Review of

S. Ananthanarayanan, M. Ardekan, D. Haenike, S. Sorian, D. Pate, A. Adl-Tabatabai, and B. Varadaraja

"Keeping Master Green at Scale"

in *Proceedings of the 14th EuroSys Conference* Dresden, Germany (2019)

T.H. Tse
Department of Computer Science
The University of Hong Kong
Pokfulam, Hong Kong

Huge monolithic repositories, also known as monorepos, are popular in leading technology companies such as Microsoft, Google, and Facebook. They support continuous integration, merging all the developers' versions to a shared master version, also known as the mainline, as frequently as hundreds of times per hour. The need for effective strategies to store the code for a large number of projects, build the systems, and maintain revision control is a challenging issue.

This paper presents SubmitQueue, a revision control system in Uber, to oversee the continuous integration of changes at scale while ensuring that the mainline is always green, that is, making certain that all the build steps execute successfully at every commit point. Rather than exhausting every possible change each time, SubmitQueue applies a probabilistic regression model with a greedy best-first algorithm to select changes that are envisaged to succeed, and skips those already executed in previous builds.

The authors have evaluated SubmitQueue by comparing against three other approaches: (a) speculate all, which exhausts every possible combination; (b) single queue, which builds interdependent changes sequentially and independent changes in parallel; and (c) optimistic speculation, which starts to build selected changes under the assumption that not-yet-completed changes will be successful. They have also assessed SubmitQueue through an internal survey of developers, release engineers, and managers. The results are more than acceptable.

This paper provides a meticulous analysis of the revision control issue. It is recommended for all release engineers. Having said that, I do have a word of concern. I appreciate the pragmatic philosophy of typical engineers, namely, ''if it works, use it.'' As an academic researcher, however, I do not fully share the authors' overconfidence on the quality assurance of the proposed approach. We know in regression testing that integration tests must be rerun during reiterations. The repeated use of the word ''guarantee'' for all build steps while skipping the re-execution of previous builds is overly optimistic.