ART: Learning Operation Tree Patterns for Cloud Remediation

Hongyi Huang*
Tsinghua University

Wenfei Wu
Tsinghua University

Abstract

Since IT service becomes overwhelmingly large-scale and complicated, operation trees running on automated remediation systems are put forward to offload repeated remediation off the operators. In general, all faulty scenes require dedicated operation trees. However, it's tedious work to build all trees from scratch. In this poster, we propose ART, a scheme to learn instinct patterns from existing operation trees and automatically generate trees in demand. And we prove that it would greatly mitigate mitigate the hardness of tree setup.

1 Introduction

As IT service grows in a larger scale and more complex, operation trees are raised to take place of IT operators in repeatable remediation tasks[2, 3]. Operation trees are tree-organized, configured with certain rules that determine the workflows and actions that execute workflows on infrastructures, and in always-running states. Even though some tools offer graphical interfaces to accelerate designing trees, they barely give deep insights into design patterns, which never ceases redundant construction and configuration.

To move forward far-and-wide deployment of automated remediation in cloud, we devise ART, a scheme to learn instinct and structural patterns from existing operation trees and automatically generate trees in demand in an advisory capacity. Although Operator Pattern[1] in Kubernetes captures workload of operators to automate repeatable tasks, nevertheless, it's defined in low-level codes manually and lacks support for self-learning and reusability.

At this point, main challenge is *how to characterize the structural patterns and make them reusable in incoming designs.*

Hence, to make it clear, we carry out an analysis of existing operation trees, and we observe that some trees have similar logical flows in structure. Inspired by the analysis, we introduce Tree Edit Distance (TED)[4] metric, which is used to measure similarity of tree structured data. We redefine operations in APTED to better characterize the structural similarity between trees in our case. And we apply enhanced-APTED to each pairs of de facto trees to capture structural patterns. Exploited *similar trees* are classified into several clusters based on TED value, and representative trees (namely patterns) are elected in per-cluster manner and stored into the database. At the same time, we assign topics, which are keywords extracted from description of trees, as *indexes* to patterns for future retrieval and record appearance

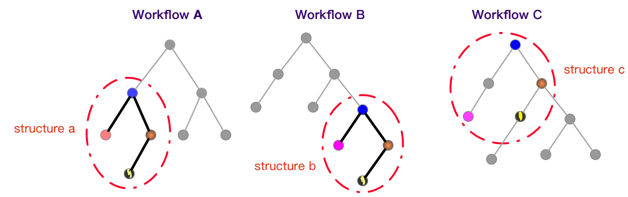

Figure 1. Workflows with similar structural pattern

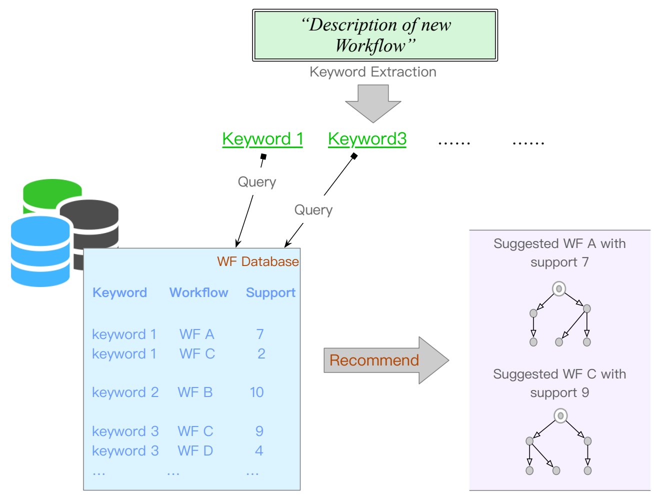

Figure 2. Mining workflow patterns for construction

of pattern workflows as *support*. Consequently, if operators is building up a new tree from scratch, as Figure 2 shows, ART would retrieve patterns based on the keywords that are extracted from description of the new tree and recommend those matched patterns of high support.

Beyond inferring from description, if a pending tree has partial structures completed, ART could decompose these structures and find a match against any prefix[1] of patterns by APTED. If successful, suffix of the matched pattern (remaining sub-tree) could be attached to that tree, which performs similar to context-based word prediction.

References

[1] Operator pattern. https://kubernetes.io/docs/concepts/extend-kubernetes/operator/.
[2] Stackstorm. https://stackstorm.com.
[3] Justin Meza, Tianyin Xu, Kaushik Veeraraghavan, and Onur Mutlu. A large scale study of data center network reliability. In *Proceedings of the Internet Measurement Conference 2018*, pages 393–407. ACM, 2018.
[4] Mateusz Pawlik and Nikolaus Augsten. Efficient computation of the tree edit distance. *ACM Transactions on Database Systems (TODS)*, 40(1):3, 2015.

*Student and presenter.

[1]Prefix of tree is defined as a sub-tree including starting node.