Abstract

Virtual memory (VM) is a fundamental component of modern computer systems, widely adopted across various computing environments, from embedded devices to large-scale data centers. While it initially served to automate memory management by transparently swapping pages between main memory and secondary storage, it now plays a crucial role in ensuring system security, process isolation, and overall flexibility. However, the increasing overhead associated with traditional VM systems — originally designed for resource-constrained environments — has led to performance bottlenecks, particularly in systems with growing memory demands and applications with poor spatial locality. The ever-increasing depth of conventional hierarchical page tables further exacerbates these challenges.

To address these limitations, alternative approaches like *Inverted Page Tables* are used, but they have their own set of problems. The plethora of different designs and approaches to optimizing these designs suggest general performance problems. This thesis proposes a approach that aims to eliminate the need for page tables and main memory accesses altogether by using specialized mapping functions instead of costly page table walks. These mappings promise faster address translation, as they omit the cost associated with the page table accesses, while providing flexibility to system designers by defining them in software. This allows tailoring the virtual memory system to specific use cases.

The thesis details the theoretical foundations and practical implementation of a platform designed to facilitate the exploration and experimentation with such mapping functions. It also presents an initial simplified VM system prototype that demonstrates the feasibility of the approach.

1 Introduction

Virtual memory provides a multitude of features that vastly simplify the life of application programmers [JM98b]. Computer systems of all scales, ranging from small embedded devices to huge data centers use virtual memory [BL17]. While originally being used to automate the task of swapping pages of processes between main memory and secondary storage transparently to the processes [JM98b], it now is the foundation for security, reliability, process isolation and flexibility [JM98a; Wal99].

With ever increasing memory sizes of systems and memory requirements of applications, the overhead of virtual memory systems, being developed for systems that only had scarce resources available [Hal+23], is degrading performance and increase power consumption significantly [ZSM20]. Orthodox hierarchical page table organizations [TB14] can get as deep as 5 levels in commodity hardware [Int17]. Not only does this add a level of indirection for each page table walk, this penalty is even higher in virtualized systems, potentially accounting for up to 5% - 90% of the runtime [YT16].

Alternative designs like inverted page tables realize page table mappings using hash functions [TB14]. While reducing the number of additional memory references, these approaches have problems on their own: Some features of virtual memory systems become harder to implement, exacting additional performance penalties [YT16] and page table lookups can get a lot more expense when following collision chains [JM98c].

Chapter 3 will show, that there are a lot of different approaches to optimize the virtual memory system. This shows that there is little agreement on how the virtual memory system is best implemented [JM98c]. However, most of the optimization approaches still relied on page table structures to do the bookkeeping of the mappings.

This thesis explores the idea of getting rid of page table structures all together and base the mapping of virtual to physical addresses on specially crafted functions (for example non-cryptographic hash functions [Mit+21]). Hash functions implemented by simple arithmetic instructions are orders of magnitude faster than the multiple memory accesses required by a page table walk [TB14] and allowing them to be defined in software gives the operating system a lot of flexibility to fit the implementation of the mapping function to the custom needs of the use case.

This thesis will describe the theory and implementation of a platform that facilitates the definition and the experimentation with such mapping

As memory sizes and application demands grow, the overhead of virtual memory systems is hurting performance. These systems, designed for resource-limited machines, are now causing significant power consumption issues.