Stride Scheduling: Deterministic Proportional-Share Resource Management – Critique

- Vidit Jan (862394973)

Summary:

The novel CPU scheduling method presented in this work is based on the idea of proportional sharing. The shortcomings of conventional priority-based scheduling algorithms are discussed, along with how they lead to inefficient resource allocation and unfair CPU time distribution across clients. The proposed method, which makes use of the proportional-share scheduling concept, is presented in this study. Stride Scheduling's main principle is to assign CPU time to each task in proportion to their weight or share, which is determined by a parameter known as the "stride." The sequence in which processes are scheduled is determined by the stride value that is given to each process. Overall this scheduling policy provides a more deterministic allocation of CPU resources compared to other policies.

Strengths:

One strength of stride scheduling is accuracy and fairness over the probabilistic approach proposed in lottery scheduling. The authors have also modified the ticket compensation policy from the lottery scheduling framework to accommodate request from clients. The extension would allow clients to specify the quantum size they require. Overall, the stride scheduling is able to significantly reduce the error between expected scheduled time and actual scheduled time. The authors demonstrated that the error was never greater than one time quantum, compared to root(n) error after n allocations. Overall, this is able to leverage the concepts of market economics using lottery tickets, as well as ensuring deterministic runtimes using abstractions such as stride, pass and tickets. The performance comparison was also comprehensive, presenting comparisons against other policies such as round robin.

Weaknesses:

The paper presented only simulated results, and not real-world performance. Also, as mentioned by the authors, the algorithm might not be very effective where there are high variations in resource usage. Stride scheduling is also more complex than lottery scheduling, requiring meticulous dynamic changes state updates. This may lead to more scheduling overheads, which was not discussed at length in the article.

Other comments:

The proposed algorithm is a noteworthy upgrade over the lottery scheduling policy. It helps alleviate quite a few issues of the lottery scheduling policy. The most important being, stronger guarantees for deterministic resource allocation. The authors also described the implementation code in great detail and coupled with abstractions such as stride, pass and tickets, this article was very easy to read.