CISC 3320

C20d Paging: OS Examples

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Acknowledgement

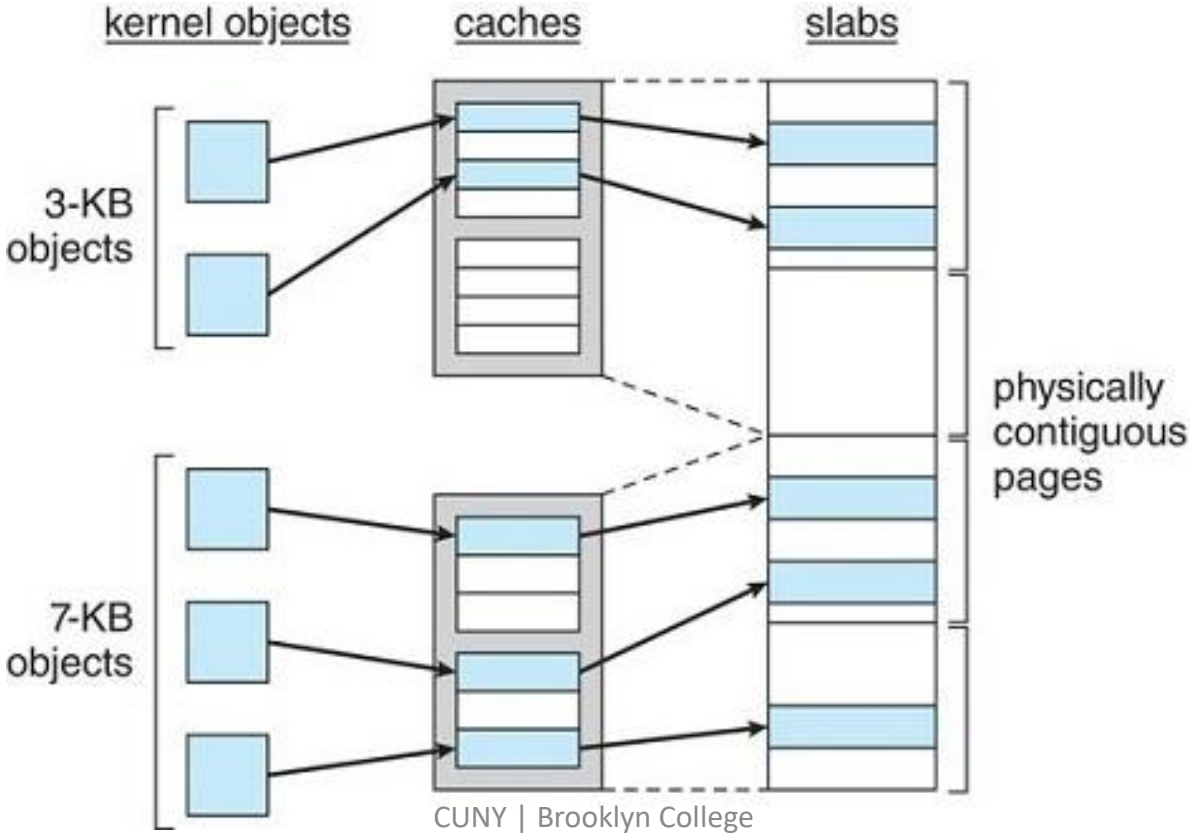
- These slides are a revision of the slides provided by the authors of the textbook via the publisher of the textbook

Outline

- Operating-System Examples
  - Linux
  - Windows
  - Solaris

Linux

- Linux uses a variation of slab-allocation algorithm



Slab Allocation

- Use cache to store kernel objects
  - A single cache for each unique kernel data structure
    - e.g., a separate cache for the data structure representing process descriptors
  - A slab is made up of one or more physically contiguous pages
  - A cache consists of one or more slabs
  - When a cache is created, a number of objects, initially marked as free, are allocated to the cache.

Linux Slab

- In Linux, a slab may be in one of three possible states:
  - Full. All objects in the slab are marked as used.
  - Empty. All objects in the slab are marked as free.
  - Partial. The slab consists of both used and free objects.

Benefits of Slab Allocation

- No memory is wasted due to fragmentation.
- Memory requests can be satisfied quickly.

SLAB

- The slab allocator first appeared in the Solaris 2.4 kernel.
- Linux originally used the buddy system
- Linux kernel adopted the slab allocator from Version 2.2 to Version 2.6.24
  - Linux refers to its slab implementation as SLAB.
- Recent distributions of Linux include two other kernel memory allocators
  - The SLOB and SLUB allocators.

SLOB

- The SLOB allocator is designed for systems with a limited amount of memory
  - e.g., embedded systems.
- SLOB stands for "simple list of blocks
- SLOB maintains three lists of objects
  - small (for objects less than 256 bytes),
  - medium (for objects less than 1,024 bytes), and
  - large (for all other objects less than the size of a page).
  - Memory requests are allocated from an object on the appropriate list using a first-fit policy.

SLUB

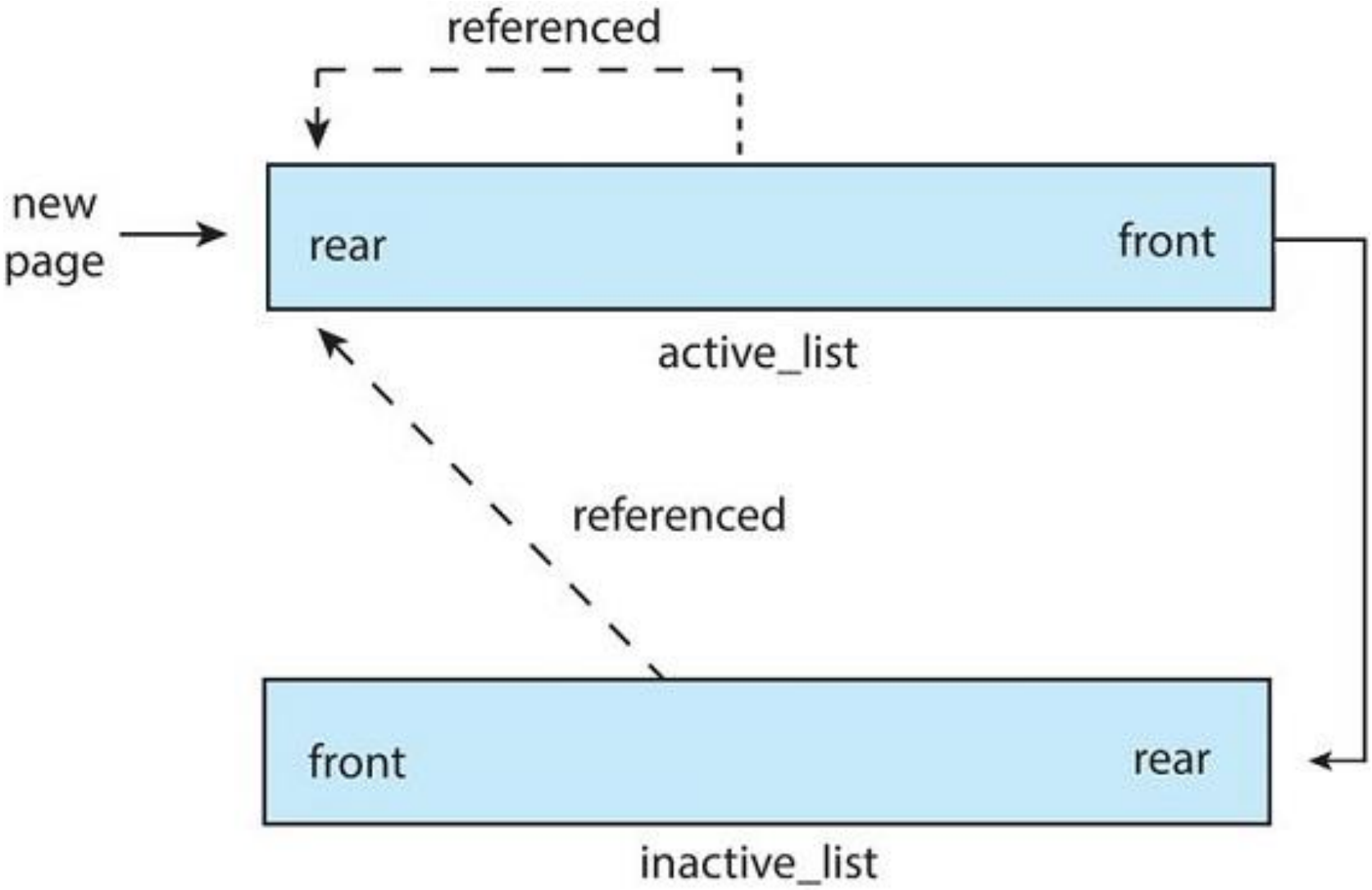
- Linux kernel replaces SLAB by the SLUB allocator since Version 2.6.24
- SLUB reduced much of the overhead required by the SLAB allocator.
  - SLAB stores certain metadata with each slab while SLUB stores these data in the page structure the Linux kernel uses for each page.
  - Additionally, SLUB does not include the per-CPU queues that the SLAB allocator maintains for objects in each cache.
    - For systems with a large number of processors, the amount of memory allocated to these queues is significant.
    - Thus, SLUB provides better performance as the number of processors on a system increases.

Linux Virtual Memory

- Demand paging, allocating pages from a list of free frames
- Global page-replacement policy similar to the LRU-approximation clock algorithm

Active List and Inactive List

- Linux maintains two types of page lists
  - active_list. Pages are considered in use
  - inactive_list. Pages have not recently been referenced and are eligible to be reclaimed.
  - Accessed bit. Set whenever the page is being referenced.
    - When a page is being allocated or referenced, the accessed bit is set and the page is moved to the rear of the active_list.
  - Balance between the two lists. when the active_list grows much larger than the inactive_list, pages at the front of the active_list move to the inactive_list, where they become eligible for reclamation.


referenced
new
page
rear
front
active_list
referenced
front
rear
inactive_list

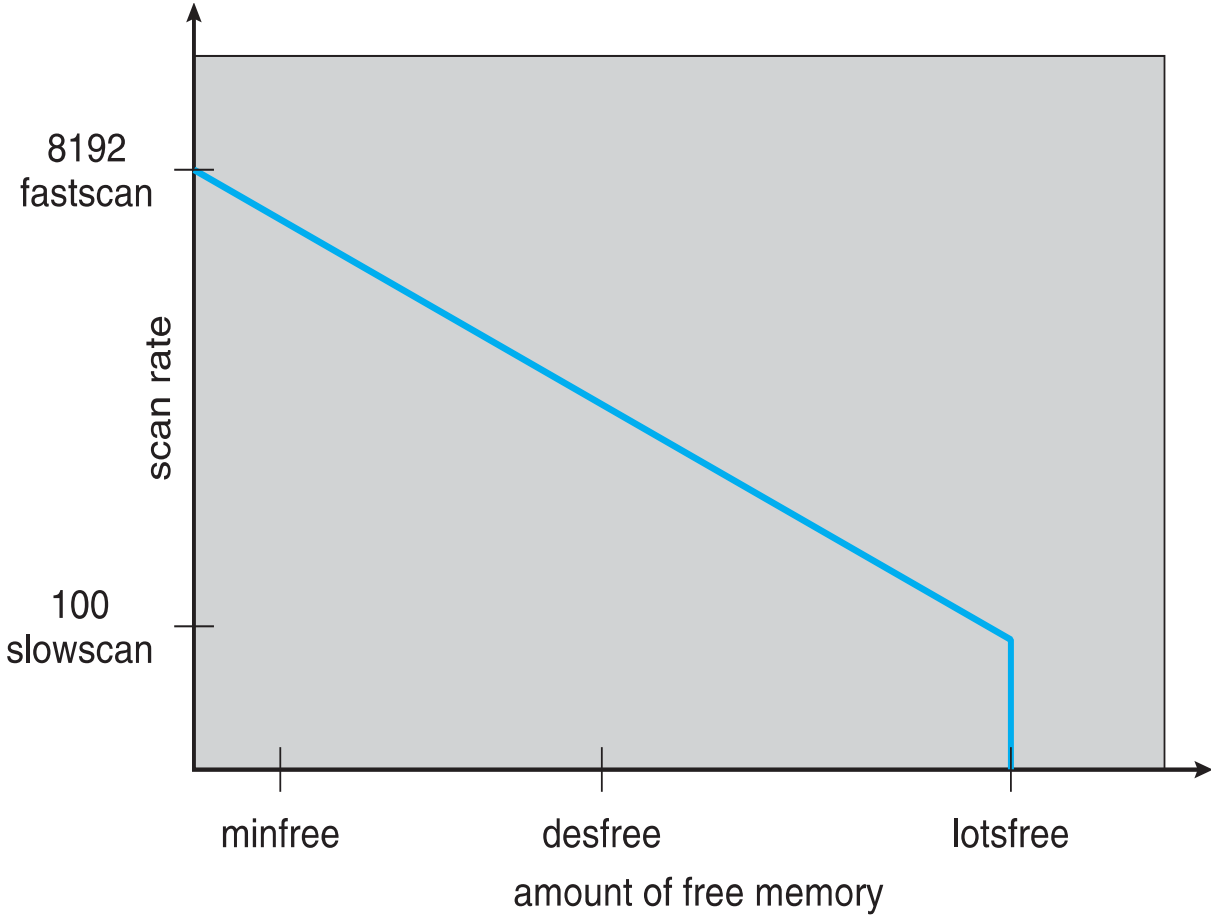
Windows

- Uses demand paging with **clustering**. Clustering brings in pages surrounding the faulting page
- Processes are assigned **working set minimum** and **working set maximum**
- Working set minimum is the minimum number of pages the process is guaranteed to have in memory
- A process may be assigned as many pages up to its working set maximum
- When the amount of free memory in the system falls below a threshold, **automatic working set trimming** is performed to restore the amount of free memory
- Working set trimming removes pages from processes that have pages in excess of their working set minimum

Solaris

- Maintains a list of free pages to assign faulting processes
- **`Lotsfree`** – threshold parameter (amount of free memory) to begin paging
- **`Desfree`** – threshold parameter to increasing paging
- **`Minfree`** – threshold parameter to being swapping
- Paging is performed by **`pageout`** process
- **`Pageout`** scans pages using modified clock algorithm
- **`Scanrate`** is the rate at which pages are scanned. This ranges from **`slowscan`** to **`fastscan`**
- **`Pageout`** is called more frequently depending upon the amount of free memory available
- **Priority paging** gives priority to process code pages

Solaris 2 Page Scanner



Questions?