CS423 Summary: The Linux Scheduler: a Decade of Wasted Cores

Hongpeng Guo

February 15, 2019

Area:

This paper investigated the scheduling mechanism of Linux operating system. Several bugs which will cause core waste were pointed out by the authors. The corresponding fix methods or tools were also presented in this paper. Overall, this paper lies in the area of operating system scheduling.

Problem:

While the Linux system tries to schedule multiple threads over processors, there are often cases that the CPU remains idle for several milliseconds when ready threads are already there. The accumulation of such idle time finally cause a waste of CPU processing ability. This paper investigated this scheduling problem and provided corresponding solutions and debugging tools.

Methodology:

The authors first studied the load-balancing scheduling algorithms of Linux OS. Based on the investigation of the above algorithms, Several "idle causing" bugs were identified, such as the group imbalance bug, the scheduling group construction bug, etc. Finally, the corresponding fix methods and a set of debugging tools were provided for developers to handle the scheduling problems.

Solution:

For each type of scheduling bugs, the authors provided the corresponding fix methods:

- The authors changed the part of the algorithm that compares the load of scheduling groups to fix the group imbalance bug.
- The construction of scheduling groups were modified in this paper to fix the scheduling group construction bug.
- etc...

Besides the specific fix method for certain bugs, the authors also provided a set of debugging tools for developers to handle the scheduling inefficiency. The tools include (1) Online Sanity Checker and (2) Scheduler Visualization tool.

Results:

The result of this paper is impressive. One barrier heavy scientific application ended up running 138 times faster. 2 Kernel make and a TPC-H workload on a widely used commercial DBMS improved performance by 13% 14% respectively. The TPC-H query most affected by the bug sped up by 23%

Takeaway:

(1)The core module should be able to take suggestions from optimization modules and to act on them whenever feasible, while always maintaining the basic invariants, such as not letting cores sit idle while there are runnable threads.

(2) It is important to develop visualization tools in system performance analysis.