Debugging and Profiling 2

Lecture 14
March 2nd 2023 | COMP 211-002 | Joshua Bakita

Welcome!

Today:

- More on I/O
  - `mmap()`
  - Performance profiling
- Debugging review

Logistics:

- 54% of the class has started on Assignment 3
- Tomorrow is the last day to drop the class with a "W" grade.

Fun fact...

Most terminals support `Ctrl+w` to delete the last word you typed, and `Ctrl+u` to delete the whole line.

Performance Profiling

Last time… but with detail

```
time <prog> <args>

\time -v <prog> <args>
```

- ➔ Good for quickly checking a program's runtime
- ➔ Low precision (millisecond-scale at best)
- ➔ Does not provide granular information about *what* is slow

```
perf record -F <freq> --call-graph
dwarf,2048 <prog> <args>
```

- ➔ Replace `<freq>` with sample frequency in HZ
- ➔ Creates the `perf.data` with profile data

Visualization:

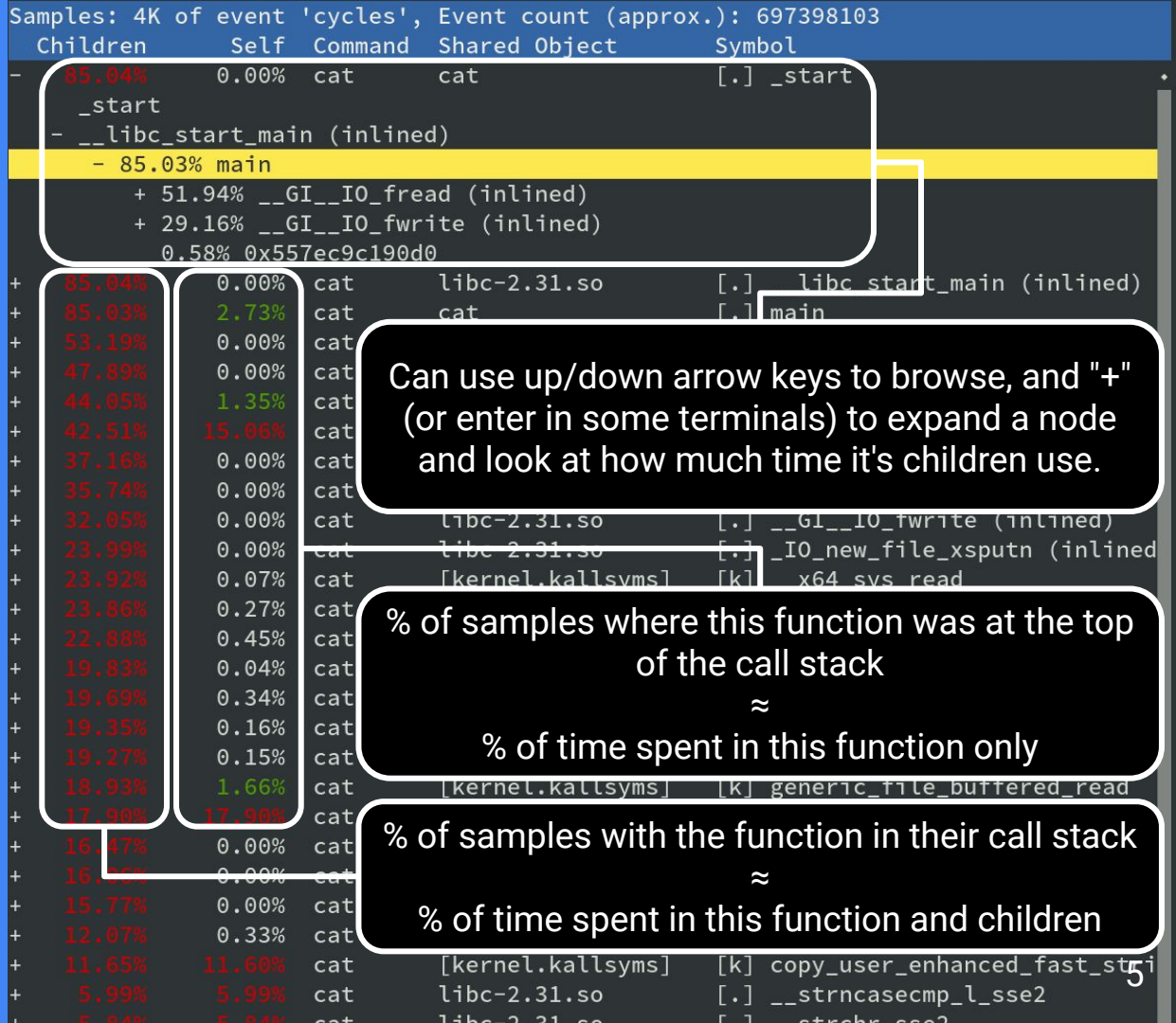
- ➔ Run `perf report` for an interactive viewer

Performance Profiling

`perf report`

➔ This is a profile of the `fread()` version of `./cat` from Lecture 8

Wouldn't it be nice if we could visualize this?



Last time… but with detail

```
time <prog> <args>

\time -v <prog> <args>
```

- ➔ Good for quickly checking a program's runtime
- ➔ Low precision (millisecond-scale at best)
- ➔ Does not provide granular information about *what* is slow

```
perf record -F <freq> --call-graph
dwarf,2048 <prog> <args>
```

- ➔ Replace `<freq>` with sample frequency in HZ
- ➔ Creates the `perf.data` with profile data

Visualization:

- ➔ Run `perf report` for an interactive viewer
- ➔ Run `/playpen/FlameGraph/211gen.sh` to generate a visualization of `perf.data` in `graph.svg`

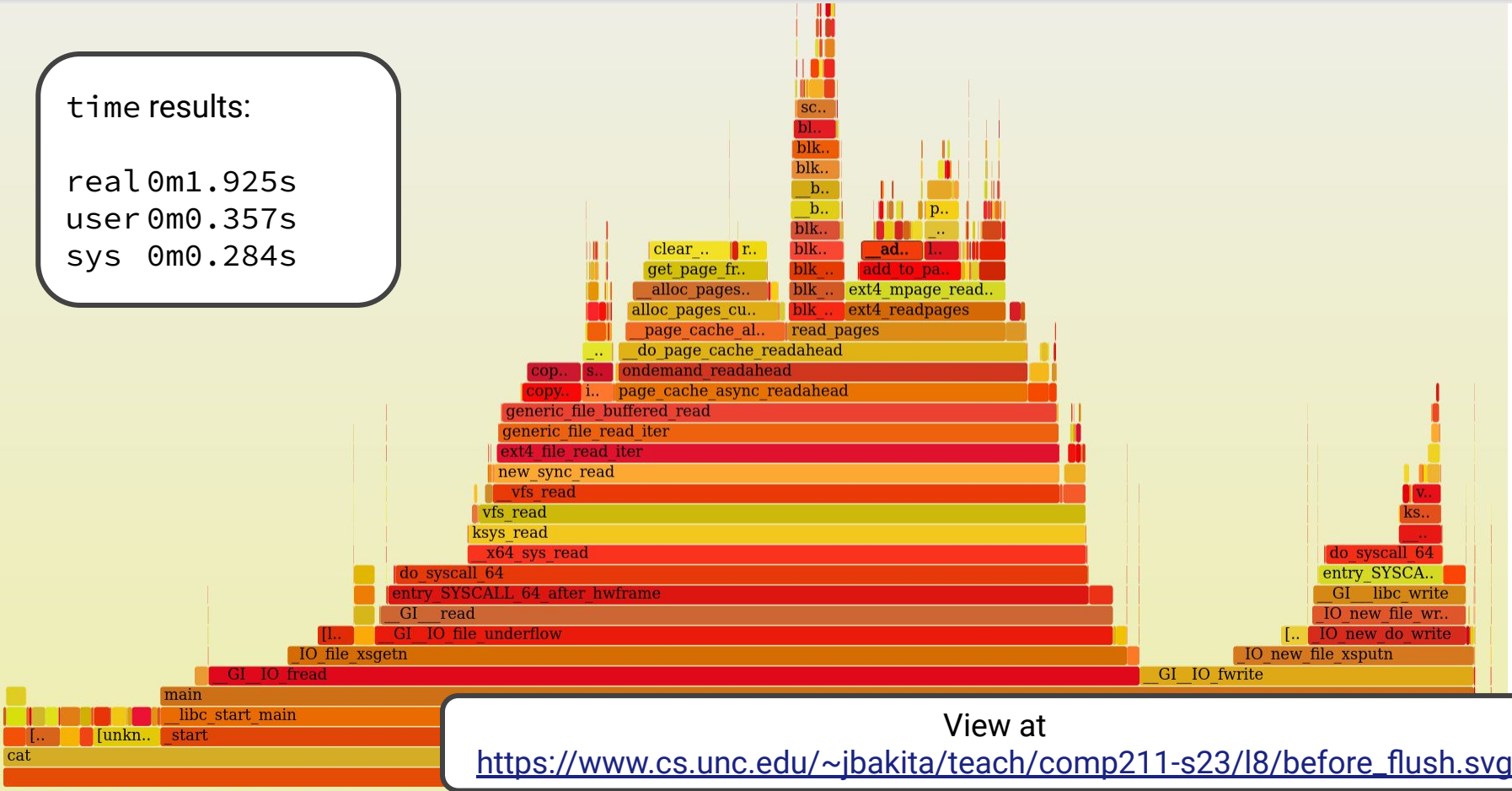
Visualizing Performance Profiles

FlameGraphs!

`./cat` with `fread()` and cold page cache

`time` results:

```
real 0m1.925s
user 0m0.357s
sys  0m0.284s
```



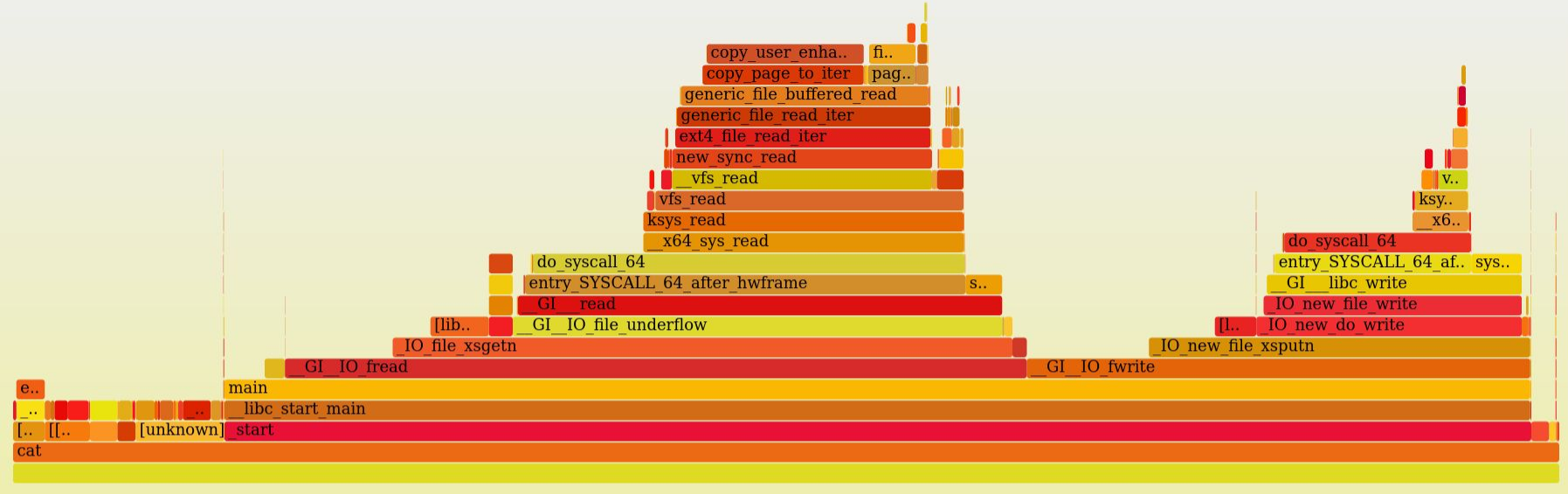
View at
https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l8/before_flush.svg

`./cat` with `fread()` and hot page cache

`time` results:

```
real 0m0.273s
user 0m0.181s
sys  0m0.092s
```

View at
https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l8/after_flush.svg



`./cat` with `mmap()`

https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l13/before_flush.svg

https://www.cs.unc.edu/~jbakita/teach/comp211-s23/l13/after_flush.svg

How Does I/O Really Work?

A sampling from one of my research presentations…

Debugging Revisited

Likely relevant to Assignment 3!

Command Line

`valgrind prog`	Run `prog` with valgrind
`gdb prog`	Start the GNU Debugger on `prog`
`info thing`	View detailed manual for `thing`
`xxd file`	Print `file` as hexadecimal
`wget addr`	Download file from `addr`
`rm file`	Delete `file`
`cd dir`	Move to `dir`
`cat file`	Print contents of `file`
`cp fileA fileB`	Copy `fileA` to `fileB`

Vim Commands (Normal Mode)

`dd`	Delete current line
`D`	Delete from cursor to end-of-line
`>>`	Increase indent
`<<`	Decrease indent
`O`	Add line above cursor and enter insert mode
`o`	Add line below cursor and enter insert mode

For Your ~/.vimrc Config File

```
set cindent
set nowrap
```

Key GNU Debugger (GDB) Commands—From L7

Full command name

Shorthand

Access the full GDB manual via `info gdb` on the command line

Control Flow

`backtrace`	`bt`	List all stack frames
`select <frame#>`	`sel`	Select a stack frame as your context
`next`	`n`	Execute the next line from your context
`step`	`s`	Execute one line
`list`	`l`	Print source code

Data

`print <expr>`	`p`	Execute expression and print result (can modify data)
`info locals`	`i lo`	Print value of every local variable in your stack frame
`x <addr>`	`x`	Print bytes at `addr` in memory
`whatis <expr>`	`wha`	Print type of `expr`

Breakpoints

`break <file>:<l>`
`break <function>`
`break <function> if <condition>`
	`b`	Set a breakpoint with optional condition at a location or function

`info breakpoints`	`i b`	List all breakpoints
`delete <breakpoint number>`	`d`	Delete a breakpoint
`continue`	`c`	Resume execution

Admin

`run <args>`	`r`	Run local program
`quit`	`q`	Exit GDB
`help <cmd>`	`h`	Print quick reference for a command
`set history save`		Save command history

Questions?

See office hour calendar on the website for availability.

Assignment 3 due Tuesday!

Contact:
Email: hacker@unc.edu
Twitter: @JJBakita
Web: https://cs.unc.edu/~jbakita

Old Well, University of North Carolina at Chapel Hill, Winter 2017