1. Exokernel An Operating System Architecture for Application-Level Resource Management, SOSP 1995

This paper is about why application level resource management is good and how thy provides a protected low-level interface, and how they export information to the application, so that the application knows what is best to do.

Exokernel provide minimal abstraction for low-level hardware access. This can increase application performance because apps have more control of how resource are uses. And also the number of kernel crossings can be reduced not only that the cost of a kernel crossing itself can be reduced as the kernel internal structure is simple. But the downside is that hardware vendors need to build and maintain libraries that fit into Exokernel on a device. In addition, application developers should understand the use of resources they need, and should be able to use the corresponding Exokernel library, as well as modify the code.

It could be better to have an explanation of whether the higher costs of implementing an application can be justified by performance improvements.

2. Lottery Scheduling: Flexible Proportional-Share Resource Management, OSDI 1994

This paper proposes lottery scheduling which is an efficient way to implement proportional share resource management. Lottery scheduling uses a randomized allocation mechanism by using lottery tickets to represent rights to a resource. The resource is allocated to the winner of a lottery, which is dictated by a random number generator.

The advantage of this scheduling is that the overhead of the schedule is smaller, and flexible control is possible as ticket changes dynamically. But it is a probability-dependent scheduler, so it is never safe for the starvation problem.

It could be nice to have a description of how many tickets each process has. Is there an algorithm for that?