Mini Project: System Adaptive Naiad

Luca Choteborsky

Background Context

Naiad: Timely Dataflow

- Uses a modified dataflow computational model
- Attaches timestamps to messages to track progress through dataflow graph
- Runs on a cluster that consists of a group of processes hosting workers that manage a partition of the timely dataflow vertices
- Uses a stateful workers to coordinate global progress and ensure correct delivery of notifications

Background Context

Limitations

- At run-time, logical graph is expanded into a static physical graph
  - Assumes a fixed number of processes
- Does not mention how work is scheduled across heterogeneous architectures
- Does not allow faulted processes to rejoin
  - Previously owned vertices are reassigned to the remaining processes

Not suitable in a cloud environment

- Machines may be reassigned or fail
- Issue raised https://github.com/TimelyDataflow/timely-dataflow/issues/529

Plan

Goal: Modify Naiad to adapt to dynamically changing resources

- Adding and removing workers is difficult as much of the shared state they use to coordinate relies on static assumptions
- Rework the model of workers and their operations as async tasks that can themselves farm out work to an elastic set of workers
  - Have each worker be assigned a machine (or a set of them)
  - The operations are async tasks which the worker's machine will execute
  - Various techniques can be used to test VM reassignment or failure (i.e. heartbeat protocol)
  - Machine allocations can be performed centrally or using a consensus

Plan

Evaluation

- Evaluate performance on a dynamically changing cluster
  - Will struggle to host a minikube cluster on my laptop
  - Likely to carry out in a cloud service such as GCP
  - Measure throughput when artificially adding and removing devices
- Compare against standard Naiad configuration
  - Will need to look into how Naiad is configured for a GCP cluster

- Extension: Compare throughput of different methods
  - Centralised vs Consensus
  - Different methods to detect machine failure

Work Completed

Work Done So Far

- Thinking