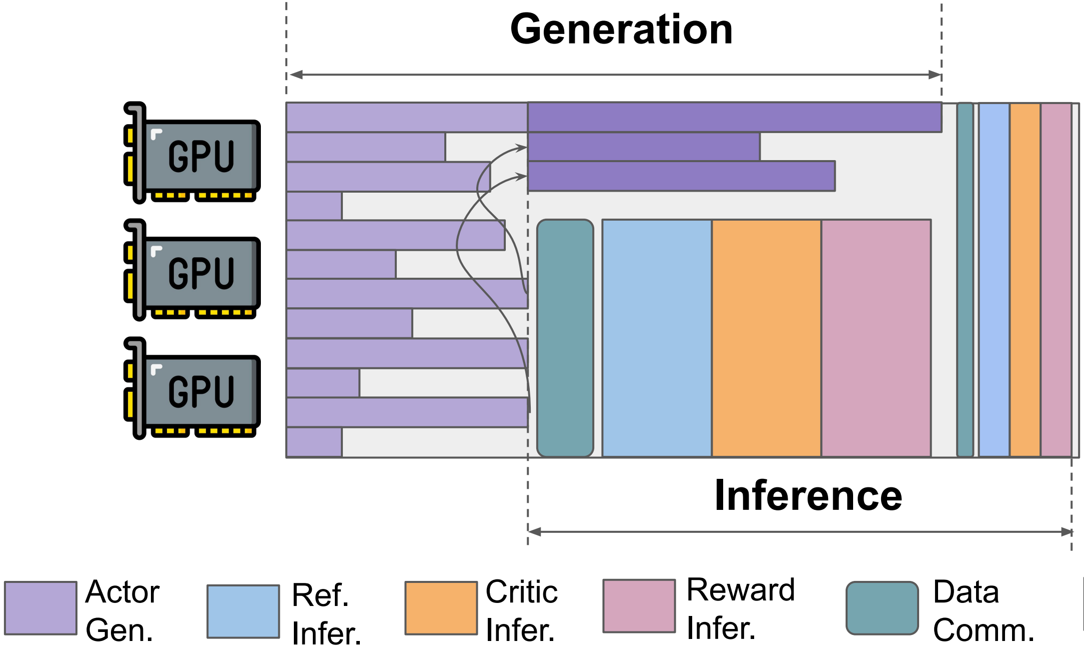


Core observation 1: Sample-level Dependency



Data-aware Inter-stage Fusion

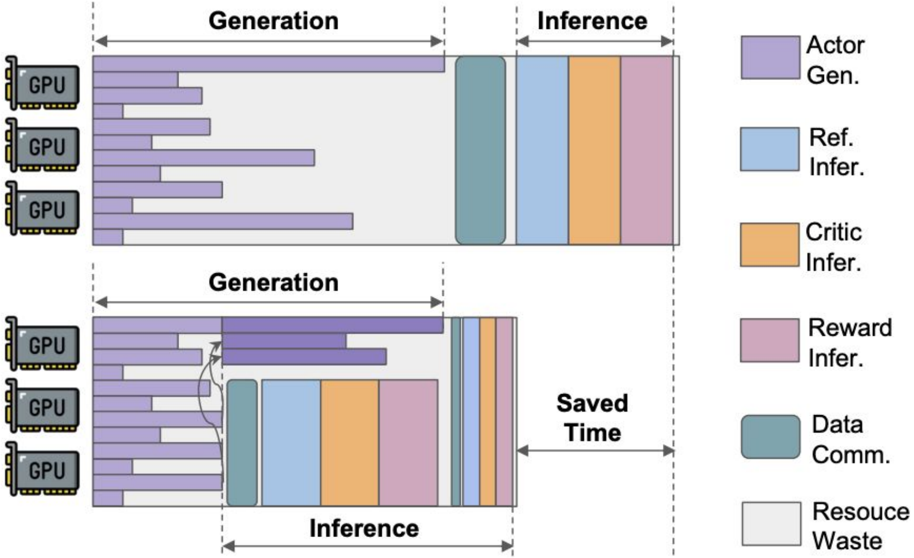
Migrate long-tailed samples together and start inference stage early to overlap with long-tailed generation.



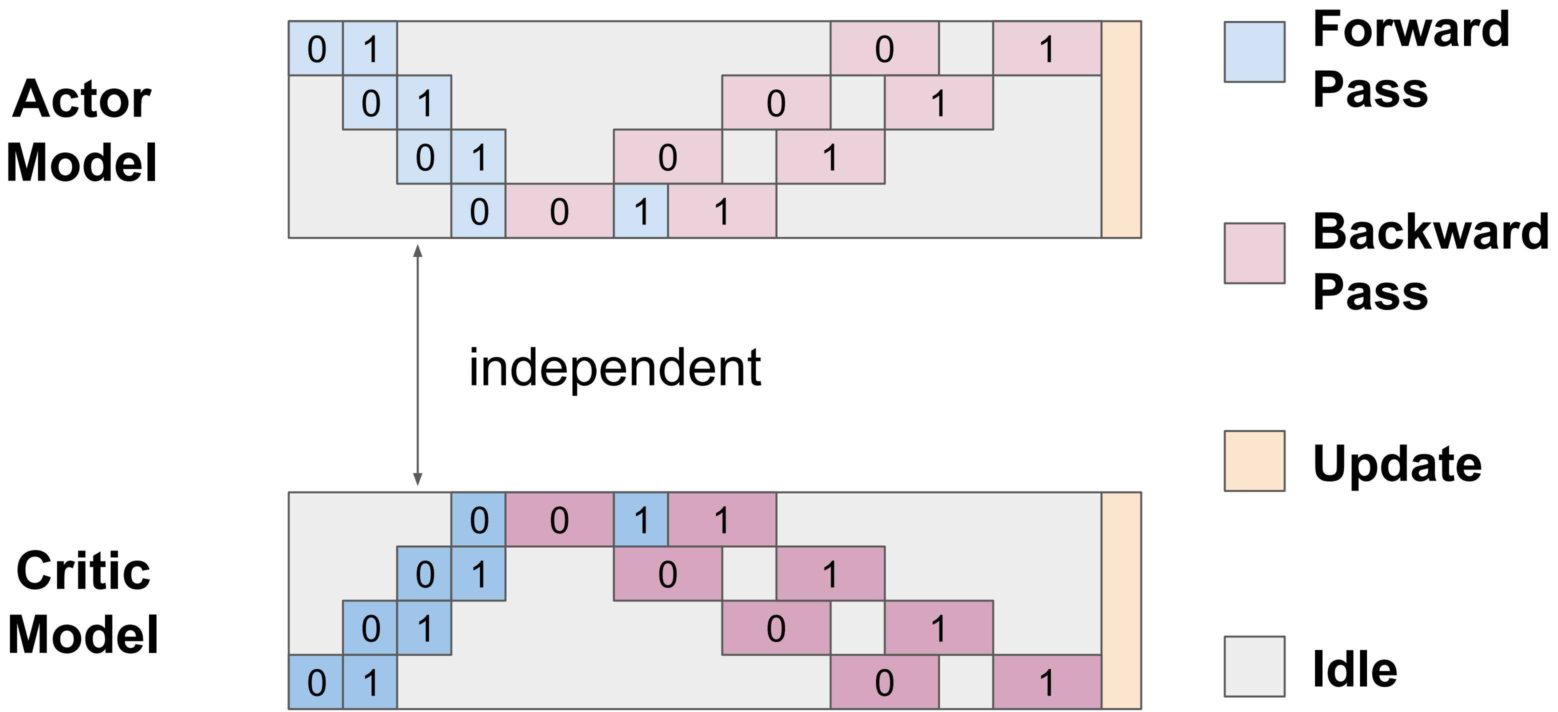
Data-aware Inter-stage Fusion

Algorithm Sketch:

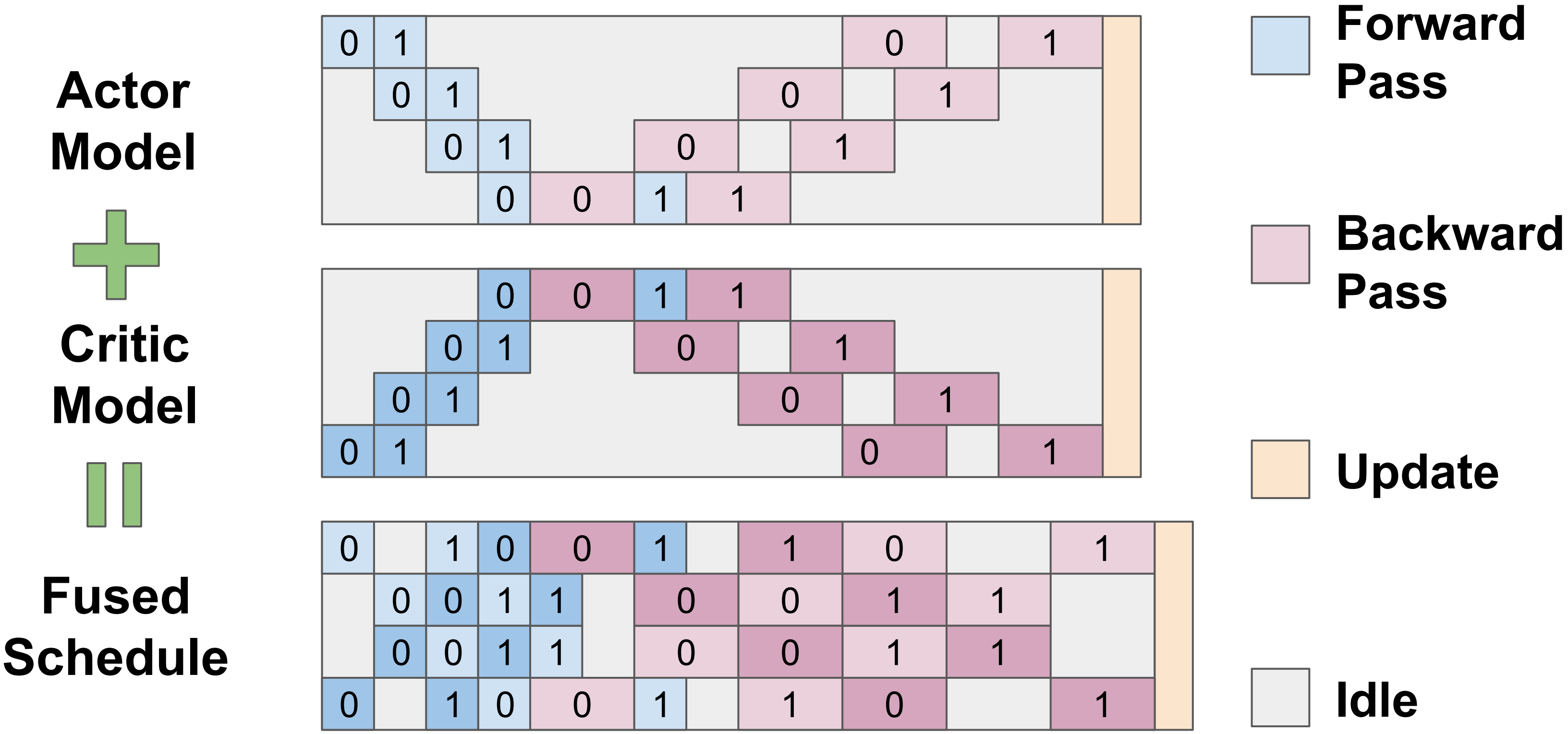
- Migration timing
  - determine migration ratio R_t
- Migration destination
  - latency/memory constraint
  - minimize #migration
- Migration mechanism
  - migrate key-value cache
  - migrate token



Core observation 2: Independent Training Tasks



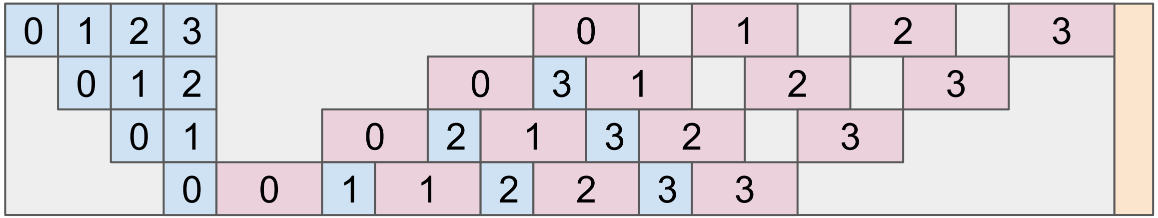
Model-aware Intra-stage Fusion



Shigang Li, Torsten Hoefler. “Chimera: Efficiently Training Large-Scale Neural Networks with Bidirectional Pipelines"

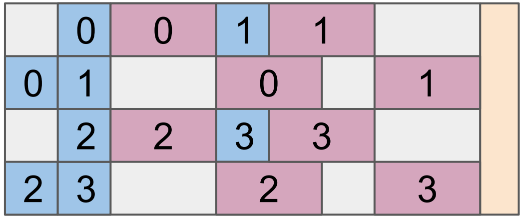
Model-aware Intra-stage Fusion

Actor Model



+

Critic Model



=

?

Forward Pass

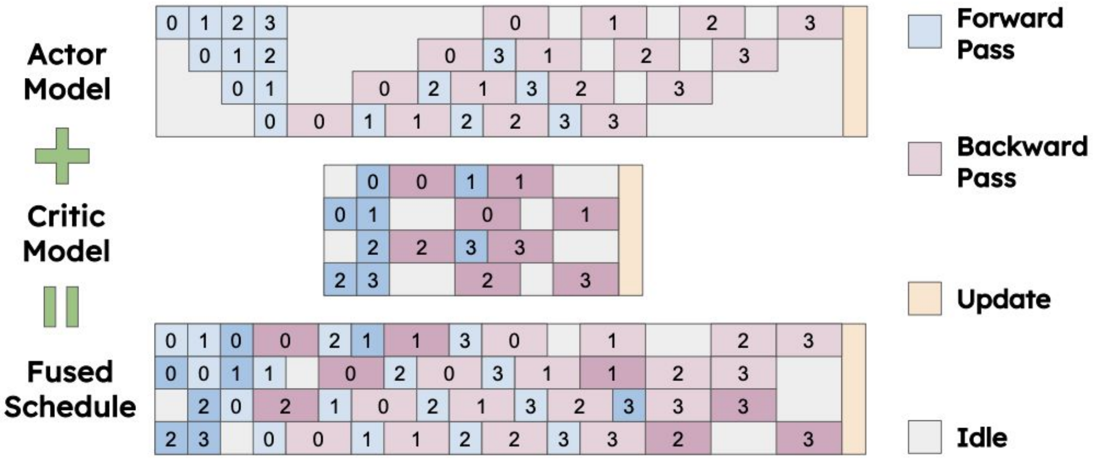
Backward Pass

Update

Idle

Model-aware Intra-stage Fusion

Fuse any two different pipeline schedules to achieve optimal latency and memory usage.



Algorithm Sketch:

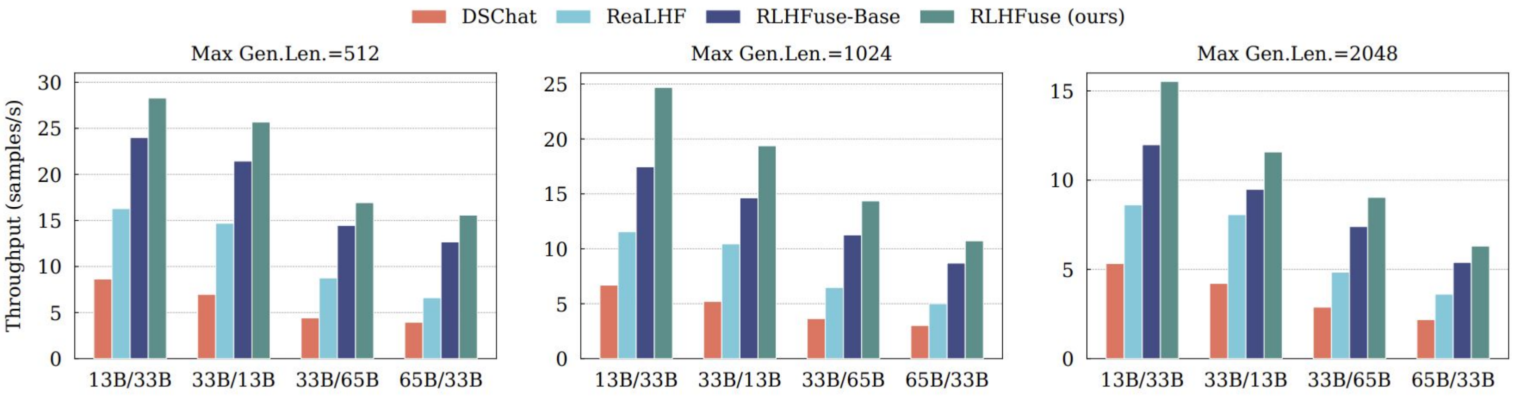
- Problem transformation
  - tackle different tp
  - fusion factor (K1, K2)
- Simulated Annealing
  - swap adjacent subtasks
  - first optimize latency
  - then optimize memory

Evaluation – Setup

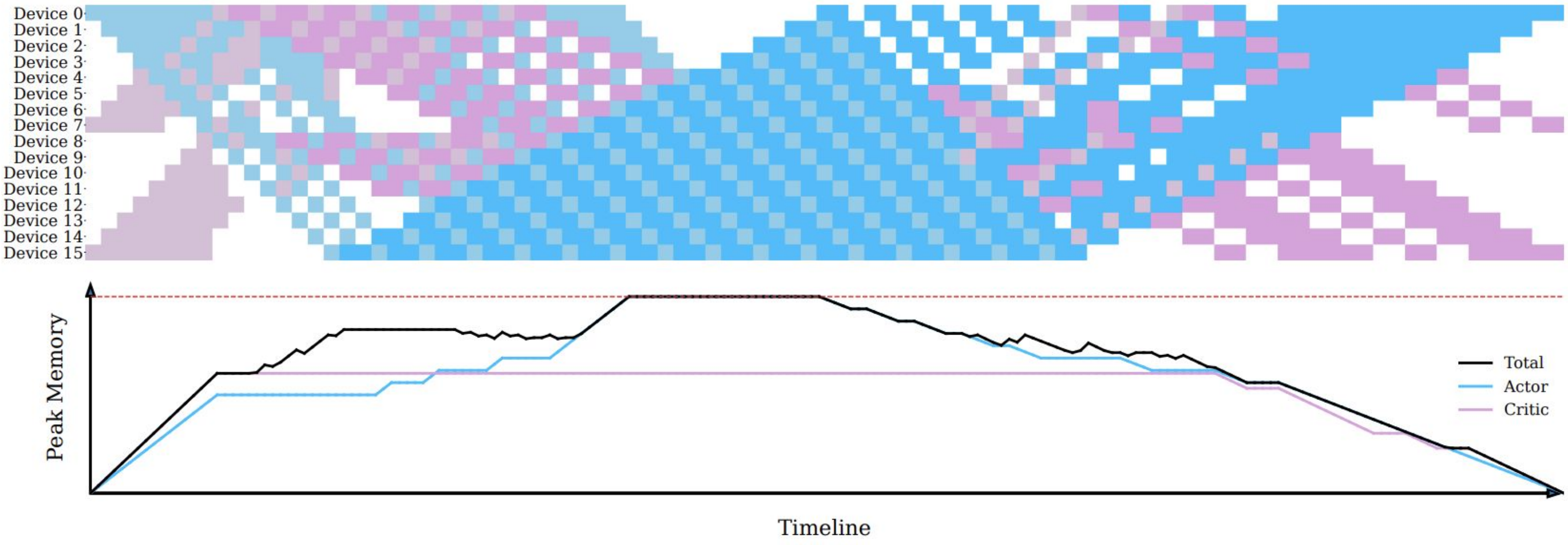
- Cluster: 32 nodes with 256 H800-80GB GPUs
- Models: LLaMA-13B, LLaMA-33B, LLaMA-65B
- Dataset: HH-RLHF
- Settings:
  - Critic/Actor: 13B/33B, 33B/13B, 33B/65B, 65B/33B
  - Maximum generation length: 512, 1024, 2048
- Baselines:
  - DeepSpeed-Chat
  - ReaLHF
  - RLHFuse-Base

Evaluation – End-to-end

RLHFuse achieves 2.5×–3.7× higher end-to-end throughput.



Evaluation – Case Study



Takeaway

- RLHFuse
  - breaks the RLHF workflow as a composition of **individual tasks**
  - splits generation task into **sample-level subtasks**
  - splits training task in **microbatch-level substasks**
  - uses **inter- and intra-stage fusion** to achieve higher GPU utilization
  - achieves up to **3.7x** higher throughput compared with SOTA systems

https://github.com/FlexFusion/FlexFusion

zhongyinmin@pku.edu.cn