Nachos Virtual Memory

Implementation notes
CS1550
Matt Craven
Assignment overview

• For this project, you are required to implement a virtual memory system within Nachos using a page table and related structures

• Your virtual memory system should enable programs that do not fit entirely within Nachos’ physical memory to execute successfully
Important points

• Nachos can support either a page table OR a TLB for virtual-to-physical address mapping, but not both
• You must implement a page table and NOT a TLB
• Please check my webpage for an updated Makefile to place in the vm/ directory
How Nachos handles memory accesses

• User programs on Nachos are MIPS programs
• The MIPS simulator included with Nachos handles load and store instructions by calling ReadMem on a load instruction and WriteMem on a store instruction
• These are defined in machine/translate.cc
ReadMem and WriteMem

- `bool Machine::ReadMem(int addr, int size, int* value);`
- `bool Machine::WriteMem(int addr, int size, int value);`
- Both functions call `Machine::Translate(int virtAddr, int* physAddr, int size, bool writing)` which is also in `machine/translate.cc`
Machine::Translate()

- This function checks the Page Table (userprog/addrspace.cc) to see if the desired virtual address is in memory, and if so, gets the physical page number where this address is stored
- If the virtual address does not have a valid physical page number associated with it, Translate returns a PageFaultException
PageFaultException

• If Machine::Translate() returns PageFaultException, this exception will be raised by ReadMem or WriteMem

• After this exception is raised, the operating system must make the desired page available in physical memory before the program is allowed to continue
Handling Exceptions

• When an exception is raised, Nachos will call ExceptionHandler (userprog/exception.cc)
• The “which” argument is the type of exception (ie, PageFaultException)
• Notice that the faulting address will be included in the exception (see ReadMem) - look at the comments in exception.cc see read this address
AddrSpace::AddrSpace()  
(userprog/addrspace.cc)

• A new AddrSpace instance is created when Nachos loads a user program
• By default, this AddrSpace will contain the minimum number of physical pages necessary to contain all the user program’s code and data
• If there is not enough physical memory, the ASSERT(numPages <= numPhysPages) will cause Nachos to quit
Loading a user program

- When Nachos instantiates an new AddrSpace(), it passes in a pointer to a Nachos file
- AddrSpace then uses ReadAt (filesys/openfile.cc) to copy the data into Nachos’ memory
- Note: ReadAt will NOT cause a PageFaultException
What you have to do

• For this assignment, you must:
  – Modify the page table to translate from virtual to physical page numbers
  – Handle PageFaultException
  – Come up with a way to store pages of memory to disk and retrieve them (hint: OpenFile)
  – Implement a replacement policy, to be used when a page must be evicted from physical memory
Where you must write your code

• All of your code changes should be in the userprog/ directory; however, you must do your make in the vm/ directory and run vm/nachos to test your virtual memory implementation

• There is no source code to edit in the vm/ directory

• You should not have to change anything in machine/ or in filesys/
Test Programs

• Some test programs are provided for you in the test/ directory
• Since these programs run on Nachos, you cannot include printf statements or other system calls in them
• The Exit() system call is provided for you - this will return a value to the shell - use this to make sure your program has executed correctly
Test Programs

• The matmult test program should be useful for testing your virtual memory system
• If you want to write your own test program, place your code in test/ and then add an entry to test/Makefile for your code, following the examples provided for the other test programs
Start Now!!!

• This is due on Wednesday, March 23 at midnight (11:59pm)
• The earlier you start, the more help we can give you
• Watch your email for updates, and instructions on how to schedule for a demo