Motivation :

There is a general thought that traditional kernel dont scale well on multiple cores. It is thought that application’s performance is decreased as the number of cores is increased as they spend more time in the kernel. Various new OS designs and models have been proposed to solve this issue. This paper experiments and researches on scaling the traditional OS’s and analyse its results.

Key Idea :

The paper analyzes the behaviour of seven applications performance on Linux after scaling. The machine was running with 48 cores. The applications chosen use the kernel services extensively and are parallely, and are collectively referred as MOSBENCH. The main contribution of this paper is a set of 16 scalability improvements referred to as the patched kernel (PK). The paper also gives the idea of sloppy counters. From the results of behaviour of these applications, the authors suggested that traditional kernel designs can be made scalable without major design changes.

Strengths :

* The results shows that MOSBENCH applications are scaled well upto 48 cores which some code changes in the kernel and each applications.
* The paper introduces sloppy counters, that are used to speed up counting references to resources. Sloppy counters can be used to augment shared counters to make some uses more scalable without having to change all uses of the shared counter.
* Apart from sloppy counters, most changes required to make applications scalable are standard parallel programming techniques. Hence, the paper makes strong argument that traditional OS can be made scalable without major changes.
* Lock free protocol, similar to Linux’ lock-free page cache lookup protocol is used to allow cores to perform lookups for the same directory entries without serializing. This approach improves the scalability.

Weakness :

* The research cannot be generalized. Different applications or more number of cores are certain to reveal more bottlenecks, just as the authors encountered bottlenecks at 48 cores that were not important at 24 cores. For example the costs of thread and process creation seem likely to grow with more cores in the case where parent and child are on different cores.
* No approach is mentioned for resolving problems which are not bottlenecked by CPU cycles, like bottlenecks in applications or by some other hardware resource such as DRAM bandwidth.

Thoughts on paper :

The paper makes the statement that it is possible to make traditional kernel scalable, by experimenting on Linux. The paper shared its results of running applications on multi core Linux kernel. Although with slight changes and approach of sloppy counters, the kernel scale well, there is room for more enhancements. I also believe that hardware resources can be exploited more using appropriate techniques to enhance the performance of a scalable system.