# Arachne: Core-Aware Thread Management Paper Critique

## Summary

This paper introduces Arachne, a threads management system that is able to achieve both low latency and high throughput by making applications *core-aware*, which gives them information about hardware resources. Arachne uses a core arbiter to allocate CPU cores to applications, then changes the allocations based on requirements of applications. Every components of Arachne is implement at user-level, without any modifications to the kernel. The authors also implemented Archne in C++ on Linux and use the implementation for evaluation. Increased throughput and reduced latency were observed when incorporating Arachne with low-latency systems.

## Strengths

This paper identifies one weakness of the current thread implementation is that applications cannot optimize parallelism because they do not know details of hardware resources. Base on those findings, the authors designed a thread implementation that improve performance of applications, especially applications that process a large number of small requests.

One more strength of the paper is that its implementation is solely at user level, making it easy to implement in any system.

## Weaknesses and Improvement suggestions

Because threads are user-level, if a thread makes a blocking system call, the entire core arbiter process could be blocked. Although the authors mentioned this problem,  no solution is introduced.

Applications is dependent on a single core arbiter to perform allocations which makes it difficult to perform optimizations or make extensions to the system.