15.Virtual Memory

* Allow the size of the user's program → memory size.

* Good for time-sharing systems.

* Swapping: moving processes from main memory to disk and back.

* Demand Paging: a paging system with swapping.

* If a page is unmodified ==> simply replace it.

* A present/absent bit is recorded in the page table.

* Page fault: when a page needed is not in memory (checking the present/absent bit or called valid bit)
  ==> trap to OS; swapping occurs.

* Kick which page out ? ==> page replacement algorithm.
  (try to choose the one with the smallest chance to be referred again)

* Referenced bits; modified bits.

* **Page Replacement Algorithm**

1) FIFO

*Belady's anomaly with the FIFO strategy

2) LRU (Least Recently Used)

- Can be implemented by a stack or a counter or a bit matrix.

- Throw out the page that has been unsed for the longest time.

- By a bit matrix, when reference page K, all bits in row K = 1
  all bits in column K = 0.

3) The Clock Page

- Circular running.

- Test R bit: if 0 kick it out else reset R bit = 0 and continue.

*** Global/Local replacement**

- global: dynamically allocate page frames from runnable processes; the number of page frames assigned to each process caries in time (may increase).

- local: assign each process a fixed amount of memory.

* **Threshing**

- No work is getting done because the processes are spending all their time paging.
  This very high paging activity is called threshing.

- Because when page fault occurs, CPU spends time in swap in/out.

- A process is threshing if it is spending more time in paging than executing.

- Circular effect:
  we want to increase the CPU utilization -->
  increase the degree of multiprogramming -->
  may result in high rate of page fault -->
  CPU utilization is decreased and dropping down.

- Due to global replacement policy.

* To prevent threshing, we must provide a process as many memory frames as it needs.

* But, how do we know how many frames it "needs" ?

* The working set strategy: it starts by looking at how many frames a process is actually using.

* This strategy defines the locality model of process execution.

* A locality is a set of pages that are actually used together. (loop, array)

* Define the working set window to examine the most recent pages referenced.

* The set of pages that a process is currently used is called its working set. (within n consective clock ticks)

* **Page Size**

- the size of a page table.
  when page size is smaller, the page table size is larger.

- memory utilization.
  when page size is smaller, memory utilization is larger.
  (* internal fragmentation is smaller *)

- I/O time to read/write a page.
  seek time + latency + transfer time.
  seek time and latency is propotational to the number of needed paging.
  transfer time is propotional to the size of a page.

- page fault rate.
  when page size is smaller, the page fault rate is larger.