Distributed RPC Reinforcement Learning Benchmark

This tool is used to measure `torch.distributed.rpc` throughput and latency for reinforcement learning.

The benchmark spawns one *agent* process and a configurable number of *observer* processes. As this benchmark focuses on RPC throughput and latency, the agent uses a dummy policy and observers all use randomly generated states and rewards. In each iteration, observers pass their state to the agent through `torch.distributed.rpc` and wait for the agent to respond with an action. If `batch=False`, then the agent will process and respond to a single observer request at a time. Otherwise, the agent will accumulate requests from multiple observers and run them through the policy in one shot. There is also a separate *coordinator* process that manages the *agent* and *observers*.

In addition to printing measurements, this benchmark produces a JSON file. Users may choose a single argument to provide multiple comma-separated entries for (ie: `world_size="10,50,100"`) in which case the JSON file produced can be passed to the plotting repo to visually see how results differ. In this case, each entry for the variable argument will be placed on the x axis.

The benchmark results comprise of 4 key metrics: 1. *Agent Latency* - How long does it take from the time the first action request in a batch is received from an observer to the time an action is selected by the agent for each request in that batch. If `batch=False` you can think of it as `batch_size=1`. 2. *Agent Throughput* - The number of request processed per second for a given batch. Agent throughput is literally computed as `(batch_size / agent_latency)`. If not using batch, you can think of it as `batch_size=1`. 3. *Observer Latency* - Time it takes from the moment an action is requested by a single observer to the time the response is received from the agent. Therefore if `batch=False`, observer latency is the agent latency plus the transit time it takes for the request to get to the agent from the observer plus the transit time it takes for the response to get to the observer from the agent. When `batch=True` there will be more variation due to some observer requests being queued in a batch for longer than others depending on what order those requests came into the batch in. 4. *Observer Throughput* - Number of requests processed per second for a single observer. Observer Throughput is literally computed as `(1 / observer_latency)`.

Requirements

This benchmark depends on PyTorch.

How to run

For any environments you are interested in, pass the corresponding arguments to `python launcher.py`.

```
python launcher.py --world_size="10,20" --master_addr="127.0.0.1"
--master_port="29501 --batch="True" --state_size="10-20-10"
--nlayers="5" --out_features="10" --output_file_path="benchmark_report.json"
```

Example Output:

‘‘‘

PyTorch distributed rpc benchmark reinforcement learning suite

master_addr : 127.0.0.1 master_port : 29501 batch : True state_size : 10-20-10 nlayers : 5 out_features : 10 output_file_path : benchmark_report.json x_axis_name : world_size world_size | agent latency (seconds) agent throughput observer latency (seconds) observer throughput p50 p75 p90 p95 p50 p75 p90 p95 p50 p75 p90 p95 p50 p75 p90 p95 10 0.002 0.002 0.002 0.002 4432 4706 4948 5128 0.002 0.003 0.003 0.003 407 422 434 443 20 0.004 0.005 0.005 0.005 4244 4620 4884 5014 0.005 0.005 0.006 0.006 191 207 215 220