T-diagrams

“Mommy, where do compilers come from?”
T-diagrams

- Different diagrams for different kinds of programs
- Visual explanation of interactions involving compilers and interpreters
Programs

Program P written in language L

Example: Sort program written in Java
Machines

Machine executing language $M$

Example: Sun workstation executing sparc machine code
Executing Programs

Program implementation language must match machine
Interpreters

Interpreter executing language L written in language M

Example: Lisp interpreter running on sparc

L
M

Lisp
sparc
Interpreting Programs

- Interpreter mediates between program language and machine language

```
sort
Lisp
Lisp
sparc
sparc
```
Virtual Machines

Interpreter creates a “virtual machine”
Compilers

Compiler translating from source language S to target language T implemented in M

Example: C compiler for sparc platform
Compiling Programs

Compiler inputs program in source language, outputs in target language
Java Programming Environment

*Javac*: Java to Java byte code (JBC) compiler

*Java*: Java Virtual Machine byte code interpreter
Where Do Compilers Come From?

1. Write it in machine code

A lot of work
Where Do Compilers Come From?

1. Write it in machine code
2. Write it in a lower level language and compile it using an existing compiler

But Mom where did the C compiler come from?
Where Do Compilers Come From?

1. Write it in machine code
2. Write it in a lower level language and compile it using an existing compiler
3. Write it in the same language that it compiles and bootstrap
Bootstrapping a Compiler

- Write the compiler in its own language (#0)
- Write a no-frills native compiler (#1)
- Use compiler #1 to compile #0 to get native compiler with more frills (#2)
- Repeat as desired
Bootstrapping a Compiler

#0 (the real thing)

#1 (no frills)

#2 (real thing compiled)
Bootstrapping a Compiler, Stage 2

#0 (the real thing)

#2 (the real thing compiled)

#3 (compiled with the real thing)

Correctness test: #2 = #3 literally
Porting a Compiler

1. Rewrite back end to target new machine

2. 

3. 

4. Compile on new machine
Porting a Compiler

1. Rewrite back end to target new machine

4. Compile on new machine
Porting a Compiler II

- Rewrite back end to target new machine
- Compile using native compiler
Cross Compilers

A cross compiler compiles to a target language different from the language of the machine it runs on.
Porting a Compiler II

- Rewrite back end to target new machine
- Compile using native compiler
- Recompile using cross compiler