Nachos Tutorial

Rong Zheng

What is Nachos 5.0j?

- Nachos stands for “Not Another Complete Heuristic Operating System “
- An instructional operating system in Java
- Includes many facets of a real OS:
  - Threads
  - Interrupts
  - Virtual Memory
  - I/O driven by interrupts
- You can (and will) modify and extend it

What is Nachos 5.0j?

- Nachos also contains some hardware simulation.
  - MIPS processor ✓
    - Can handle MIPS code in standard COFF, except for floating point instructions
    - You can (and will) write code in C, compile it to MIPS and run it on Nachos.
  - Console ✓
  - Network interface
  - Timer

How does Nachos work?

- Entirely written in Java
- Broken into Java packages:
  - nachos.ag (autograder classes)
  - nachos.machine (most of the action, Project 1)
  - nachos.network
  - nachos.security (tracks priviledge)
  - nachos.threads (Project 2, Project 3)
  - nachos.userprog (Project 4)
  - nachos.vm (Bonus project)

- More on Java doc

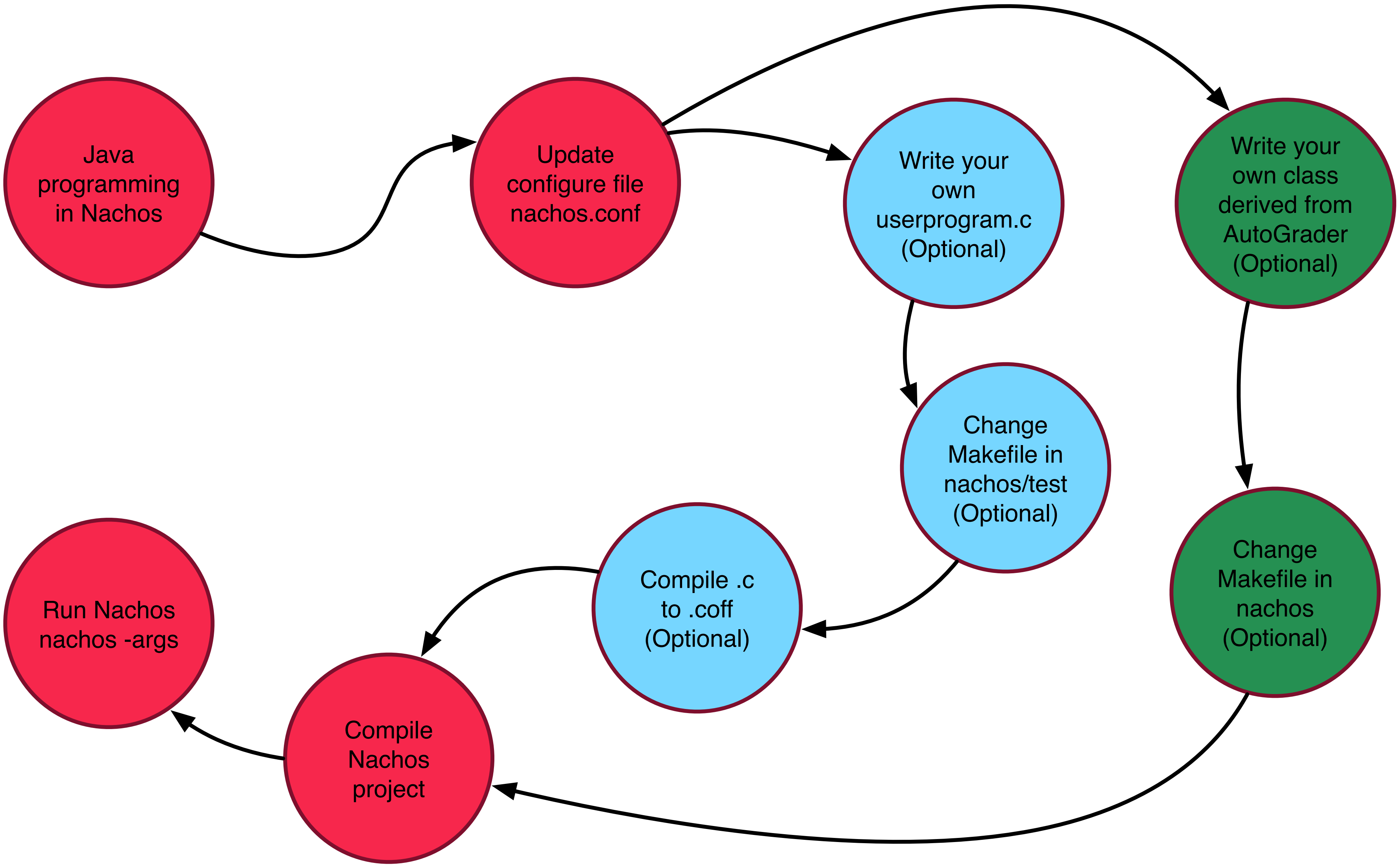
Installation

- Java 1.5 or higher
- Follow the Nachos tutorial instruction for the platform of your choice
- Easy → hard
  - ☺ VirtualBox + Kubuntu 32-bit preinstalled with Java, Nachos, cross compiler
  - 😐 Installing Nachos on OSX and Linux (cross-compiler for 64-bit not fully tested)
  - ☹ Installing Nachos on Windows/Cygwin

Cross-compiler

- Download the one for your own platform from here
  - *ppc* for power PC
  - *win32* for Windows/Cygwin 32bit
  - *64* for 64-bit
- Alternatively, download mips-x86.linux-xgcc.tgz to penguin.cas.mcmaster.ca (*need to connect via vpn from home*)

Nachos Project Workflow


Java programming in Nachos
Update configure file nachos.conf
Write your own userprogram.c (Optional)
Write your own class derived from AutoGrader (Optional)
Change Makefile in nachos/test (Optional)
Change Makefile in nachos (Optional)
Compile .c to .coff (Optional)
Run Nachos nachos -args
Compile Nachos project

Compilation

Kernel

- Changes to Java source codes
  - – Make changes to nachos/Makefile if necessary
    - In this class, only needed for adding new autograder class or possibly new classes in the bonus project
  - – Go to nachos/projx, make
- Changes to .c file under nachos/test
  - – Go to nachos/test
  - – Make changes to Makefile if necessary, make

User program

Executing Nachos

- nachos.conf contains configurations to run nachos

```
Machine.stubFileSystem = false #change to true for project 3
Machine.processor = false # change to true for project 3
Machine.console = false # change to true for project 3
Machine.disk = false
Machine.bank = false
Machine.networkLink = false
ElevatorBank.allowElevatorGUI = true
NachosSecurityManager.fullySecure = false
ThreadedKernel.scheduler = nachos.threads.RoundRobinScheduler
#nachos.threads.PriorityScheduler (for project 3)
Kernel.kernel = nachos.threads.ThreadedKernel #(change to nachos.userprog.UserKernel
 for project 4, and nachos.userprog.UserKernel  for bonus project)
```

More info see tutorial 2.5.1

Executing Nachos

- Command line options

-d	Enable some debug flags(see Table 8), e.g. -d ti
-h	Print this help message
-s	Specify the seed for the random number generator
-x	Specify a user program that `UserKernel.run()` should execute, instead of the default Kernel.shellProgram, e.g. `nachos -x halt.coff`
–	Specify an autograder class to use, instead of the default `nachos.ag.AutoGrader`
-#	Specify the argument string to pass to the autograder
-[]	Specifiy a config file to use, instead of the default `nachos.conf`

c	COFF loader info
i	HW interrupt controller info
p	processor info
m	disassembly
M	more disassembly
t	thread info
a	process info (formerly “address space”), hence a

Executing Nachos

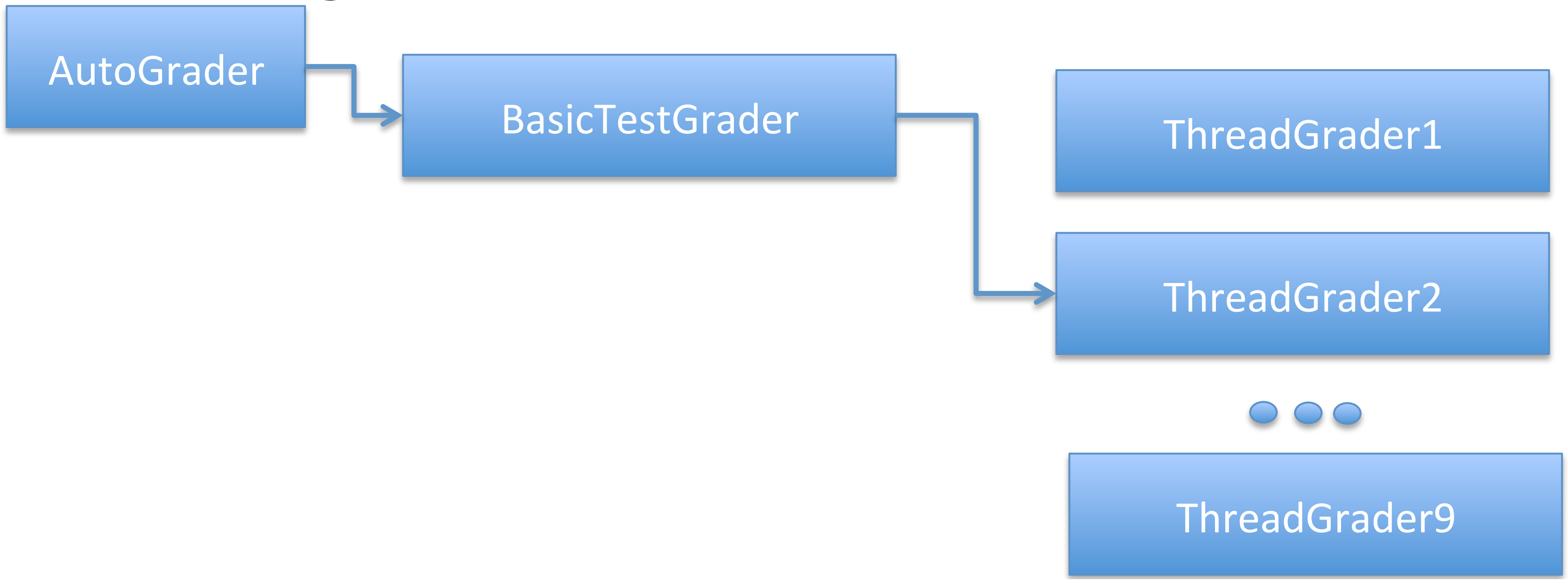
- Make sure ARCHDIR and PATH variables are correctly set
  - Use absolute path like /u30/rzheng/Lab/mips-x86.linux-xgcc/ instead ~/Lab/mips-x86.linux-xgcc/
- (Go to nachos/test, make clean; make)
- Go to nachos/proj1, make
- Run "nachos" with proper command line options
- Examples:
  - nachos
  - nachos –d t
  - nachos –x halt.coff

Common problems

- -bash: nachos: command not found
- Error: Could not find or load main class nachos.machine.Machine

Autograder

- Start from Project 2, you will be provided with additional classes that extend autograder for testing
- Use for debugging kernel implementations
- ThreadGraderxx extends BasicTestGrader by overriding run()



Run ThreadGraderxx

- Include the class in Makefile
- Make
- nachos -- nachos.ag.ThreadGraderxx

Project 1

- Goal
  - Nachos installation
  - Cross-compiler installation
  - Code tracing to understand the basic

Booting Nachos

- When you run Nachos, it starts in `nachos.machine.Machine.main`
- `Machine.main` initializes devices - interrupt controller, timer, MIPS processor, console, file system
- Passes control to the `autograder`.
- `AutoGrader` will create a kernel and start it (this starts the OS)

Demo

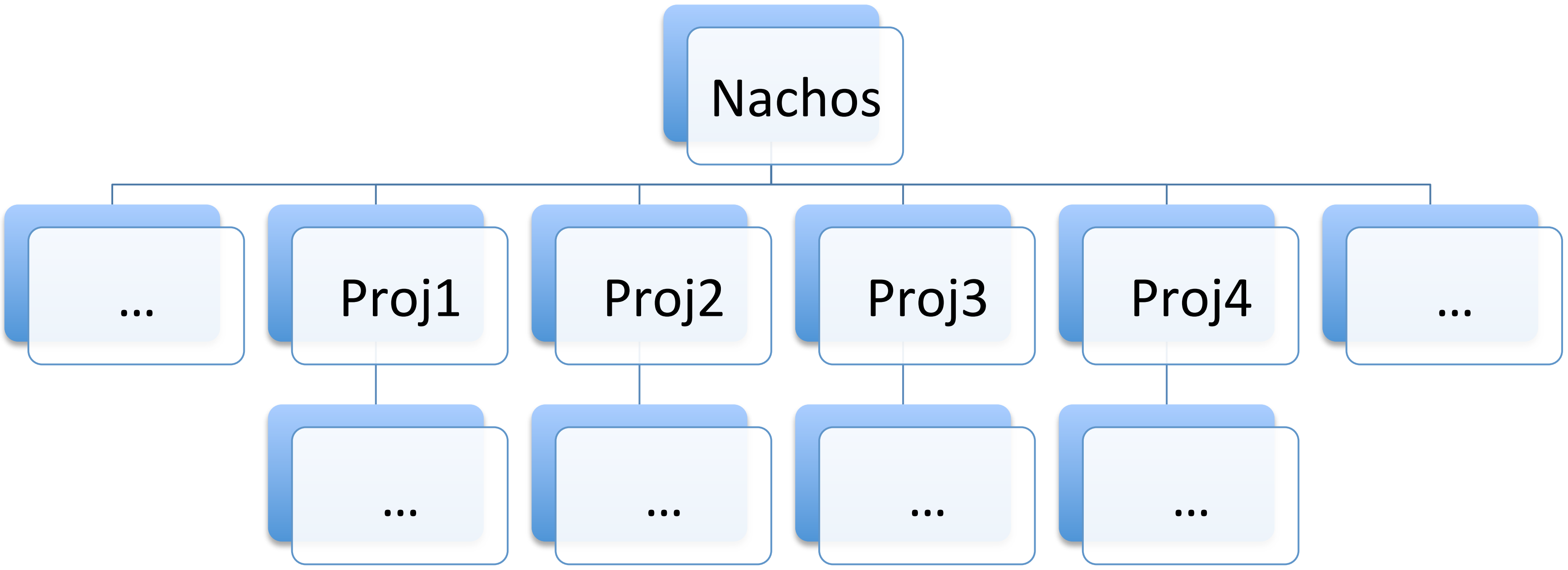
Suggestions

- Use eclipse or other IDEs from the start
  - Ease debugging
  - Tutorial 2.6
- Turn on relevant debugging options
- Follow the tutorial and trace codes
- Make incremental changes and test relentlessly

SVN

- After groups are set up, you can create/access your repository through
- https://websvn.mcmaster.ca/nachos/group#
- # is your group number
- Please keep the same nachos directory structure in your SVN repository to allow TA to fetch your codes for grading purposes
- SVN howto

Nachos directory structure



Important Deadlines

- MSAF not accepted for team projects
- Avenue discussion
- Group signup: Jan 16th
- Project 1 due Jan 23rd