Workload Characterization and Performance Improvement for SAT Solvers

In computer systems, a workload is the amount of computing resources and time it takes to complete a task or generate an outcome. Any application or program running on a computer can be considered a workload. System performance can be precisely evaluated by understanding the system's workload. Workload characterization is the process of precisely describing the system's global workload in terms of its main components. It provides profiling of the requests or components that place different demands on the system resources. This study helps us identify the issues and concerns for the pattern of workload and signify the options to explore in improving the performance of the workload.

We will be characterizing the workload for SAT solvers in this project. A SAT solver is an algorithm for establishing satisfiability. It takes the Boolean logic formula as input and returns if it finds a combination of variables that can satisfy it or if it can demonstrate that no such combination exists. The SAT problem is the first discovered NP-complete problem, and is also the core of a large class of NP-complete problems. Therefore, solving the SAT problem plays an important role in the study of artificial intelligence systems and computational theory.

A competition is organized for writing SAT solvers and benchmarks for the same. [1] We aim to analyze the behavior of these SAT solvers against the provided benchmarks using perf tool and capture events like cache misses, branch mispredictions, no. of cycles, no. of instructions etc. From the analysis of solvers with benchmarks, we will select a few complex solvers which take many cycles and instructions to execute. These selected solvers will be simulated in ChampSim using their traces generated by the pin tool to create the baseline design. We then study and explore the changes required in ChampSim to improve the performance of the baseline design for the selected SAT solvers.

References

[1] https://satcompetition.github.io/2020/index.html