verl: Flexible and Efficient Reinforcement Learning Library for LLM Reasoning and Tool-Calling

Presenter: Hongpeng Guo, ByteDance Seed

Why is large-scale RL important?

Systems challenges of large scale RL

Why verl for LLM RL?

Recent updates & Roadmap

Why is Large-Scale RL Important?

Large-Scale RL for Reasoning and Agents

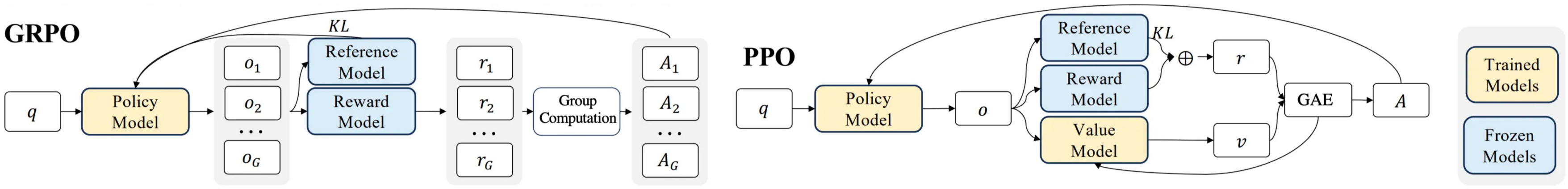
Learning to reason with large-scale RL greatly boosts the performance of LLMs

Model	Large-Scale RL?	AIME 2024	MATH 500	GPQA Diamond	Code Forces
GPT-4o (OpenAI 2024)	❌	44.6	60.3	50.6	>11.0%
o1 (OpenAI 2024)	✅	74.4	94.8	77.3	>89.0%

> ***Deep research** … was trained on **real-world tasks requiring browser and Python tool use**, using **the same reinforcement learning methods behind OpenAI o1**, our first reasoning model.*
>
> *– OpenAI Deep Research Blog, 2025*

Why is Large-Scale RL System Challenging?

Diverse RL algorithms as Complex Dataflows



Reinforcement learning can be modelled as **complex dataflow graph** with:

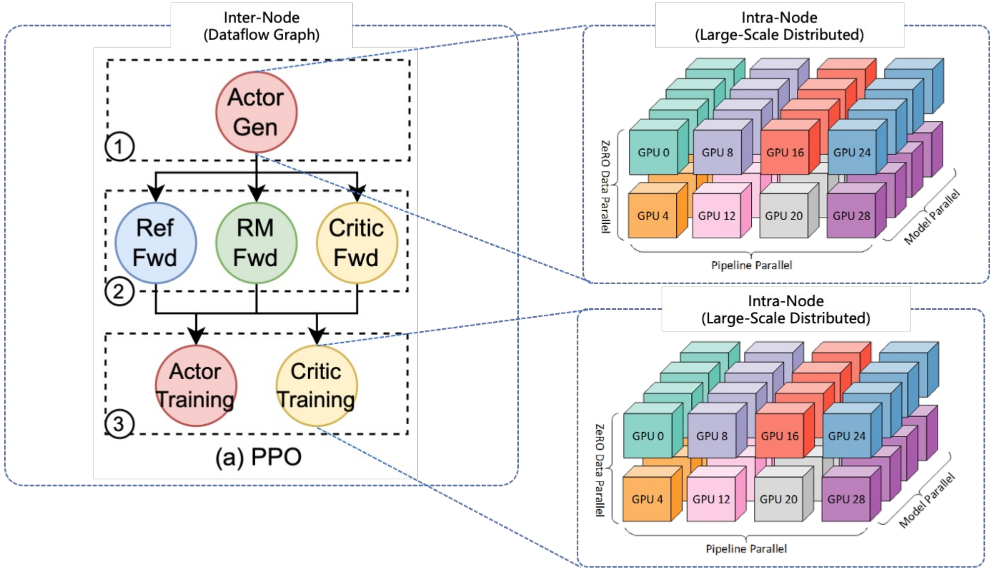
- **multiple models**:
  - Policy model (actor): the LLM to train
  - Reward model: provides immediate rewards
  - Reference model: makes sure the updated policy model not to deviate too far with KL
  - Value model (critic): predicts the long term value of the state
- **multiple workloads**: generation, inference, training, weight sync

(Schaarschmidt et al. 2019; Liang et al. 2021; Sheng et al. 2025

https://fireworks.ai/blog/reinforcement-learning-with-verifiable-reward

RL with LLMs is Large-Scale Distributed Dataflow

each **operator** in the RL dataflow = a large-scale **distributed** computing workload



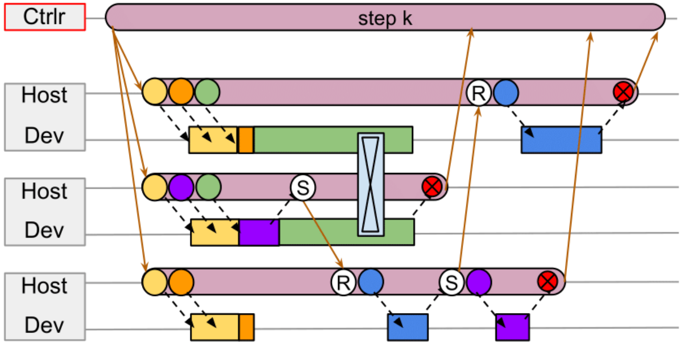
HybridFlow, Sheng et al., 2024

Why verl for RL with LLMs?

Flexible and Efficient!

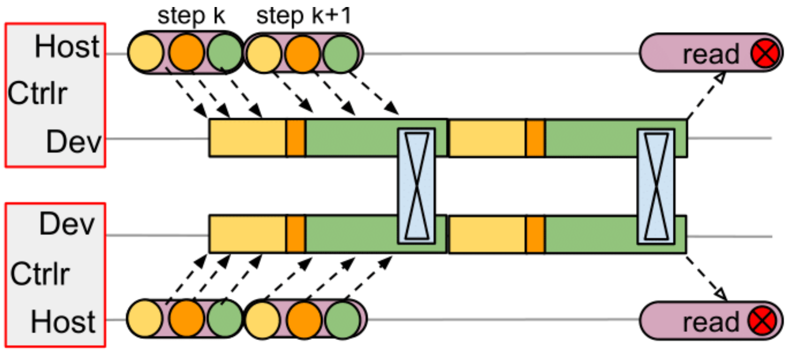
Background: Single-Controller vs. Multi-Controller

Single-Controller (MPMD, flexible):
A centralized controller manages all the workers, running different programs



I.e., Tensorflow 1, Rllib, …

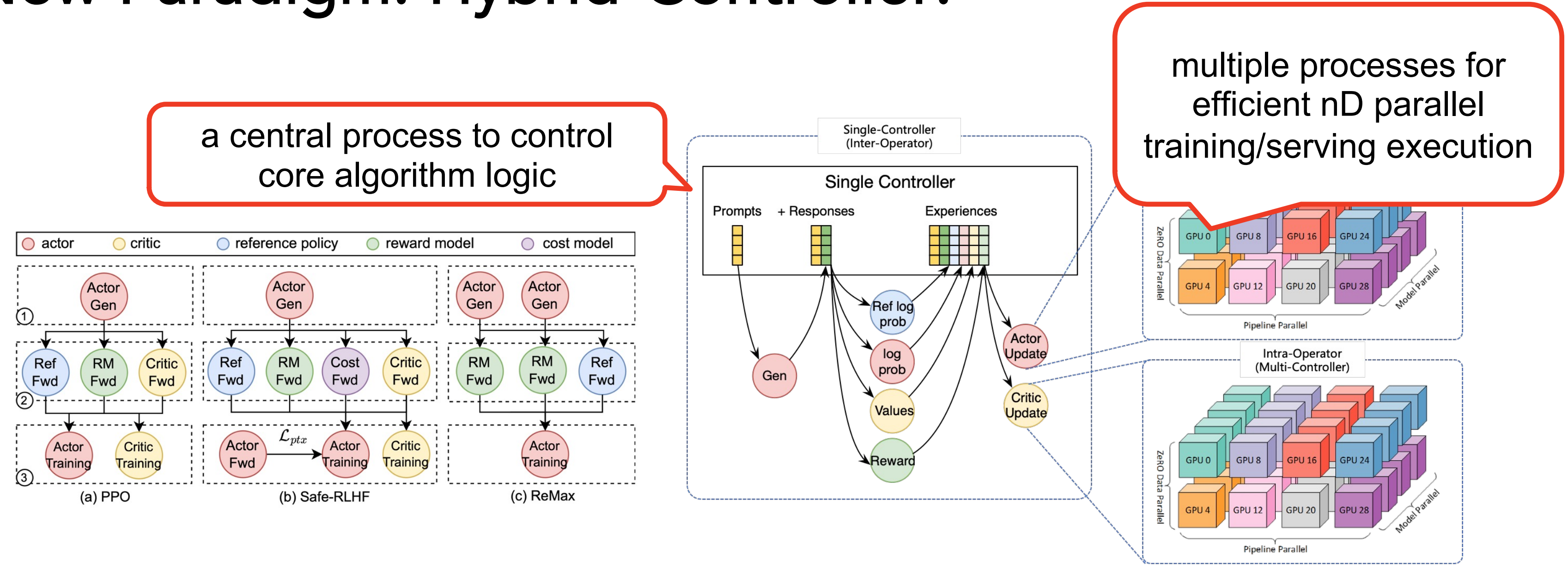
Multi-Controller (SPMD, efficient):
Each worker has its own controller, running the same program with different data



I.e., PyTorch, JAX, …

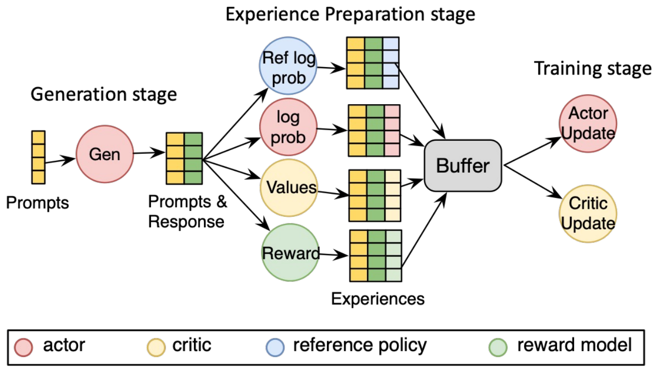
Pathways, Barham et al., 2022

New Paradigm: Hybrid-Controller!



- Hybrid-Controller = Single-Controller + N x Multi-Controller
- In the hybrid-controller, a single-controller manages multiple multi-controllers to process the dataflow

Flexibility in Programming: “Single-Controller”




```python
for prompts in dataloader:
    # Stage 1: Generation
    batch = actor.generate_sequences(prompts)
    # Stage 2: Experience Preparation
    batch = reward.compute_reward(batch)
    batch = reference.compute_log_prob(batch)
    batch = critic.compute_values(batch)
    batch = compute_advantage(batch, "gae")
    # Stage 3: Training
    critic.update_critic(batch)
    actor.update_actor(batch)
```

- Programming interface based on the “**single-controller**” paradigm
- RL algorithm core logic in **a few lines of code**!
- Diverse RL algorithms supported: PPO, GRPO, RLOO, GSPO, PRIME, DAPO, etc.

HybridFlow, Sheng et al., 2024

Efficiency: “Multi-Controller”

verl is efficient for intra-operator with the **“multi-controller”** paradigm and features like:

Training Backends:
- FSDP
- FSDP2
- Megatron

Generation Backends:
- vLLM
- SGLang
- …

Parallelism Algorithms:
- Data Parallelism
- Tensor Parallelism
- Pipeline Parallelism
- Context / Sequence Parallelism
- Expert Parallelism

Efficient Kernels:
- Flash Attention 2
- Torch Compile
- Liger Kernel
- …

Open-Source Community: Impactful and Inclusive

So far, verl has gained:

- 13.3k stars
- 2.4k forks
- 1.9k PRs
- 360+ contributors

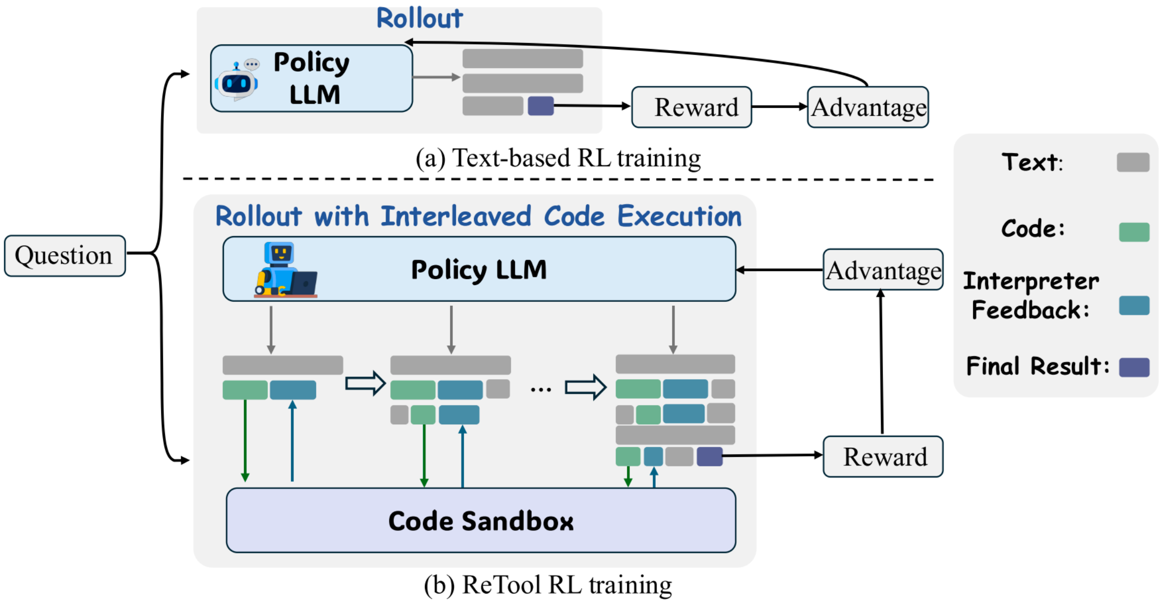
Waiting for your participation!

Capabilities

- VLM recipe: deep-eye
- LLM recipes: DAPO, retool
- Image/video support
- Large MoE
- Multi-GPU LoRA
- Sandbox/search tools

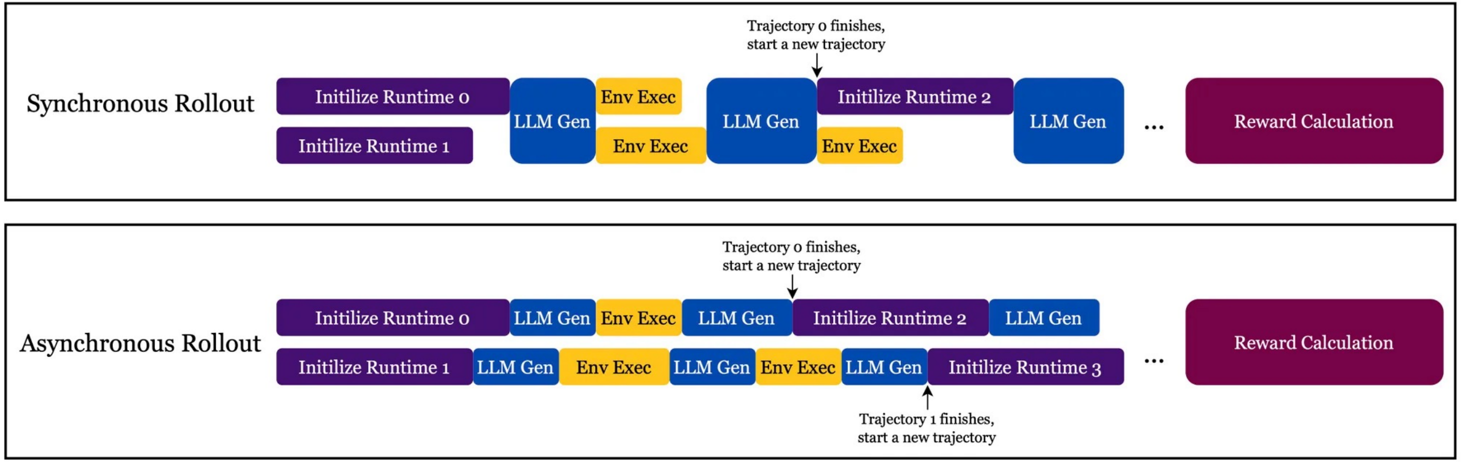
Recent Updates
& Roadmap

Approaching Agentic RL



ReTool, Feng et al., 2025

Async Multi-Turn Rollout with Tools

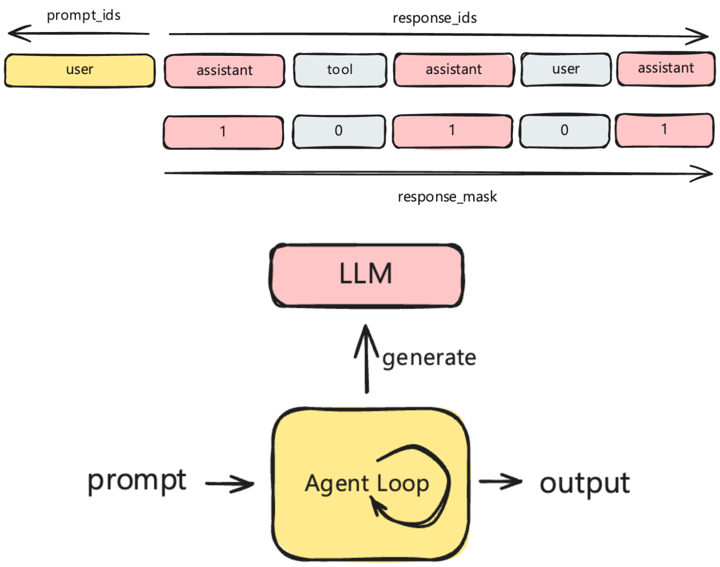


- Synchronous Engine: returns all the outputs in the batch at the same time
- Asynchronous Server (rollout.mode=async): returns each output as soon as it is ready 🚀🚀🚀

https://novasky-ai.notion.site/skyrl-v0

Token-in-token-out Agent Loop Interface

Given one prompt, run a user defined loop with multi-turn/tool calling trajectories.
Token ids are used for server generation input / output to avoid ambiguity.




```python
class DemoLoop(AgentLoopBase):

    async def run(self, messages: list[dict[str, Any]], ...) -> AgentLoopOutput:
        prompt_ids = await self.loop.run_in_executor(None,
            lambda: tokenizer.apply_chat_template(messages, ..., tokenize=True)
        )
        num_turns = 0
        while not is_done(prompt_ids, num_turns):
            response_ids = await self.rollout_server.generate(
                request_id=request_id, prompt_ids=prompt_ids, ...
            )
            prompt_ids += response_ids
            tool_response_ids = await call_tool(response_ids)
            prompt_ids += tool_response_ids
            num_turns += 1
            response_mask += ...

            output = AgentLoopOutput(
                prompt_ids=prompt_ids,
                response_ids=response_ids,
                response_mask=response_mask,
                num_turns=num_turns,
            )
            return output
```

Efficient RL with MoEs like DeepSeek-V3-671B

verl is working on supporting efficient RL training for **MoE like DeepSeek-V3-671B/Qwen3-235b**, based on the following features:

- Runnable with 96 H100 GPUs
- Training: **MoE models** based on Megatron, ~0.12 MFU
- Inference: **Multi-node** tensor parallel inference
- Verified reward curve on orz57k & proprietary datasets from community
- Planned: fp8 rollout, gpt-oss-120b (sglang PR #9379)

For more details, please check issue tracker #1033.

Credit to NVIDIA, Amazon, RedNote, etc

Recent Roadmap

- Modular design: composable model engines with better abstraction
  - Algorithm agnostic engine abstraction: FSDP2, Megatron, and more
- Partial rollout & fully-async training pipeline (AReal, Kimi, 2025)
- Rollout performance optimizations (fp8)
- Agentic RL recipes (e.g. SWE-bench)
- Efficient multimodal data transfer via references

Github roadmap issue tracker #2388.