Automatic Memory De-duplication Support in Hypervisor

Furquan Shaikh, Fangzhou Yao, Roy Campbell

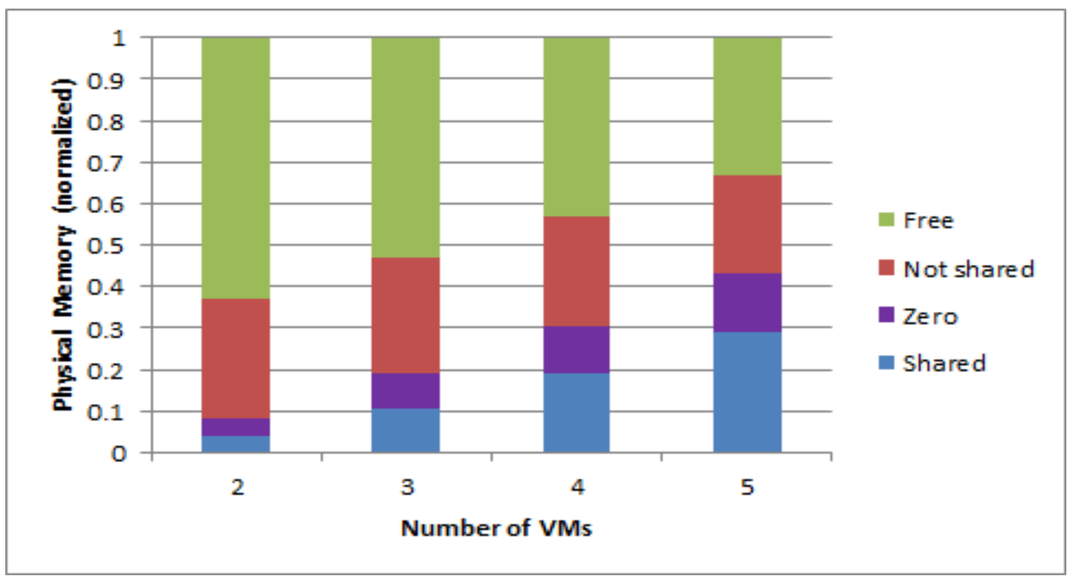
Abstract

Virtualization techniques are widely used nowadays in datacenters and cloud computing environments. Such environments are installed with large number of similar virtual machines sharing the same physical infrastructure. Thus, exploiting opportunities for optimization to reduce the cost becomes inevitable. In this paper, we focus on the memory usage optimization across virtual machines by de-duplicating the memory on per-page basis. We maintain a single copy of the duplicated pages in physical memory using copy-on-write semantics. Unlike some existing strategies, which are intended only for applications and need user's advice, our approach provides an automatic memory de-duplication support within the hypervisor to achieve benefits across operating system code, data as well as application binaries. We have implemented a prototype of this system within Xen hypervisor to support both para-virtualized and full-virtualized instances of operating systems.

Objectives

- Built-in support within the hypervisor to detect duplicate memory pages across its guest systems
- Copy-on-write mechanism to de-duplicate pages and maintain a single copy of memory page
- Automatic detection and update of memory footprint at runtime without any hints from the guest operating system

Motivation



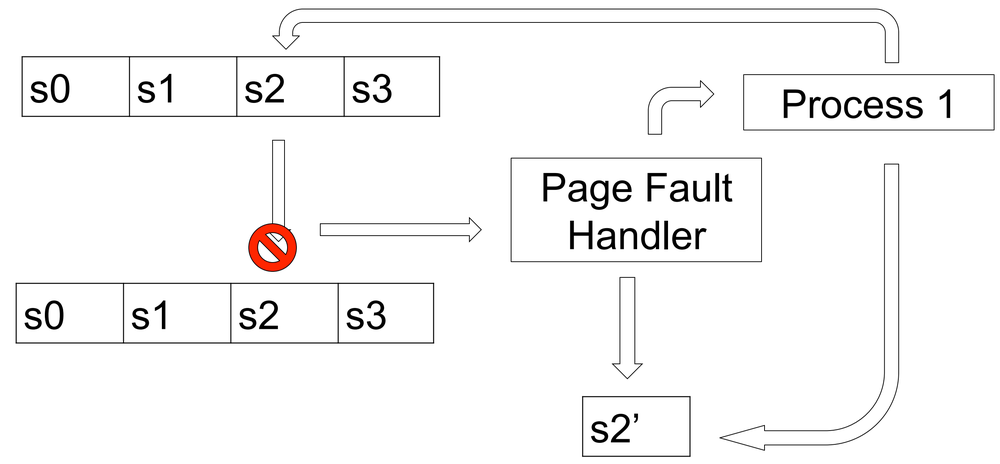
Distribution of physical memory in virtualized environments considering zero pages as a separate component

Our aim is to de-duplicate those long-term shared pages, **read-only** pages including including kernel code, kernel static data, application binaries and application libraries are contained in pages that are marked as read-only, and **read-write** pages that stay constant for a certain period.

Identical pages are found by *SHA1 hashing* **periodically** when the CPU is *idle*.

Design

- Copy-on-write (COW) technique

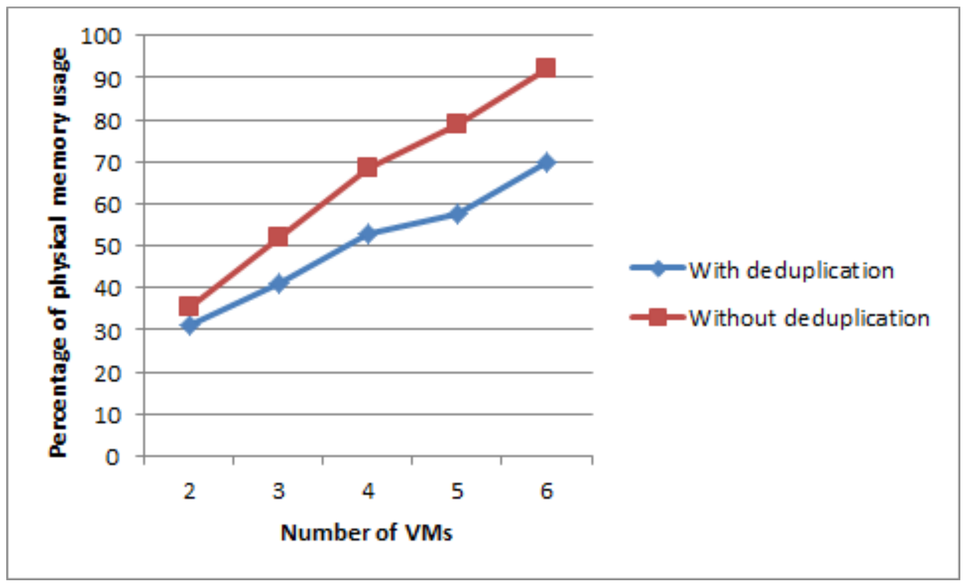


- Red-black (RB) Tree based Implementation

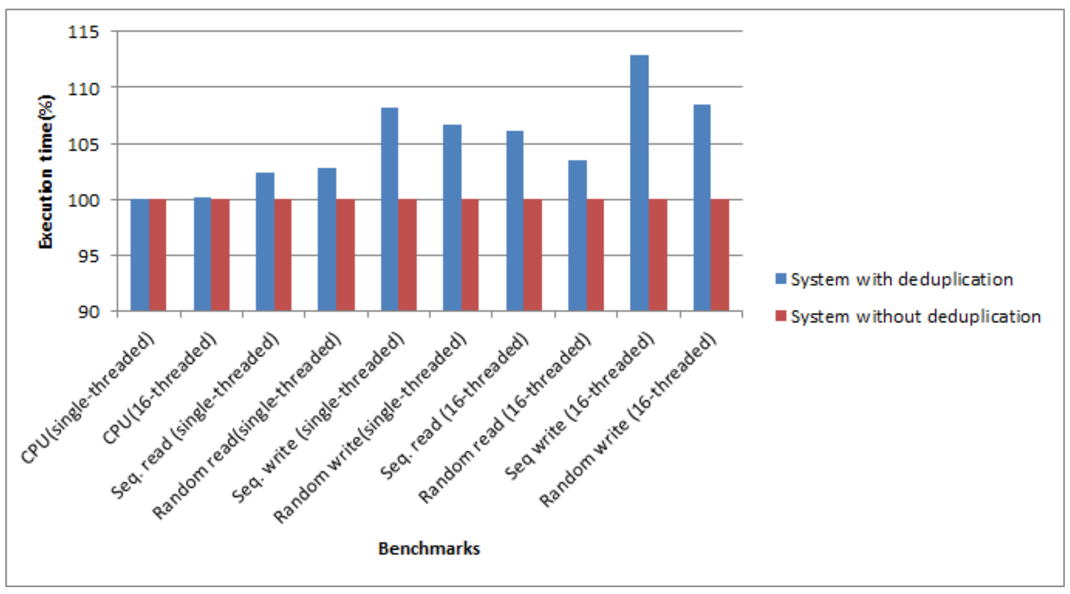
We use RB tree to store the hash data. This is a compromise to time and space trade-off, they are both O(log n) in this data structure.

Also, this is a standard data structure presented in Linux Kernel and Xen hypervisor kernel.

Results



Memory usage after de-duplication as the number of virtual machines increases, there is typically a 20% memory save.



System performance benchmarked by SysBench