Trace Compass scalability

Quoc-Hao Tran with Pr. Michel Dagenais

June 11th, 2021

Polytechnique Montreal

DORSAL Laboratory

Agenda

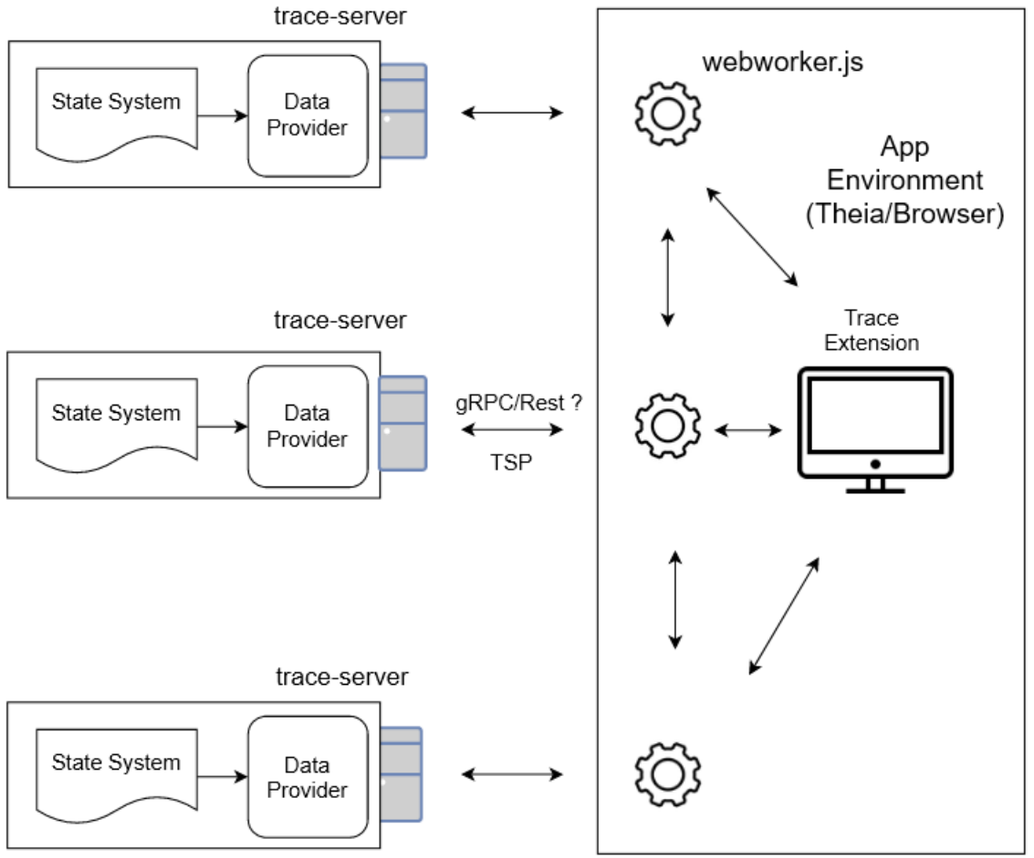
- Background
- Trace coordinator
- Benchmark
- Future work

Background

- Analyzing traces from clusters with high number of nodes
- Aggregated view of group of nodes
- Identifying different components for an appropriate distributed tracing framework

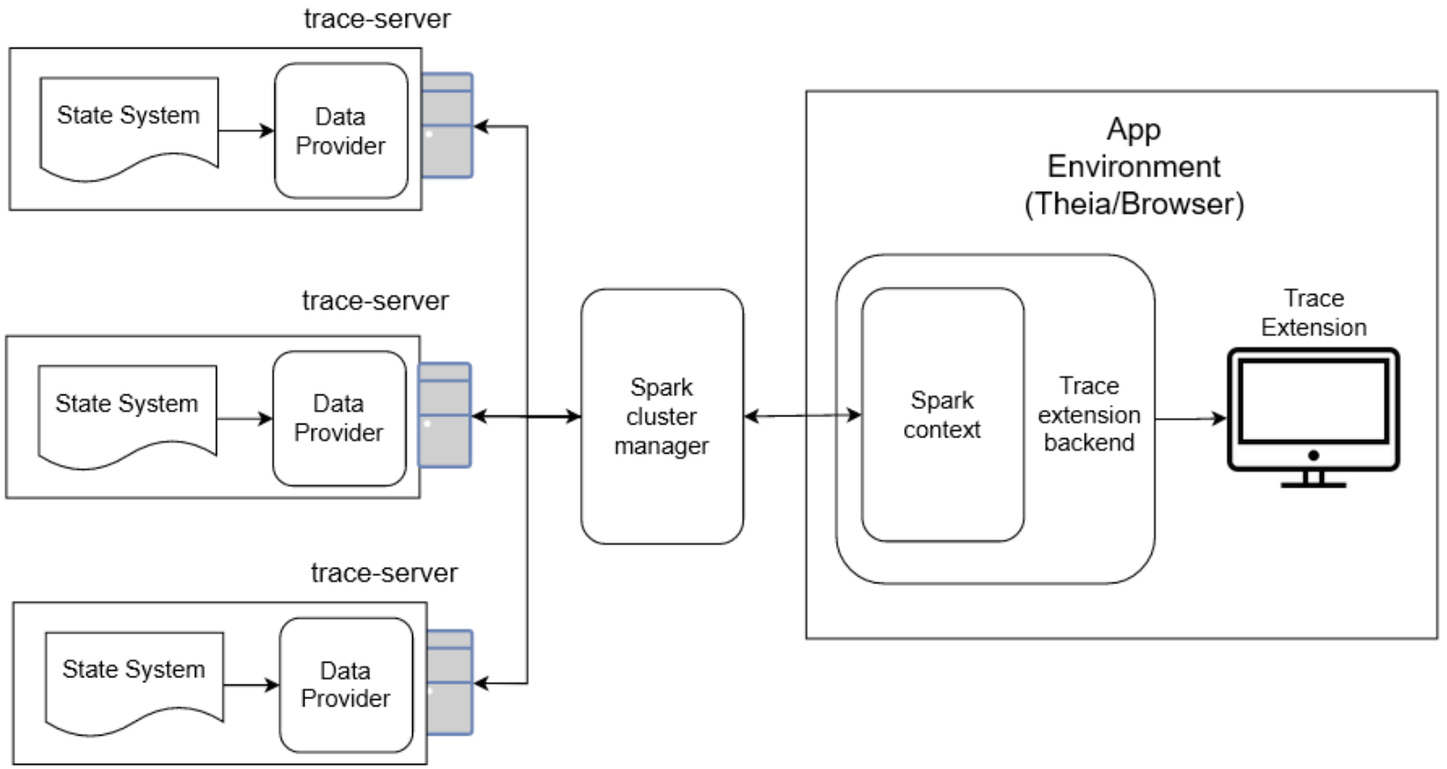
Background

- Use Theia backend or web workers as an aggregator

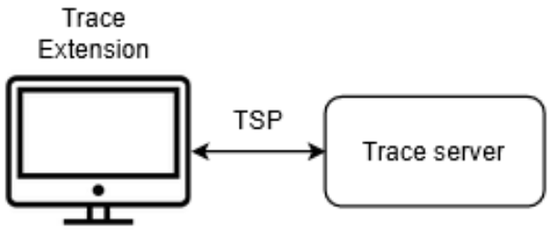


Background

- Use existing framework to handle cluster of trace server

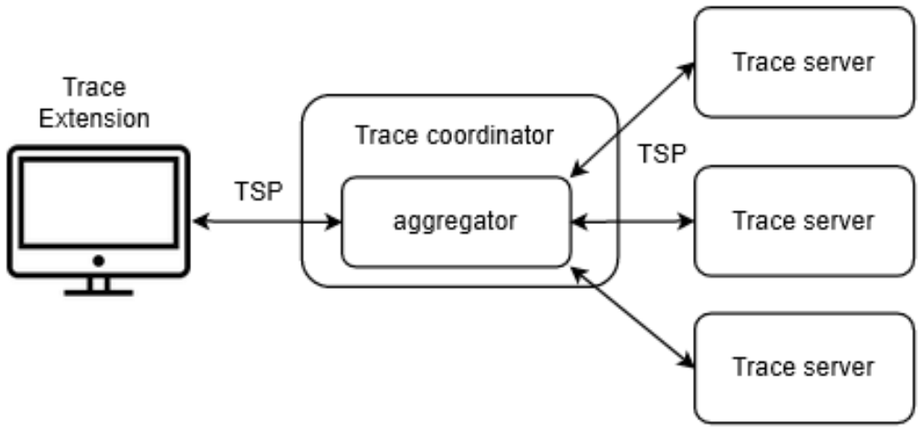


Trace coordinator




```
"parameters": {
    "requested_times": [0, 7777777],
    "entry_id_range": [100000, 200000]
}
```

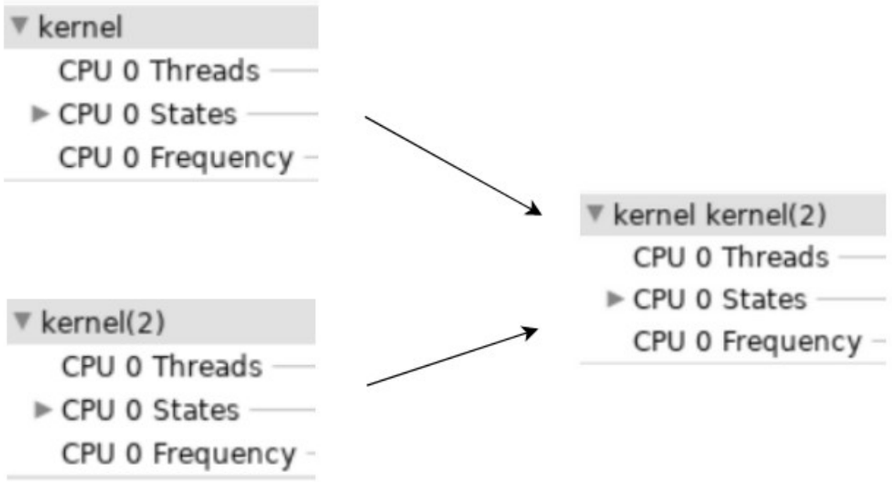
vs



- Offset id on different servers to avoid overlap

Trace coordinator

- Resources view tree



- Cpu usage xy model

Process	TID	%	Time
LIST_LEGACY	2053	3.898 %	3.898 ms
SYNC_UI_UPDATE	2006	3.424 %	3.424 ms

```
"title": "CPU Usage",
"series": [
  {
    "seriesId": 0,
    "yValues": [
      0.0, 71.05076638765084, 2.8670432326743427, 0.0, 299.2677096914385,
```

```
"title": "CPU Usage",
"series": [
  {
    "seriesId": 0,
    "yValues": [
      0.0 / nb of trace-server, 71.05076638765084 / nb of trace-server,
```

Benchmark

- Polytechnique student lab
- 9 trace servers
- 90 traces, ~2gb total
- Core i5 4570 2.9ghz
- 16go memory, hdd

- Indexing time excluded
- Series of query for 3 different views
- Measuring time for each type of query and total time

Benchmark

- First time open view

```
trace-server-time / trace-coordinator-time {
    "cpu_tree": 3.45402709637372,
    "cpu_states": 0.3903864765127581,
    "ram_tree": 4.037338817590531,
    "ram_states": 0.3486329644728335,
    "resources_tree": 0.8161131242145846,
    "resources_states": 0.34172575278328043,
    "total": 2.4860296245640394
}
```

- Gain on querying CPU/Ram usage tree. O(n) aggregation (e.g average % usage)
- Resources view tree O(n^2) aggregation, e.g grouping same label entries (cpu threads, states…etc.) together
- Might be overwhelming trace servers when querying states

Benchmark

- Subsequent open view

```
trace-server-time / trace-coordinator-time {
    "cpu_tree": 1.1976441159528814,
    "cpu_states": 0.4619321116528856,
    "ram_tree": 0.9157599521612699,
    "ram_states": 0.25364370618689314,
    "resources_tree": 0.6570926985777819,
    "resources_states": 0.25832348138068734,
    "total": 0.4801070624703138
}
```

- Gain dismissed when tree is cached because there was no cache in the coordinator

Future work

- Investigate the behavior when querying different types of models
- Aggregate partly on trace-server coordinated by a central node – e.g appropriate tree model depends on type of query from the coordinator
- Integrate with Spark
- Multi-tier trace server/coordinator

The End

Thank you for your attention

Q & A