Using SATF Scheduling in Real-Time Systems

Lars Reuther Martin Pohlack
Dresden University of Technology
{reuther,mp26}@os.inf.tu-dresden.de

1 Introduction

Disk scheduling algorithms based on the rotational position of the disk head (*Shortest Access Time First scheduler, SATF*) are known to be a good approximation of an optimal disk scheduling algorithm [3]. However, because of the required knowledge of the disk they are also believed to be difficult to implement outside of the disks firmware at driver level [2].

Unfortunately, scheduling cannot always be done in the disk drive. With queueing a large number of requests in the disk drive, the driver resp. the operating system loses the ability to control the point in time a single request is executed. But this control is essential for certain systems, e.g. if a system must meet deadlines of disk requests. Another drawback is the limited number of requests a disk can queue (we experienced 32 to 64 requests with current disks), however scheduling algorithms work better the larger the number of requests they can choose from.

The aim of our work is to

(a) show the feasibility of an SATF scheduler at driver level and

(b) use that scheduler to build a disk driver which can give Quality-of-Service guarantees

2 SATF scheduler

SATF scheduler require the ability to estimate the time a disk needs between two requests. We use the following function to calculate that time:

$$t = t_{seek}(\delta_{cylinder}) + t_{overhead} + t_{rotation}(\delta_{angle}, t_{overhead}, t_{seek}(\delta_{cylinder}))$$

t_{seek} is the time the disk head needs to move from its current position (i.e. the sector behind the last request) to the cylinder of the next request, it depends on the distance $\delta_{cylinder}$ to the target cylinder. $t_{overhead}$ denotes a fixed command overhead. $t_{rotation}$ is the time which must be waited until the target sector arrives at the disk head. It depends not only on the angle δ_{angle} between the current position and the next request but also on the time already elapsed due to the seek and command overhead.

We developed a set of automated benchmarks to gather the required information about a disk. The information include the geometry mapping, rotational speed, seek curve and command overhead.

Figure 1 shows the performance of our SATF scheduler compared to the internal disk scheduler and a SSTF (*Shortest Seek Time First*) scheduler implementation at driver level.

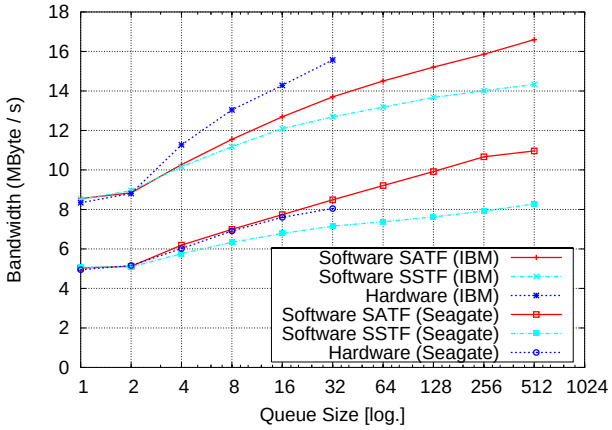

Figure 1: Disk scheduler performance, random workload, 64KB blocksize, disks IBM Ultrastar 36Z15 with 15,000 rpm, Seagate Barracuda 36ES2 with 7,200 rpm

The results show a 12% performance loss of our SATF implementation for the IBM disk at the maximum queue size the disk can handle, but also that the performance of the driver-level scheduler increases further for queue sizes the disk cannot handle. For the slower Seagate disk our scheduler outperforms the internal scheduler entirely.

3 QoS-aware disk scheduler

The measurements we just presented show that the scheduling of disk requests at driver level instead in the disk drive is still feasible even with current high performance disks. In our current work we want to use this result to build a disk scheduler which can give QoS guarantees.

In a QoS-aware system, the disk scheduler must ensure that deadlines of individual disk requests are met. This raises restrictions on the set of requests the scheduler can choose from. However, to still enable the scheduler to optimize the disk throughput, on each scheduling decision we determine the maximal subset out of the available requests so that the deadlines of all requests can be met even if a request is not executed in the current round.

We want to integrate this idea in the *Quality Assuring Scheduling* Framework presented in [1]. Using data streams where not all disk requests have to be executed will give the scheduler more options to maximize the utilization of the disk.

References

[1] C.-J. Hamann, J. Löser, L. Reuther, S. Schönberg, J. Wolter, and H. Härtig. Quality Assuring Scheduling - Deploying Stochastic Behavior to Improve Resource Utilization. In *22nd IEEE Real-Time Systems Symposium (RTSS)*, London, UK, December 2001.

[2] Lan Huang and Tzi-Cker Chiueh. Experiences in Building a Software-Based SATF Scheduler. Technical report, State University of New York at Stony Brook, July 2002.

[3] David M. Jacobson and John Wilkes. Disk scheduling algorithms based on rotational position. Technical report, HP Laboratories, 1991.