Main Memory

Memory-Management Unit (MMU)

MMU is a hardware device that maps logical addresses to physical addresses at tun time.

Base register now called relocation register. The value in relocation register is added to every address generated by a user process at the time it is sent to memory.

User program deals with logical addresses; it never sees the real physical addresses.

Contiguous Memory Allocation

Main memory must support both OS and user processes.

Main memory usually divided into two partitions: one for OS and one for user processes. OS usually held in high memory. Each user process contained in a single contiguous section of memory.

Relocation register and limit register used to protect use processes from each other, and from changing OS code and data.

Relocation register contains value of smallest physical address.

Limit register contains range of logical addresses.

Memory Allocation

Initially, all memory is available and is considered one large hole. A hole is a block of avai mem

When a process arrives, it is allocated memory from a hole large enough to accommodate it. If hole is too large, the unused part is returned to the set of holes

When a process terminates, it releases its block of memory, these holes are merged to form a large hole.

In general, holes of various sizes are scattered throughout memory

Operating system maintains information about a) allocated memory and b) free memory (holes)

Dynamic Storage-Allocation Problem

How to satisfy a request of size n from a list of holes?

First-fit: Allocate the first hole that is big enough

Best-fit: Allocate the smallest hole that is big enough; mist search entire list, unless ordered by size. Produces the smallest leftover hole.

Worst-fit: Allocate the largest hole; must also search entire list, unless ordered by size. Produces the largest leftover hole.

First-fit and best-fit better than worst-fit in terms of storage utilization.

First-fir generally faster than Best-fit.

Fragmentation

External Fragmentation: There is enough total memory space to satisfy a request, but the available spaces are not contiguous.

Statistical analysis of first-fit reveals that given N allocated blocks, another 0.5 N blocks will be lost to external fragmentation.

That is, 1/3 of memory may be unusable -> 50 percent rule.

Internal fragmentation: Allocated memory may be slightly larger than requested memory; the size difference is memory internal to partition, but not being used.

E.g., 4000 bytes requested, 4096 bytes (4KB) allocated.

Overcoming External Fragmentation

One solution to external fragmentation is compaction.

Shuffle memory contents to place all free memory together in one large block.

Compaction is possible only if relocation is dynamic, i.e., address binding done at execution time using a relocation register.

Another solution is to permit the physical address space of a process to be noncontiguous.

Paging

Physical address space of a process can be noncontiguous; process is allocated physical memory whenever the latter is available.

Divide physical memory into fixed-sized blocks called frames. Frame size is power of 2, typically between 4KB ( bytes) and 1GB ( bytes).

Divide logical memory into blocks of same size called pages.

To run a process of size N pages, need to find N free frames and load pages of process to frames.

OS keeps track of all free frames.

External fragmentation is avoided; still have internal fragmentation.

Paging is used in most operating systems.

Kernel maintains a page table for each process.

Page table has an entry for each page of the process, entry indicates the memory frame allocated to the page. Page table is used to translate logical addresses to physical address

Address Translation Scheme

Address generated by CPU (i.e., logical address) is divided into:

Page number (p) – used as an index into a page table which contains base address of each frame in physical memory.

Page offset (d) – location in the frame

Base address of frame is combined with the page offset to define the physical memory address that is sent to the memory unit.

If size of logical address space = bytes and page size = bytes, then high-order m-n bits of logical address represent the page number. Low-order n bits represent the page offset.

Page number = address reference / page size offset = address reference Mod page size

Logical address space of 2048 pages with a 4KB page size, mapped onto a physical memory 512 frames.

Logical memory = 2048\*2^12(4KB) = 2^23. 23 bits in logical address. 23 bits required in logical address. physical memory = 512 \* 2^12 = 2^21. 21 bits required in physical address.

Virtual Memory

Less I/O need to load or swap portions of programs into memory --> each program runs faster.

Virtual memory involves the separation of logical memory as perceived by developers from physical memory.

Only part of program needs to be in memory for execution.

Logical address space can therefore be much larger than physical address space.

Virtual Address Space – logical view of how process is stored in memory

Usually start at address 0, contiguous addresses until end of address space.

Meanwhile, physical memory organized in page frames.

MMU must map logic addresses to physical addresses.

The hold between stack and heap is part of virtual address space.

No physical memory needed until heap or stack grows to a new page.

Processes can share system libraries by mapping shared library into virtual address space.

The pages where the libraries reside in physical memory are shared by processes.

Processes can share memory by mapping shared physical pages into virtual address space.

Demand paging – bring a page into memory only when it is needed.

When a page is referenced, invalid reference 🡪 abort, not-in-memory 🡪 bring to memory.

Valid-Invalid Bit

With each page table entry a valid-invalid bit is associated

v 🡪 page is legal and in-memory

i 🡪 page is illegal or legal but not-in-memory

initially valid-invalid bit is set i on all entries.

Access to a page marked invalid causes a page fault

Steps in Handling Page Fault

1. During MMU address translation, referencing a page marked invalid will cause a page fault and the hardware will trap to operating system.
2. OS checks an internal table (kept with PCB) for the process to decide: Invalid reference 🡪 abort. Just not in memory 🡪 bring the page in memory.
3. Find a free frame
4. Swap page into frame via scheduled disk operation.
5. Modify the page table to indicate page is now in memory: set validation bit = v
6. Restart the instruction that caused the page fault.

Aspect of Demand Paging

Pure Demand Paging – start process with no pages in memory

OS sets instruction pointer to first instruction of process, page is non-memory-resident 🡪 Page Fault. And page fault occurs for every page on first access.

A given instruction could access multiple pages 🡪 multiple page faults

Hardware support needed for demand paging.

Free-Frame List

When a page fault occurs, the operating system must bring the desired page from secondary storage into main memory.

Operating system typically allocate free frames using a technique known as zero-fill-on-demand – the content of the frames are “zeroed-out” before being allocated.

When a system starts up, all available memory is placed on the free frame list.

Performance of Demand Paging

Three major activities:

Service the interrupt-careful coding means just several hundred instruction needed.

Read the page – lots of time

Restart the process – again just a small amount of time

Page Fault Rate p 0 <= p <= 1. If p = 0 no page faults, if p = 1, every reference is a fault.

Effective Access Time (EAT) = (1-p) x memory access time + p x page-fault service time.

EX. Memory access time = 200 nanoseconds, average page-fault service time = 8 milliseconds.

EAT = (1-p) x 200 + p x 8000000

If one access out of 1000 causes a page fault (p=0.001), then EAT = 8200 ns = 8.2 ms

Copy-on-Write

COW allows both parent and child processes to initially share the same pages in memory.

If either process modifies a shared page, only then is the page copied.

COW allows more efficient process creation as only modified pages are copied.

Page Replacement

How