Resilient QoS-Aware Workload Co-Scheduling and User Anomaly Detection in Shared Computing Clusters

Renyu Yang and Jie Xu, School of Computing, University of Leeds

Introduction

Background

Workloads

- **Batch jobs**: run-to-complete (one-off) jobs that run pre-defined processing actions without user interactive input and normally have pipelines and parallelisms determined by parallel programming models. Examples include Map reduce, Spark, Tez, ML jobs, etc. *[short-lived]*
- **Long-running applications (LRAs):** interactive services that are typically user-facing and web-serving, with tail latency requirements. Examples include database services, microservices, stream jobs, some online etc. *[From hours to days]*

Users

- benign users and abnormal users including malicious users (e.g., malware), bot users (make attempts to behave like human), zombie users, spammers, etc.

Resilience in Shared Cluster Management

- To co-schedule a mixture of batch jobs and LRAs onto multi-dimensional computing resources, with an assurance of runtime quality of service (QoS) and batch jobs
- To protect benign users from malicious behaviours, and effectively detect anomalies in a timely and cost-effective manner

Limitations of SOTA Approaches

- Existing schedulers are mostly resource-centric, not QoS-centric. Detecting and mitigating QoS violation become more intractable due to the network uncertainties and latency propagation across dependent microservices
- Existing works on LRA scheduling are often application-agnostic, without particularly addressing the constraining requirements imposed by LRAs, such as co-location affinity constraints and time-varying resource requirements
- Existing methods for malware detection often fail to cope with evolving camouflage and attack types that are increasingly difficult to be differentiated from the benign users, and become siloed and subject to the amount, shape, and quality of platform-specific data

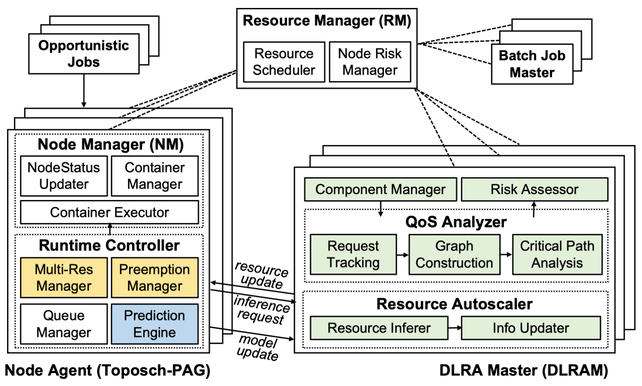
Research Objectives

- To develop a new scheduling framework to prioritize the QoS of DLRAs whilst balancing the performance of batch jobs and maintaining high cluster utilization
- To investigate robust and optimised algorithmic solution to deal with a number of complex co-location affinity and temporal multi-resource constraints
- To devise novel mechanisms for better detection of user anomalies in the face of hidden malicious behaviours and inadequate labelled samples

Methodology

QoS-Aware Co-scheduling for Distributed LRAs and Batch Jobs [1]

System Architecture



[Toposcaling Architecture]

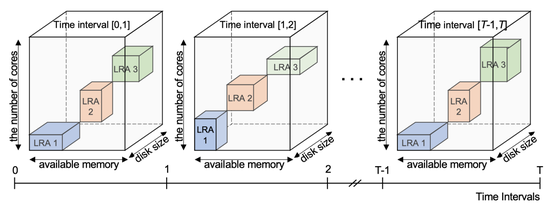
Technique Highlights

- Tracked footprints of every single request across microservices and constructed e2e latency graph
- Used critical path analysis to identify microservices with high risk of QoS violation
- Conducted risk assessment at microservice and node level
- Intervened batch scheduling by adaptively reducing the visible resources to batch tasks to give way to DLRAs.
- Used a prediction-based vertical resource auto-scaling mechanism, via resource-performance modelling and fine-grained resource inference, for prompt QoS recovery

Affinity-Aware Resource Planning-Ahead for LRAs [2]

System Model

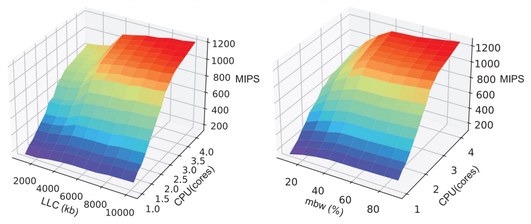
- Multi-dimensional time-varying resource requests
- App-specific affinity constraints, i.e., placement preferences or exclusions when co-located with other LRAs
- Incorporating in the deployment plan a huge number of resource and affinity requirements



[Multi-resource request changed over time]

Algorithm Suite

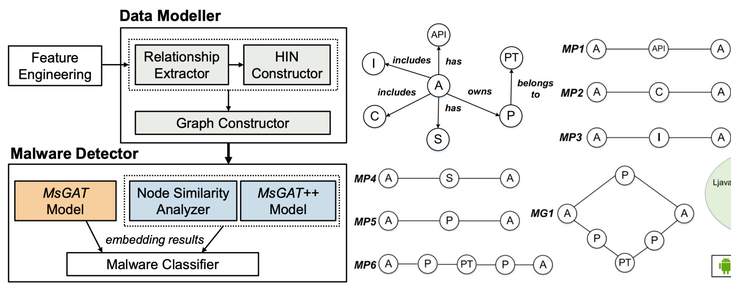
- Minimised the compute nodes in use
- Investigated a broad range of algorithms including Application-Centric, Node-Centric, and Multi-Node approaches, and tune them for large-scale real-world scenarios



[Online learning of QoS sensitivity to multi-resources]

Rapid and Adaptive GNN-Based User Anomaly Detection (incl. Malicious and Bot Accounts) [3-5]

Overall Design



[Basic Pipeline]

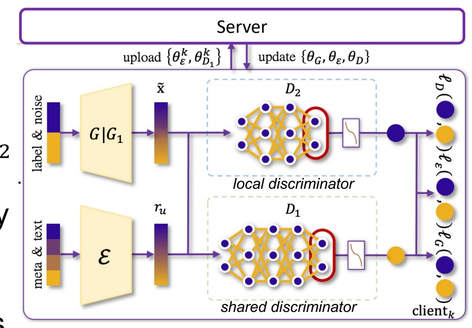
[Meta-schema, meta structure (meta- paths and graphs) and a toy example of HIN]

Technique Highlights

- Modelled system entities and their behavioural relationships as a HIN *(Heterogenous Info Network, aka. Heterogenous Graph)* for specifying implicit higher order relationships
- Used MSGAT, an enhanced GAT, to aggregate neighbours' embeddings at both intra- and inter- meta-structure level
- Developed an incremental learning model, MSGAT++, to pinpoint the similarity between a new sample and existing ones

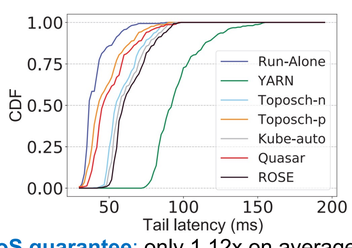
Model Enhancement

- Federated knowledge distillation (KD)
- A global generator G for KD and a local generator G_k for data enhancement
- D_1 shared among clients but with local data; D_2 customised for individual platforms
- A multi-stage adversarial mechanism for jointly optimizing classification in D_1 and D_2 at intra-client level and a contrastive mechanism for aligning different feature spaces across clients
- DRL based GNN architecture search

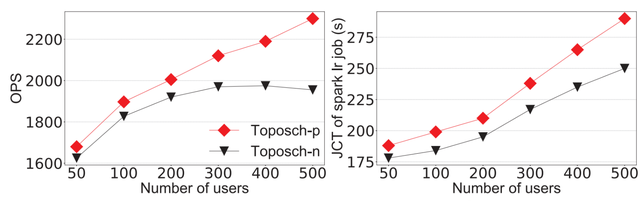


[Federated GAN-based knowledge distillation]

Results

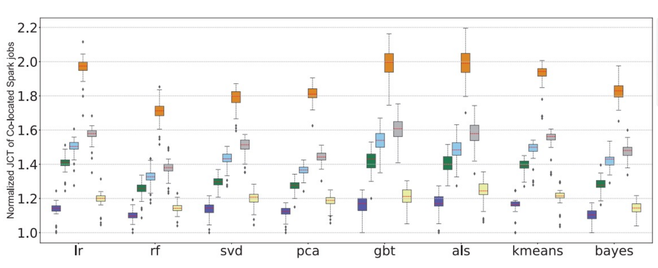


QoS guarantee: only 1.12x on average (1.05x ~1.19x) compared with the case of Run-Alone

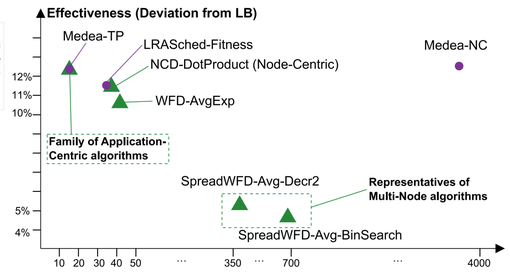


Performance of MongoDB co-located with Spark `lr` jobs

Autoscaling: the OPS can increase proportionally, matching the growing user demands.Batch jobs give ways to prioritized LRAs' QoS

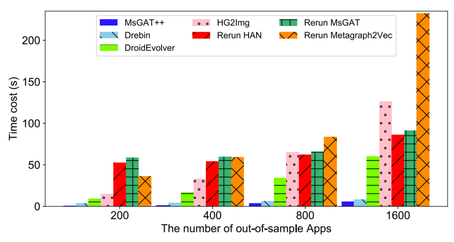


Batch's compromising: compared with native YARN, TOPOSCH-n and TOPOSCH-p result in 17% and 26% average increase, respectively



Quality-time tradeoff: comprehensive study on different algorithm categories

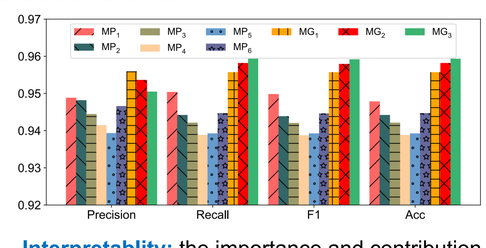
Metrics	Approaches	20%	40%	60%	80%
$F1$	Node2Vec	0.8355	0.8378	0.8542	0.8601
	GCN	0.8653	0.8677	0.8721	0.8763
	GAT	0.8435	0.8633	0.8752	0.8801
	Metapath2Vec	0.9231	0.9321	0.9328	0.9395
	RS-GCN	0.9212	0.9510	0.9515	0.9560
	RS-GAT	0.9507	0.9631	0.9653	0.9664
	HAN	0.9511	0.9617	0.9671	0.9705
	Metagraph2Vec	0.9750	0.9766	0.9764	0.9771
	SVM (Drebin)	0.9312	0.9387	0.9446	0.9477
	DroidEvolver	0.9412	0.9517	0.9566	0.9605
	HinDroid	0.9643	0.9669	0.9684	0.9746
	MatchGNet	0.9395	0.9511	0.9604	0.9753
	Aidroid	0.9321	0.9399	0.9414	0.9455
	MsGAT (Hawk)	**0.9857**	**0.9859**	**0.9871**	**0.9878**



Time efficiency: MsGAT outperforms other approaches in out-of-sample detection

Dataset	Vendor-19			
Setting	$\alpha = 1$	$\alpha = 0.5$	$\alpha = 0.1$	$\alpha = 0.05$
FedAvg	71.30±0.60	61.06±1.52	60.88±2.85	59.81±2.48
FedProx	84.37±0.43	78.25±1.02	51.86±0.04	63.27±2.32
FedDF	86.37±1.23	80.17±2.21	63.16±1.37	67.01±1.78
FedEnsemble	81.12±2.22	76.70±1.21	64.51±2.56	68.05±1.15
FedDistill	79.68±0.58	68.77±1.13	52.88±0.06	70.25±0.39
FedGen	90.05±0.33	84.83±0.96	65.12±0.60	70.79±2.39
FedFTG	88.31±1.41	82.17±1.52	66.01±1.25	68.39±1.94
FedACK-A	**91.31±0.52**	84.79±1.05	66.10±2.90	68.21±1.95
FedACK	88.58±1.91	**87.05±2.03**	**76.04±3.40**	**75.27±2.50**
Gain	↑ **1.26~20.01**	↑ **2.22~25.99**	↑ **10.03~24.18**	↑ **4.48~15.46**

Accuracy of different methods on a) MsGAT and b) federated KD enhancement



Interpretablity: the importance and contribution of different meta-structure to the embedding

Main Collaborators include Xiaoyang Sun, Dr. Clément Mommessin, Dr. Natasha Shakhlevich (Leeds), Dr. Hao Peng, Dr. Jianyong Zhu, Dr. Tianyu Wo, Prof. Chunming Hu, Yiming Hei (Beihang University), Jin Ouyang, Junqing Xiao (Alibaba Group), Yingguang Yang, Dr. Pengyuan Zhou, Prof. Yong Liao (USTC), Prof. Albert Y. Zomaya (Sydney), etc.

Key Outputs supported in part by UK **EPSRC Grant (EP/T01461X/1), Alan Turing Pilot Project, and Alan Turing PDEA Programme**:

[1] QoS-Aware Co-Scheduling for Distributed Long-Running Applications on Shared Clusters. *IEEE Trans. on Parallel and Distributed Systems, 2022*
[2] Affinity-Aware Resource Provisioning for Long-Running Applications in Shared Clusters, *Journal of Parallel and Distributed Computing, 2023*
[3] Hawk: Rapid Android Malware Detection through Heterogeneous Graph Neural Network. *IEEE Trans. on Neural Networks and Learning Systems,2021*
[4] RoSGAS: Adaptive Social Bot Detection with Reinforced Self-supervised GNN Architecture Search. *ACM Trans. on the Web, 2022*
[5] FedACK: Federated Adversarial Contrastive Knowledge Distillation for Cross-Lingual and Cross-Model Social Bot Detection. *ACM WWW, 2023*

Any Correspondence to Dr. Renyu Yang (r.yang1@leeds.ac.uk, renyu.yang@buaa.edu.cn). **More Details at** https://yangrenyu.github.io/