ECSched: Efficient Container Scheduling on Heterogeneous Clusters

Yang Hu, Cees de Laat and Zhiming Zhao
System and Network Engineering Lab, Informatics Institute, University of Amsterdam

Ecsched

ECSched, an efficient container scheduler that can make high-quality and fast placement decisions for concurrent deployment requests on heterogeneous clusters. ECSched maps the scheduling problem to a graphic data structure and model it as minimum cost flow problem (MCFP). In the model, the capacities and costs on edges encode the container deployment requirements of resources and affinities. ECSched can compute the optimal solution online based on classical MCFP algorithms and problem-specific optimizations. In the evaluation, we show that ECSched exceeds the placement quality of state-of-the-art container schedulers with relatively small overheads.

Features

Features of container-based infrastructure:

- Heterogeneous Cluster Configuration
- Flexible Resource Allocation
- Application-Oriented Specification

Requirements

Requirements of container deployment requests:

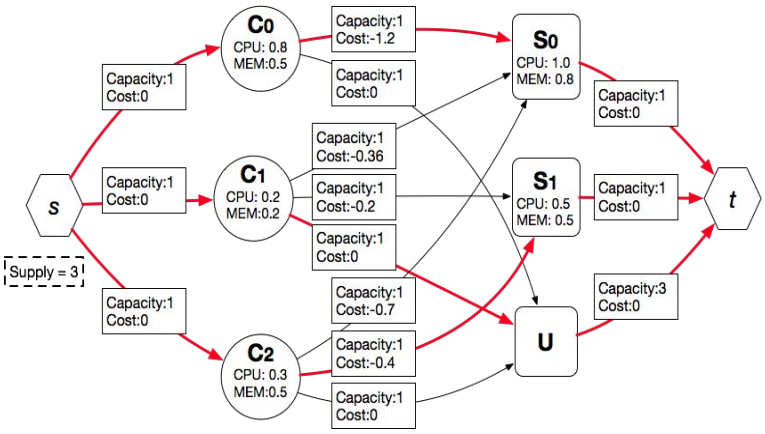
- Multi-resource Demands
- Server Affinity (Data Locality)
- Container Affinity (Data Communication)

Approach

To map the container scheduling problem to MCFP, ECSched represents it using a flow network, which can be described as follows:

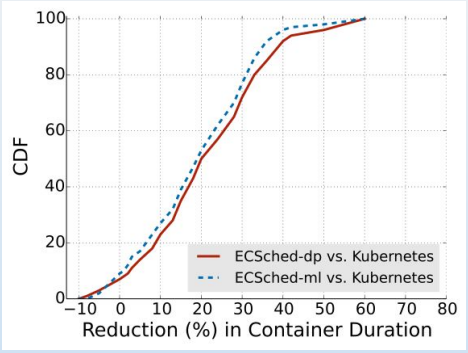
- **Source Node *S*:** with a supply K (K = 3 in example)
- **Container Node *Ci*:** has an edge from S with capacity 1
- **Server Node *Si*:** has an edge from Ci with capacity 1 (eligible)
- **Unscheduled Node *U*:** Ci have an outgoing edge to U with capacity 1
- **Sink Node *T*:** Si have an edge to T with capacity 1 U have an edge to T with capacity N (number of containers)

ECSched encodes container resource demands by flexibly assigning costs on the edges. There are two strategies for cost assignment in ECSched: dot-product heuristic (dp) and most-loaded heuristic (ml).

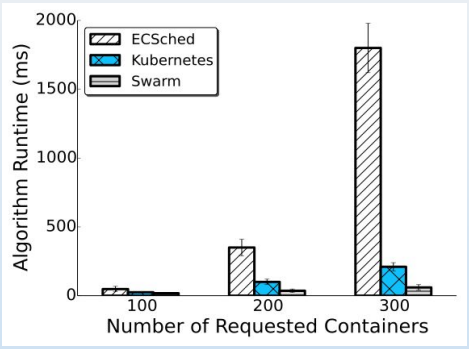


Evaluation

Compared to the state-of-the-art scheduler in Kubernetes, ECSched reduces container duration significantly. It reduces 20% at median, and the top twentieth of containers improve by over 35%.



Evaluated in 1000-machine cluster, ECSched can respond in sub-second time when processing 100 containers concurrently, and respond in about 1.8 seconds for 300 containers. The overhead is relatively small and acceptable in practice.



Yang Hu <y.hu@uva.nl>, Cees de Laat <delaat@uva.nl>, Zhiming Zhao <z.zhao@uva.nl>
https://ivi.fnwi.uva.nl/sne/

Universiteit van Amsterdam
System and Network Engineering
SWITCH
ENVRI
VRE4EIC