

CS 4414/5416: Systems Programming

Tools of the Trade

David Bindel

2026-09-24

Logistics

- Lab 5 is Friday: inspecting assembly, FFIs
 - This may be useful for P2 as well
- Next week begins the Unix abstractions model

- I have read all your comments – thank you
- Course staff has a grading party tonight
 - Yay.
- Final is on systems, not Rust

I am not a mind-reader – your comments help!

- You mostly like Rust (a few would prefer C++)
- Poll Everywhere seems like a popular switch
- Consensus is that I go too fast (OK, I hear you)
 - Help us all by *asking questions*
- Labs are divisive (except 4, of course)
- 4414/5416 differentiation is coming up

- Yes, I heard the practice was easier
 - Sorry, calibration is hard
 - Syntax $\neq$ types
- Prelim was inherited (will change next year)
- Course staff would like to see you!
- For a few of you: why do you want to be here?

Recap

This module: fundamentals of *single-thread* performance

- Hardware effects (particularly memory)
- Performance modeling and measurement
- Using the compiler and tools effectively

Concurrency and multi-threaded performance comes later.

- Single core exhibits lots of ILP
 - Compiler is good at helping out
 - Helps to code so independent ops are obvious
- But often memory is the bottleneck
 - Favor compact data, limit working sets, regular access

Combine HW context, tricks, and principles:

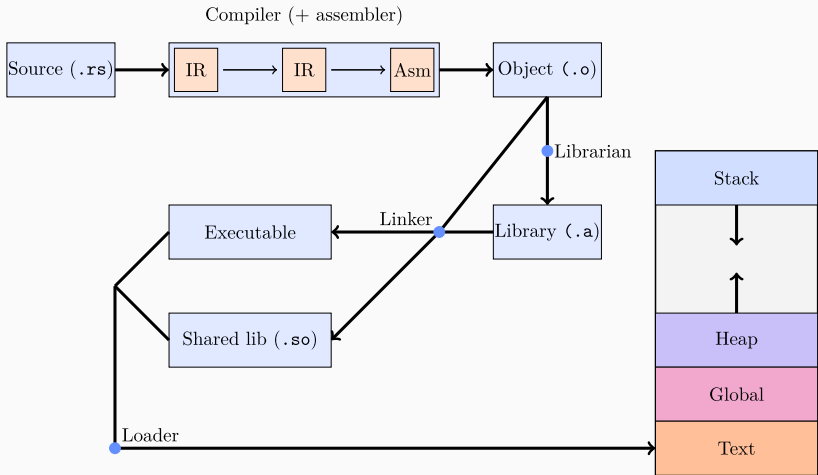
- Think before you write (last week)
- Stand on the shoulders of giants (last week)
- Time before you tune (last week)
- Tune your data structures (Tue)
- Help tools help you (Tue)

- A few words on build, compile, link, and load
- And then some time playing at the terminal

What is your plan for fall break?

Link and Load

The Pieces



- Compiler maps source to *object files*
 - Multiple intermediates, optimization passes
 - I am including “assembler” as part of this
 - Basic compilation unit is a crate
 - ... but for efficiency, may have codegen units
- Linker combines objects into library or exe
- Loader gets it all into memory for execution

Real world is complicated:

- Compiler front end encompasses lots
 - May call librarian, linker
- Cargo (or CMake, autotools) orchestrates
 - Configuration
 - Build

What Does a Linker Do?

- Figure out where everything should go
- Match symbols to memory locations
 - *Name mangling* for many languages
 - May be stripped out of final executable

Most work starts before execution (map stuff into memory).

What about after the start of execution? *Dynamic loader*

- DLL: Dynamic Load Library
- **.so**: Shared Object

Need *position-independent code* (relative jump offsets)

- Tool building
 - E.g. library interposition techniques
- Mixed-language programming
 - Linking in language support is a headache!
 - Also issues like name mangling

Chapter 7 in CS:APP3e is pretty good.

Ask Me a Question!

Help Tools Help You

- Linters (like clippy)
- Profilers (like samplify)
- Benchmark tools (like criterion)
- Static analysis (like llvm-mca)
- Dynamic analysis (like cachegrind)
- And, of course, the compiler...

What Can the Compiler Do?

```
RUSTFLAGS="-C opt-level=3 \  
          -C target-cpu=native \  
          -C llvm-args=-pass-remarks=loop-vectorize \  
          -C llvm-args=-pass-remarks-missed=loop-vectorize \  
          -C llvm-args=-pass-remarks-analysis=loop-vectorize \  
          -C llvm-args=-pass-remarks=slp-vectorizer \  
          -C llvm-args=-pass-remarks-missed=slp-vectorizer \  
          -C llvm-args=-pass-remarks-analysis=slp-vectorizer" \  
cargo build --release
```

- See cargo-pgo

To the Terminal!

Wrap-Up

Hardware context, some tricks, and some principles:

- Think before you write (last week)
- Stand on shoulders of giants (last week)
- Time before you tune (last week)
- Tune your data structures (this week)
- Help your tools help you (this week)

Next module: Unix abstractions

- Basic abstractions (files, pipes, VM)
- Processes and inter-process communication
- Introduction to sockets

May start to be more interesting for 5416!

- Lab tomorrow is on FFIs for C and viewing assembly
- P2 submission due 9/30

On Ed: I may do a (real) multi-lab project for the first three labs next year.
What do you think would be a compelling project?