1. Efficient Virtual Memory for Big Memory Servers (ISCA 2013)

Virtual Memory and Paging cause systems that have a lot of memory and particular workloads to be slow because they spend a lot of time dealing with TLB misses. Allowing a contiguous section of virtual memory to map to a contiguous section of physical RAM allows us to avoid many TLB misses and the penalties from them, making those workloads faster.

This paper has strengths that we can add direct segment hardware that allows us to map some contiguous virtual memory to this contiguous block of physical memory. This mapping is dynamic and applications can choose how much of their virtual memory to this contiguous physical memory. In addition, Linux 2.6.32 was modified to reserved virtual and physical memory for the direct-mapping implementation. These are also the contributions of this paper.

However, this paper has weakness that new hardware support is needed. In other words, it is necessary that architecture-defendant implementation.

So, it would be better if commented on how not to use architecture-defendant implementation.

2. Coordinated and Efficient Huge Page Management with Ingens (OSDI 2016)

Modern computing is hungry for RAM, and hardware address translation overheads have grown with memory capacity. This motivates hardware manufacturers to provide TLBs with thousands of entries for large page sizes (called huge pages). However, current huge page has several problems. page fault latency and synchronous promotion, bloat, fragmentation and unfair performance and so on. So this paper present Ingens, a framework for huge page support that relies on a handful of basic primitives to provide transparent huge page support in a principled, coordinated way.

This paper has strengths that it can provide transparent huge page support that reduces latency, latency variability and bloat while providing meaningful fairness guarantees and reasonable tradeoffs between high performance and memory savings. It builds on a handful of basic primitives: utilization tracking, access frequency tracking, and contiguity monitoring.

But Ingens has weak point that this strategy risks losing performance due to high MMu overheads in low memory pressure situations. And Ingens promotes huge pages through a sequential scan from lower to higher VAs. This approach is may be unfair to processes whose hot regions lie in the higher VAs. Because different applications would usually contain hot regions in different parts of their VA spaces. (HawkEye ASPLOS'19)

So, it would have been better if the promote method, which meets the more accurate fairness between various processors, was considered.