Today

• How does the OS manage the memory using the Virtual Memory scheme?
• Virtual Memory
• Page Fault
• Page Replacement
Virtual Memory

- Each process has the **illusion** of a large address space.
  - Begins at 0 and ends at $2^{32} - 1$.
- However, physical memory might be much smaller than the sum of the memory request of the ready processes.
- How do we give this illusion to multiple processes?
  - **Virtual Memory**: some addresses may reside in disk
Virtual Memory

Virtual memory

Physical memory

page 0
page 1
page 2
page 3
page 4
page N

page table

disk
Virtual memory

- **Separates** user's logical memory from physical memory.
- Only **part of the program** needs to be in memory for execution.
- Logical address space can therefore be **much larger** than physical address space.
- Allows address spaces to be **shared** by several processes.
- Allows for more **efficient** process creation.
How does VM work?

- Modify Page Tables with another bit (“is present”):
  - If page in memory, $is_{\text{present}} = 1$, else $is_{\text{present}} = 0$.
  - If page is in memory, translation works as before.
  - If page is not in memory, translation causes a page fault.
Page Faults

• On a page fault:
  • OS finds a **free frame**, or evicts one from memory (which one?)
    - Want knowledge of the future?
  • Issues disk request to **fetch data** for page (what to fetch?)
    - Just the requested page, or more?
  • **Block** current process, **context switch** to new process (how?)
    - Process might be executing an instruction
  • When disk completes, set **present bit to 1**, and current process in ready queue
Page Faults

1. Trap
2. Reference
3. Page is on backing store
4. Bring in missing page
5. Reset page table
6. Restart instruction
What to replace?

- What happens if there is no free frame?
  - Find a suitable page in memory, swap it out.

- Page Replacement
  - When process has used up all frames it is allowed to use,
  - OS must select a page to eject from memory to allow new page.
  - The page to eject is selected using the Page Replacement Algorithm.

- Goal: Select page that minimizes future page faults.
Page Replacement

1. Swap out victim page
2. Change to invalid
3. Swap desired page in
4. Reset page table for new page
Modified/Dirty Bits

- Use **modify (dirty) bit** to reduce **overhead of page transfers**.
- Only modified pages are written to disk.
- Non-modified pages can always be brought back from the original source.
  - Process text segments are rarely modified, can bring pages back from the program image stored on disk
Page Replacement Algorithms

- **Random**: Pick any page to eject at random
  - Used mainly for comparison
- **FIFO**: The page brought in earliest is evicted
  - Ignores usage
- **OPT**: Belady’s algorithm
  - Select page that won’t be used for longest time
- **LRU**: Evict page that hasn’t been used the longest
  - Past could be a good predictor of the future
- **MRU**: Evict the most recently used page
- **LFU**: Evict least frequently used page
FIFO

- First-In-First-Out
- Simple
- A FIFO queue holds all pages in memory.
- The OS replaces the page at the head of the queue.
- A newly brought page is placed at the tail of the queue.
FIFO

- Reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
- 3 frames (3 pages can be in memory at a time **per process**): 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

```
1  4  5
2  1  3   9 page faults
3  2  4
```

- 4 frames: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

```
1  5  4
2  1  5   10 page faults
3  2  2
4  3  3
```

- Belady’s Anomaly: more frames → more page faults
Belady’s Anomaly

![Graph showing the number of page faults against the number of frames.

- X-axis: Number of frames (1 to 7)
- Y-axis: Number of page faults (2 to 16)
- The line graph shows a decrease in page faults as the number of frames increases, illustrating Belady's Anomaly.]
Optimal Algorithm

- Replace page that will not be used for longest period of time.
- 4 frames example
- 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

```
1  4
2
3
4  5
```

- Used for measuring how well your algorithm performs
OPT Approximation

• In real life, we do not have access to the future page request stream of a program.
• So we need to make a best guess at which pages will not be used for the longest time.
Today

- How does the OS manage the memory using the Virtual Memory scheme?
- Virtual Memory
- Page Fault
- Page Replacement