ECE7995
Caching and Prefetching Techniques in Computer Systems

In-class questions for Presentation:

*Token-Ordered LRU: An Efficient Page Replacement Policy and Implementation in Linux Systems*
The definition of working set is as follows: at virtual time \( t \), a program’s working set \( W_t(\theta) \) is the subset of all pages of the program which has been referenced in the previous \( \theta \) virtual time units (a.k.a. working set window). Answer the following questions

1. Can we use wall-clock time to replace the virtual time in the definition? Why?

2. Then what is the virtual time?

Do you think that the multiprogramming level (number of concurrently running programs) of a computer would affect a program’s working set size? If yes, how?

Do you think that processor speed would affect a program’s working set size? If yes, how?

Do you think that the number of I/O requests or the amount of I/O data of a program would affect a program’s working set size? If yes, how?

How would temporal locality affect a program’s working set size?
• In the global LRU replacement policy, pages that belong to any processes are placed into a common LRU stack and managed un-discriminatorily. In contrast, in the local LRU replacement policy, pages that belong to different processes are placed into different stacks. Answer the following questions:

1) In the local LRU replacement policy, if the working set size of a process is smaller than its LRU stack size, do you think LRU can always hold its working set? Why?

2) In the global LRU replacement policy, the pages of a processes can slip out of the stack in these two scenarios: (1) the program is waiting for user inputs; (2) the program is reading data from hard disk. Explain why these pages become LRU pages. Which are true/false LRU pages?

• In Windows, sometimes pages of background programs (associated with inactive windows) can be replaced, do you think if these are pages should be replaced according to the distinction of true/false LRU pages.

• Interactive programs such as MS Word and Powerpoint are prone to lose their resident pages, which may hurt their responsiveness due to consequent access to on-disk data. Do you know what can be done to improve their responsiveness.

• Experiences show that once a process establishes its working set, other thrashing process can hardly steal pages from it. Can you explain it.

• Once a process grabs the token, is there a concern that the process may abuse the token privilege, such as holding memory more than it actually needs? Why.