



CS 3204 Operating Systems

Lecture 25
Godmar Back



Announcements

- Project 2 due **Fri March 24**
 - 2 day extension to stay in synch with other CS 3204 sections
- Midterm **Fri March 24**
 - We start at 10:00am!
- Project 3 page table design document due **Wed March 29**
 - See project page



CS 3204 Spring 2006 3/22/2006 2


Accessing User Pointers & Paging

- Kernel must check that user pointers are valid
 - P2: easy, just check range & page table
- Harder when swapping:
 - validity of a pointer may change between check & access (if another process sneaks in and selects frame mapped to an already checked page for eviction)
- Possible solution:
 - verify & lock, then access, then unlock

```


if (verify_user(addr))
    process_terminate();
// what if addr's frame is just now
// swapped out by another process?
*addr = value;

```



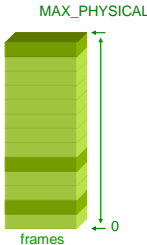

CS 3204 Spring 2006 3/22/2006 3

Physical Memory Management



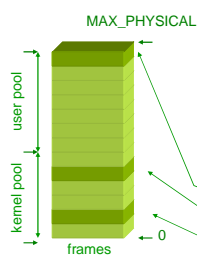
Physical Memory Management

- Aka frame table management
- Task: keep efficiently track of which physical frames are used
- Allocate a frame when paging in, or eager loading
- Deallocate a frame when process exits or when page is evicted (later)





CS 3204 Spring 2006 3/22/2006 5

Approach 1: Bitmaps



- Use bitmap to represent free, used pages
- Sometimes division in user & kernel pool
- Pintos (palloc.c) does that: keeps two bitmaps
 - Kernel pool: 010010
 - User pool: 0000001
- You will manage user pool only



CS 3204 Spring 2006 3/22/2006 6

Approach 2: Buddy Allocator

- Logically subdivide memory in power-of-two blocks
- Round up on allocation to next power of 2
- Split block on allocation (if necessary)
- Coalesce on deallocation (if possible)
 - Coalescing can be delayed
- Used in Linux: allocation requests are always multiple of pages, max blocksize is 4MB

CS 3204 Spring 2006
3/22/2006
7

Buddy Example - Allocation

CS 3204 Spring 2006
3/22/2006
8

Buddy Example - Deallocation

CS 3204 Spring 2006
3/22/2006
9

Fragmentation

- Def: *The inability to use memory that is unused.*
- Internal fragmentation:
 - Not all memory inside an allocated unit is used; rest can't be allocated to other users
- External fragmentation:
 - Impossible to satisfy allocation request even though total amount of memory > size requested

CS 3204 Spring 2006
3/22/2006
10

Internal Fragmentation

CS 3204 Spring 2006
3/22/2006
11

External Fragmentation

CS 3204 Spring 2006
3/22/2006
12

Buddy allocator & fragmentation

- Q.: what average internal fragmentation in buddy allocator?
 - in bitmap allocator?
 - in first-fit allocator from project 0?
- Q.: external fragmentation?

Page Size & Fragmentation

- How should a system's architect choose the page size? – Trade-Off
- Large pages:
 - Larger internal fragmentation
 - (not an issue if most pages are full...)
 - Page-in & write-back cost larger
- Small pages:
 - Higher overhead to store page table (more entries to maintain)
- Modern architectures provide support for "super pages" – 2MB or 4MB

Page Replacement

Page Replacement Algorithms

- Goal: want to minimize number of page faults (situations where a page must be brought in from disk.)
 - Also: want to reduce their cost (ideally, evict those pages from their frames that are already on disk.)
- Number of algorithms have been developed

Optimal & LRU

- Optimal:
 - "know the future"
 - Obviously impractical, just a benchmark for comparison/analysis
- FIFO – evict oldest page
 - Evict the frame that was accessed least recently.
- LRU – evict least recently used page
 - How do we know if a frame is accessed?
 - Hard to know, approximate via clock algorithm
 - Various versions of clock

1-bit Clock Algorithm

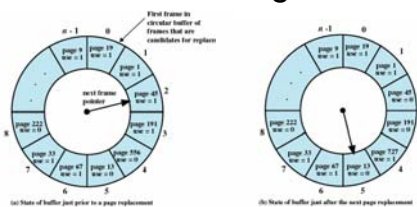


Figure 8.16 Example of Clock Policy Operation

- Note: frame pointer moves on allocation attempt

