Computer Systems Organisation (CS2.201)

Summer 2021, IIIT Hyderabad

26 July, Monday (Lecture 27) – Virtual Memory

Taught by Ziaul Choudhury

Virtual Memory

Introduction

When multiple processes are executing, the CPU's context continually switches among these processes, giving the illusion to each process that it is the only one with access to the memory and the processor.

Virtual memory is what gives this illusion to the processes. It is an elegant interaction of hardware exceptions, hardware address translation, main memory, disk files and kernel software that provides each process with a large, uniform and private address space.

There is a region in the hard disk called the swap space that contains the virtual addresses of each process.

Physical and Virtual Addressing

Each byte in the main memory has a unique physical address or PA. Under virtual addressing, the CPU accesses memory by generating virtual address or VA, which is converted to a PA before being sent to the memory (using address translation).

Virtual addresses enable the programs to remain completely independent, because using the physical addresses would enforce a range of available addresses on each program's memory (which would destroy the illusion of being in complete control of the memory).

Address Spaces

An address space is an ordered set of nonnegative integer addresses. A virtual address space with $N = 2^n$ addresses is called an n-bit address space.

However, the sum total of virtual addresses over all processes typically exceeds the main memory's capacity. This becomes possible by treating the main memory as a cache for the virtual addresses space – at a given point, not all the processes' virtual memories are needed in the main memory. Only the running processes' data is required. This is called on-demand caching.

Pages and Paging

The main memory and the disk are the upper and lower levels, respectively, of the caching process. The blocks (transfer units between memory and disk) in this relation are called pages. Physical pages that map to virtual pages of the same size are called page frames.

The advantage of a page-level mapping is that once the address of the first byte of the page is translated, the individual addresses can be translated simply by using the offset from the address of the first byte of the page. The mapping between the virtual and physical pages is contained in a data structure called a page table, which is simply an array of page table entries (PTEs) which either point to an object in the hard disk (a virtual page that is not yet mapped) or an object in the physical memory (which is a mapped virtual page).

At any point, the set of virtual pages is divided into un-allocated (not yet created for the process), cached (stored in memory) and un-cached pages (only stored in hard disk).

Just like a normal cache, hits and misses occur when trying to access pages from main memory. Note, however, that the main memory is a full associative cache.