Operating Systems Design: Critique

2019712600
Oh, Seungmin

- **Everything You Always Wanted to Know about Synchronization but were Afraid to Ask**

  Recently, the importance of synchronization has increased as distributed systems and multi-core systems are taken for granted. Nevertheless, recent studies on synchronization yield only analysis results for specific modern systems or specific applications. Therefore, we need an analysis result that is general and exhausted.

  The advantage of this paper is that result is general and exhausted. While the existing studies only confirmed specific cases, environments, and criteria to analyze the performance in their target environment, this paper gained generality by studying sockets, cores, and various other environments. In addition, it is more thorough than other studies as analyzed in various environments and factors. Furthermore, they have created their own cross-platform synchronization suite. On the other hand, the limit is that there are few explanations about the power consumption, file system overhead problems that can occur during sync, and lock problems.

  If additional content is written here, it would be good to know the filesystem layering overhead, lock issues, and power considerations that can occur in sync.

- **Read-Log-Update**

  RCU is a useful synchronizing mechanism that can synchronize multiple reads at the same time. However, the RLU can run on more writers, bringing performance benefits in many ways.

  The advantage of this paper is that it changes the framework itself, which was considered the most basic rule. There are advantages such as usability and time difference between reading and writing, but there is a big difference in being able to support multiple writers decisively. On the other hand, RLU is not convenient in all cases and there is a problem that the stability of copy method can be more reliable than log format.

  In the following paper, it would be good if the RLU and RCU should be used separately, and the methodology of using them properly mixed.

- **The Scalable Commutative Rule: Designing Scalable Software for Multicore Processor**

  The scalability of the system has recently emerged as an important issue in multicore systems. However, it is very difficult to judge how much more scalable it is than it is today. This paper proposes a new rule and tool for judging system scalability at high level interface.

  The advantage of this paper is that not only the scalability is determined by the experimental value, but also the interface can be used to determine how scalable it is. They also make COMMUTER which can evaluate scalability. On the downside, it would have been nice to have a detailed explanation of how well the scalability was actually evaluated compared to the experimental values.

- **An Analysis of Linux Scalability to Many Cores**

  This paper analyzes scalability on seven Linux systems with different characteristics. With this analysis, we propose a new technique.

The beauty of this paper is not just analysis, but new techniques, and how each system should go. On the other hand, the downside is that the systems are limited and the techniques that are common to all systems are not general.

In future papers, it may be useful to separate more systems according to scalability-critical criteria and to analyze the technique of each system.