Ray: A Distributed Framework for Emerging AI Applications

Identifying the trend of continuous interaction with environment for future AI application, this paper propose a distributed machine learning system to address new requirement in terms of performance and flexibility \cite{}. This system enables simulation, training and serving for RL applications, providing a unified interface to support task-parallel and stateful computation.

Contribution

1. First distributed framework that unifies training, simulation and serving
2. Unify actor and task-parallel abstractions on top of a dynamic task execution engine
3. A system design principle to help scalability and fault tolerance, which separate lineage recording from other system components.
4. Bottom-up distributed scheduling strategy with concerns about data locality and local node loads
5. heterogeneity in time and resource (GPU for training and CPU for simulation)
6. there are shortages in present systems to satisfy the requirement of RL
   1. without naturally support for fine-grained simulation or policy serving
   2. little support for distributed training and serving
   3. not support for training and simulation

Not intend for implementing deep neural networks or complex simulators from scratch -> the system should enable seamless integration with existing simulators and deep learning framework.

1. task scheduler and metadata store which maintains the computation lineage and a directory for data objects
2. Lineage-based fault tolerance for tasks and actors
3. Replication-based fault tolerance for metadata store

Programming and Computation Model

Ray models an application as a graph of dependent tasks that evolves during execution. A task represents the execution of a remote function on a stateless worker. (the actual computation)

The developing experience for the API extension and the reflection about the fault tolerance are insightful to read. The design of GCS is one of the unique ideas proposed in this system for its internal state record and check, demonstrating better debuggability than other production machine learning system. The effort to construct a better usage system is acknowledgeable, including system-wise design and extension awareness. Also, the evaluation manner, which compares different stages with other systems, is illustrative and easy to follow.

Although this system provides an all-in-one solution for machine learning task, especially for reinforcement learning, it lacks uniqueness for addressing problems in machine learning life cycle. There are multiple replacements designed for specific stage challenges and this system is unable to overrun other systems. The evaluation of SGD performance is not persuasive, considering the possible optimization for TensorFlow ecosystem, and the difference for multi-core performance is increasing, leading skepticism for scalability of this proposed system. Also, the influence of this system seems to be much smaller than the other machine learning system, such as PyTorch and TensorFlow. This system is built with actor design pattern, which could suffer from dead lock. And this possible concern is not illustrated in the paper.