Latency-Driven, Application Performance-Aware, Cluster Scheduling
Diana Andreea Popescu, Andrew W. Moore
University of Cambridge
Contact: diana.popescu@cl.cam.ac.uk

Motivation

- Network latency variability is common in multi-tenant data centers, leading to performance variability [1,3]. Even small amounts of delay, in the order of microseconds, may lead to significant drops in application performance [1].
- For example, we obtained different performance values for Memcached in different data centers, and in the same data center at different times after restarting the VMs.
- We place the applications according to how latency-sensitive they are, and to the current measured latency in the data centre, which is not constant [13]. If latency increases, the application may be migrated.

Modeling Application Performance

- We studied the effect of network latency on application performance, as defined for a certain application.
- We did this by artificially injecting arbitrary network latency into a networked system using a bespoke hardware appliance [1.2].
- We fit a curve to the observed results to find p(injected latency)~ normalized application performance metric, where p is the performance.
- For the small latency values the model can be assimilated to a constant function whose value is the baseline performance.

NoMora Cluster Scheduling Policy

- NoMora architecture:
  o Functions that predict application performance dependent upon network latency;
  o Network latency measurement system (Pingmesh [9], PTPmesh[10]);
- Flow network: T - task of a job, R - rack, M - machine(host), X - cluster aggregator, U - unscheduled aggregator, S - sink, C - number of cores on a machine; a, b, c, d costs on arcs
- Jobs: have a root task (the server/the master and the clients/workers)
- Placement algorithm:
  o the root task is scheduled on any available machine (the root task is assigned a single arc to the cluster aggregator, with a cost of 0);
  o if a task that is not a root task enters the system at the same time as the root task, or before the root task is scheduled, it will not be scheduled, waiting instead;
  o if the root task is scheduled, then a new task’s placement is determined based on the application performance prediction, and current network latencies to the root task’s placement.

NoMora Evaluation

- Simulation setup:
  o Google cluster trace [12]
  o Network latency measurements from [13]
- Topology - number of hosts per rack 16, number of racks per pod 48
- Evaluation metrics:
  o Average application performance: measures task placement quality;
  o Algorithm runtime.
- Average application performance improves by up to 13.4% and by up to 42% if migration is enabled, compared to the baselines.
- The task placement latency improves by a factor of 1.79x and the median algorithm runtime by 1.16x compared to the baselines.

---

[2] Balancing the network load on multi-tenant data centers by using a hardware-based latency injection appliance, Manjukumar Tharakan, Dieter van Hee, 2017
[7] PTPmesh: Data center network latency measurements using PTP, Diana Andreea Popescu and Andrew W. Moore, IEEE MASCOTS 2017
[8] Pingmesh: A Large-Scale System for Data Center Network Latency Measurement and Analysis, Guo et al., ACM SIGCOMM 2015
[10] PTPmesh: Data center network latency measurements using PTP, Diana Andreea Popescu and Andrew W. Moore, IEEE MASCOTS 2017
[12] A Fine Look at Data Center Network Conditions Through The Eyes of PTPmesh, Diana Andreea Popescu and Andrew W. Moore, JOINTES 2018

This work was supported by the EU FP7 METRICS ITN (grant number 607728), EU Horizon 2020 research and innovation programme 2014-2016 under the SSICLOPS (grant agreement No. 644686).