Lottery Scheduling: Flexible Proportional-Share Resource Management

Summary

This paper presents a randomized mechanism called lottery scheduling. This mechanism can be used effectively in implementing proportional-share resource management as well as modular resource management. Modular resource management of lottery scheduling is supported through ticket transfers, ticket compensation and the currency abstraction. A prototype for lottery scheduling is implemented on Mach 3.0 microkernel with its unoptimized performance comparable to Mach's standard time-sharing scheduler.

Key ideas

Lottery scheduling allocate resources through the use of lottery tickets abstraction. The holder of a winning ticket is granted use of resource in each allocation. The more ticket a client hold, the more likely that client is granted a resource. Lottery tickets are: *abstract*, *relative* and *uniform*:

1. Abstract because they represent resources rights without knowing details of hardware
2. Relative because value of a ticket changes dynamically as there are more competition
3. Uniform because rights to more complex resources can be represent simply as tickets

Winning tickets are chosen in a way that is probabilistic fair, clients with the majority of tickets is not guarantee the resource, while any client that holds at least one ticket will eventually be granted the resource. This property of lottery scheduling solves the problem of starvation in conventional priority-based scheduling.

Ticket transfers can happen when a client is blocking, solving the problem of priority inversion.

Sometime ticket *inflation* and *deflation* can happen between trusting client to adjust allocation. Clients that do not use all of their allocated resources is granted a compensation ticket. A currency abstraction can be use to increase flexibility and ensure resource rights.

Contribution

This paper presents a scheduling mechanism that is simple, efficient and responsive. The lottery scheduling mechanism solves conventional problems of priority-based scheduling like starvation and priority inversion by applying a randomized process. This mechanism can also be generalized to apply to many kind of resources.

Critique

Not much is mentioned about how tickets would be allocated and how they would transfer from process to process.

Because winning ticket is chosen in a randomized manner, processes cannot be guarantee to finish execution in a fixed amount time. This can be a problem in systems that require high precision.