Operating Systems W11L1 - Memory Management II (ctd)

Class	Operating Systems
Created	@Nov 10, 2020 5:08 PM
Materials	
Reviewed	☐
Type	Lecture

Design Issues

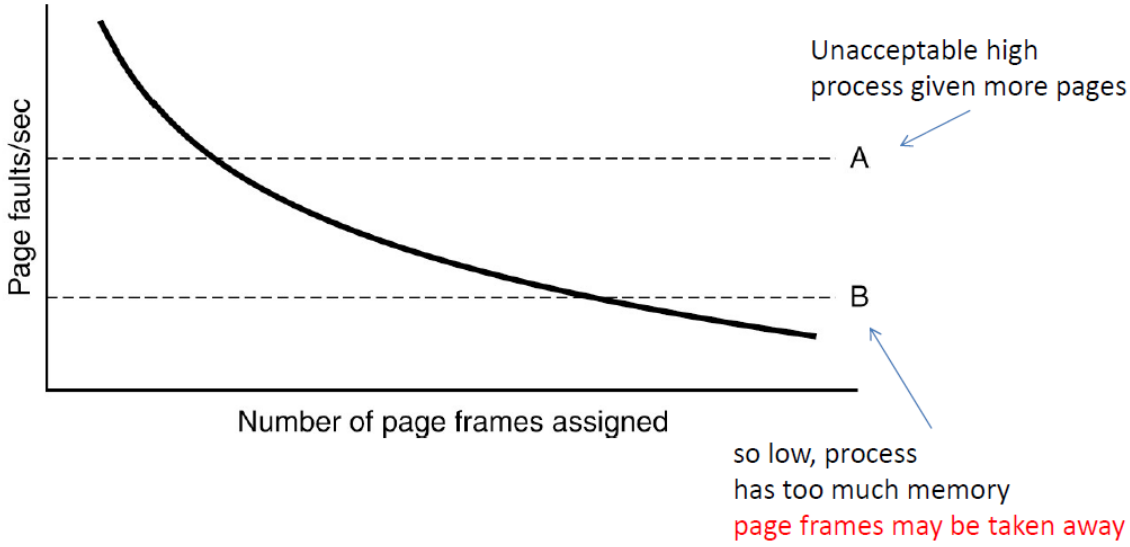
- **Local Algorithms:** Allocating every process a fixed fraction of the memory
- **Global Algorithms:** Dynamically allocate page frames
- All in all, global is more sophisticated but works better

Global Allocation

- Understand minute details of general case before tackling any edge cases
- Method 1 → Periodically determine the number of running processes and allocate each process an equal share
- Method 2 (better) → Pages allocated in proportion to each process total size (in terms of number of pages)
    - How do we consider number of pages to take and when to take them?

▼ Page Fault Frequency (PFF) Algorithm

We always want to be between A and B, calculated by OS based on available physical memory



Load Control

- If PFF shows that some processes need more memory but none need less, we *swap* some processes to disk in order to free up all the pages they are holding
  - We want to keep the CPU busy
- Knowing to code is easy, but design choices and problem solving is where the skill in CS shines through

Page Size

- If you pick a large page size, you will get internal fragmentation
- Small pages lead to large page tables and determinetal effect on performance due to more transfers to disk

▼ Math Involved

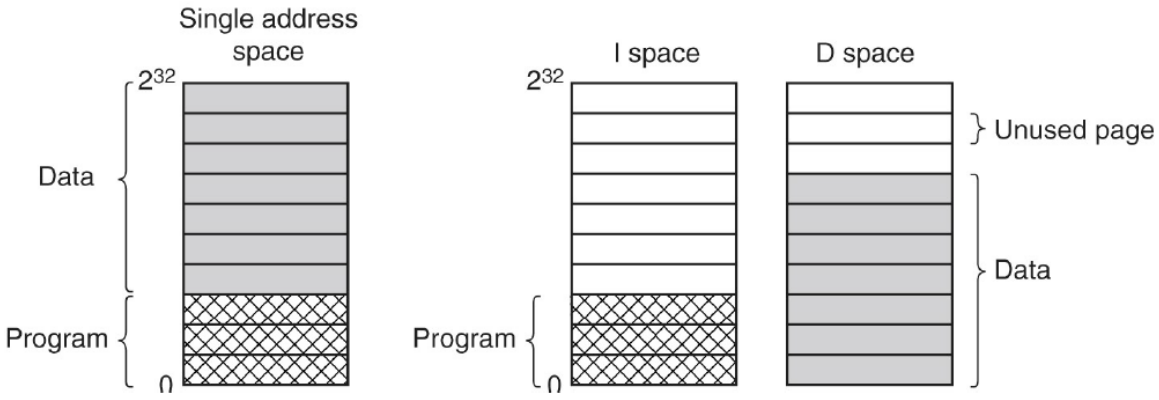
- Assume:
  - s = process size
  - p = page size
  - e = size of each page table entry
- So:
  - number of pages needed = s/p
  - occupying: se/p bytes of page table space
  - wasted memory due to fragmentation: p/2
  - overhead = se/p + p/2
- We want to minimize the overhead:
  - Take derivative of overhead and equate to 0:
  - $-se/p^2 + \frac{1}{2} = 0 \rightarrow p = \sqrt{2se}$

- Getting an instruction is still accessing memory

Separate Address Spaces

- Two spaces per process (instruction and data)

▼ Diagram



•The linker must know about it
•Paging can be used in each separately

- Nt something we'll go into very in-depth, but worth looking at and learning from
- Everything has a cost and benefit — If something is great, you must consider what the cost to get that is

Shared Pages

- Several processes using the same pages
  - This sharing is done via *libraries*, such as `math.h` or something

Shared Libraries

- **Position Independent Code:** Compilers must not produce instructions using absolute addresses in libraries
- Loaded with program or at function calls
- Know the advantages and disadvantages of static and dynamic linking (CSO material)

Cleaning Policy

- Paging daemon that sleeps most of the time, awakened to inspect state of memory periodically

- If there aren't enough free page slots, daemon begins to select pages toe evict

Conclusions

"Very widely used" means always used, but there could be some random case where it's not (the prof couldn't think of one, but he's preparing if there is one haha).

- Virtual memory is very widely used
- Many design issues for paging systems:
  - Page replacement algorithm
  - Page size
  - Local vs Global Allocation
    - Global algorithms work better
  - Load control
  - Dealing with shared pages