Introduction to Virtual Memory in Nachos

Qin Liu

ACM Honored Class
Shanghai Jiao Tong University

Copyright Announcement

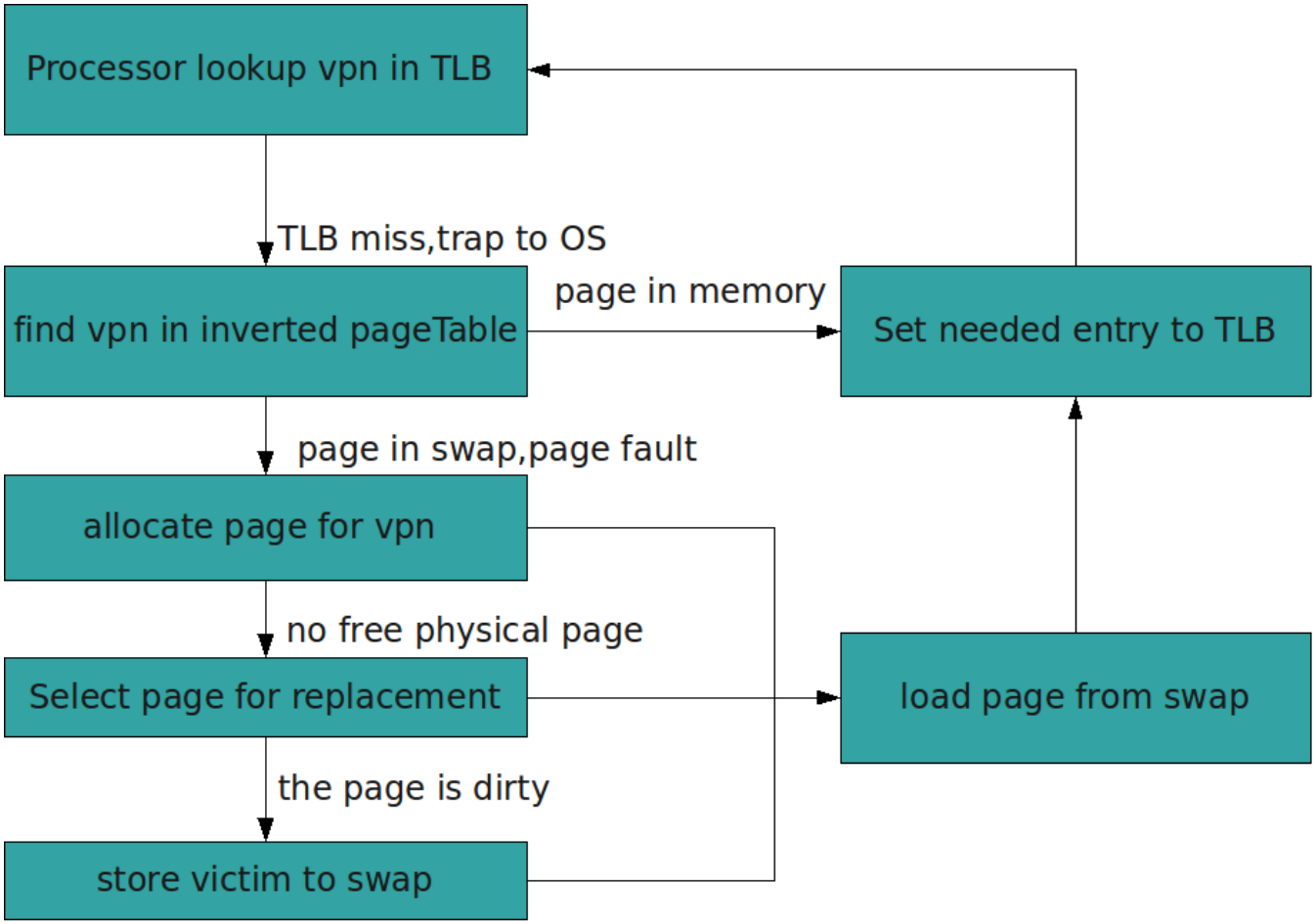
These slides were originally written by Tianqi Chen, and I add some new stuff.

Outline

Virtual Memory in Nachos

- Where do we store virtual pages in Nachos
  - Memory
  - Swapfile in Disk
  - The original file (load on demand)
- How do we map vpn (virtual page number) to physical memory
  - TLB(in processor)
  - Inverted page table(in memory)
- What if the page is not in memory
  - Find it in swapfile
  - Replace some page out if there is not enough physical memory
- How can we find the page in swapfile
  - Swapfile lookup list(in memory)

The Scenario



Key Questions in the Scenario

- What strategies are used in allocation and replacement.
- What data structures we need to support the scenario.
- What problems will arise when multiple threads running in the scenario.

Replacement Strategy

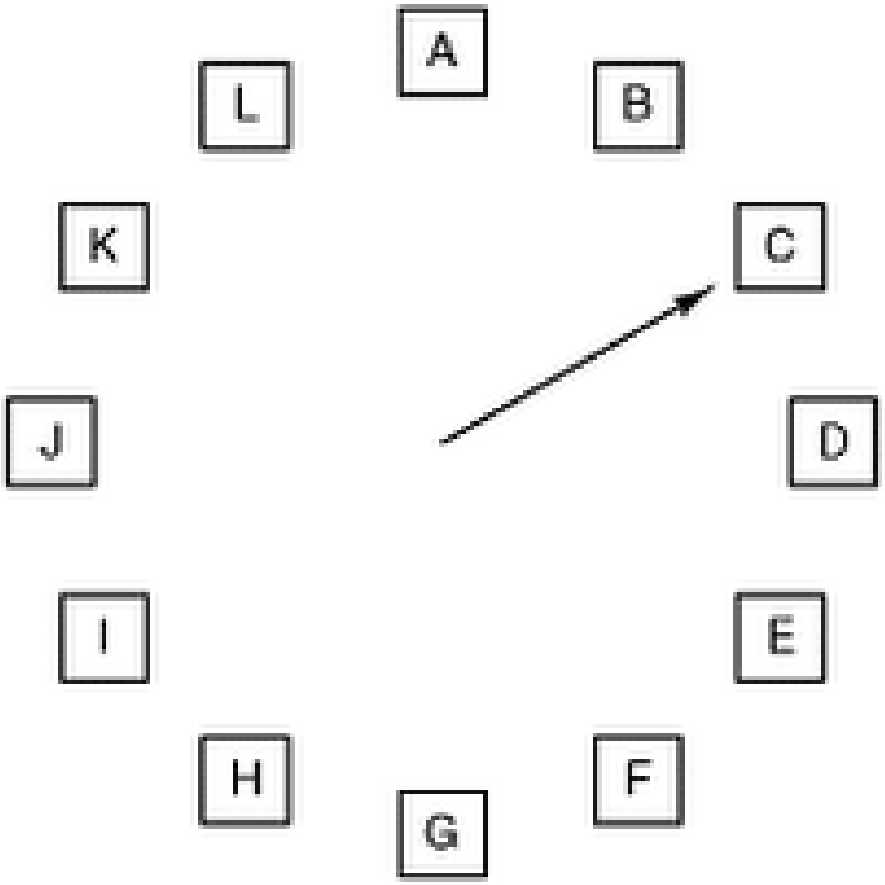
Questions:

- Which TLB entry to replace when resetting TLB.
- Which physical page to replace when swapping.

Replacement Strategies:

- Random
- Second-Chance
- Clock

Clock



When a page fault occurs, the page the hand is pointing to is inspected. The action taken depends on the R bit:

R = 0: Evict the page

R = 1: Clear R and advance hand

Swapping Strategy

- Keep all virtual pages in virtual memory until the program ends.
- *Write back* strategy, only write page when contents of the page are dirty.
- Reuse deallocated swap file pages.

Lazy Loading Strategy

- Load the excutable file when needed.
- Allocate stack page on demand.

Data Structures Supporting VM

- Global inverted page table (for paging).
- CoreMap that maintain information about physical page allocation (for replacement).
- Swapfile lookup list for information of swap file and reuse deallocated swap pages.

Checkpoints in Context Switch

- Update information(dirty) of PageTable from TLB (in SaveState).
- Flush TLB (in RestoreState).

Checkpoints When Swapping a Page Out

- Remove page information from
  - PageTable
  - CoreMap.
  - TLB.
- Update information in Swapfile look up list.

Checkpoints When Program Ends

- Remember to free the related information in
  - PageTable
  - CoreMap
  - Swapfile lookup list.
- When machine halts
  - Delete Swapfile

Synchronization Problems

You may get into troubles when you are working with multiple threads.

- Be aware of context switch.
- Note that context switch must occur during file operation.
- More problems may come which LotteryScheduler.

Example I: Live Lock

- #1 demand TLB Entry #1:1 for instruction.
- #1 execute a memory access instruction(sw).
- #1 demand TLB entry #1:2 for data.
- Context Switch, flush TLB.
- #2 demand TLB Entry #2:1 for instruction.
- #2 execute a memory access instruction(sw).
- #2 demand TLB entry #2:2 for data.
- Contex Switch, flush TLB.
- #1 demand TLB Entry #1:1 for instruction.
- ...

Example II: Multiple Allocation

- #1 demand page #1:1 , it select #2:1 to swap out.
- Context switch to #3 when saving #2:1
- #3 demand page #3:1 , it also select #2:1 to swap out.

Example III: Load after Save

- #1 demand page #1:1 , it select #2:1 to swap out.
- Context switch to #2 when saving #2:1
- #2 demand page #2:1 , try to load #2:1

Loading when saving can leads problem.

Solution

- Know every detail of every step.
- Use synchronization strategy.
- Run lots of extreme test (e.g fork 4 8queens using 5 physiscal pages).
- Good luck.

Q & A