Capturing, Indexing, Clustering, and Retrieving System History by Cohen et al.

CSC 724 Paper review - Vaibhav Singh, vsingh7

January 29, 2019

1 Summary

The paper presents ways to extract ***signatures*** from a system to identify whether a particular system state is similar to a pre-observed state. This allows owners to draw parallels between different system states, and between different sites of the same distributed system, to better diagnose problems. The paper does this by collecting the system's metrics of merit (say, average transaction time), and low level metrics like CPU usage, to cluster seemingly different incidents together and present a **"*syndrome*"** of a group of problems.

2 Description

The algorithm presented in the paper assumes a set of reference metrics (throughput, response time), and a corresponding set of thresholds, which together form the Service Level Objective or SLOs for the system. The paper uses automated (L1 norm k-median and L2 norm k-means) clustering to group together SLO violations (by finding out instances where SLO metric deviations were more than the corresponding threshold) and figure out a collection of metrics which together forms the syndrome (or symptom) of the problem.

The signatures are constructed by picking a subset of models from an ensemble, which historically had high accuracy in determining SLO state. The models are then segregated based on whether they were "abnormal" or "normal" during periods of SLO violations.

3 Strong Points

The paper is arguably the first attempt at attempting to propose a technique for constructing the appropriate representation of states in a system, and evaluating its efficacy.
The paper is (apparently) the first paper to automate generating system representation states and facilitating syndrome identification and incidence clustering.

4 Weak Points

The paper might not be very helpful in root causing an issue to a particular *new* faulty state in the system, for example, a code bug with memory leaks causing high memory usage in the system.
A clustering based algorithm is good for "catch-all" problems, where apparently similar issues can be clubbed in one group, but even for old issues the paper might be too generic for any real root causing.

5 Improvement

The paper's ideas of capturing and indexing faults in the system should be complemented by a more "hands-on" approach of say, running static and dynamic analysis to catch problem areas in code.