How much does a compiler cost - and other assorted history

Russel Arbore

In the beginning…

- Programs written directly in assembly or machine code
- The first programs resembling compilers popped up in the 50s
  - Autocode by Alick Glennie in 1952
  - FORTRAN led by John Backus in 1957
  - ALGOL 58 by Friedrich Bauer in 1958
- The first multi-target compiler was the COBOL compiler - in 1960, a COBOL program was run on both the UNIVAC II and the RCA 501
- A first cost estimate: the FORTRAN compiler took 18 man years of development

IBM 704

The early 60s - self-hosting

- NELIAC was the first self-hosted compiler
  - Written in and implemented ALGOL in 1958 by Harry Huskey
  - Used small assembly bootstrap compiler
- LISP
  - First self-hosted compiler written in 1962 by Tim Hart and Mike Levin
  - Bootstrapped from an existing LISP interpreter

Program Optimization

- Pioneered by Frances Allen and John Cocke
- Introduced using graph data structures to encode programs in *Program Optimization*, 1966
- Introduced program intervals and basic data flow analysis in *Control Flow Analysis* and *A Basis for Program Optimization*, both in 1970
- *A Catalogue of Optimizing Transformations* in 1971 described many new optimizations
- *A Program Data Flow Analysis Procedure* in 1976 described data flow analysis as we know it today

Program Optimization, in the wild

- Peephole optimizations were introduced by William McKeeman in 1965, used in the XPL compiler
  - XPL is a derivative of PL/I
- Capex Corp. developed the “COBOL Optimizer” in the mid 1970s
  - Patched specific patterns in object code generated by the IBM COBOL compiler

The rise of C

- In 1964, MIT sought to build a successor to their Compatible Time Sharing System, called Multics
- Decided to use the language PL/I, *before a compiler for PL/I existed*
- The PL/I compiler only arrived in 1966
- Many companies pulled out of Multics due to the clown show
  - Digitek was contracted to build the PL/I compiler, and totally flopped
  - A subset of the Multics folks developed EPL (early PL/I) and developed a compiler on their own
- Bell Labs pulls out of the project in 1969

The rise of C

- Ken Thompson, who was previously working on Multics, implemented a new operating system for the PDP-7
- Later ported to PDP-11 - called “Unix”
  - Implemented in assembly
- The original high level language was “B” - an interpreted language
- Dennis Ritchie and Ken Thompson wanted to continue developing the OS in a high level language, but B wasn’t going to cut it - C!
  - B consumed too much memory

The rise of C

- C grew features as needed to implement Unix
  - Structs
  - Bitfields
  - Preprocessor
- Grew organically, but standardization came late
- C came to dominate systems software in the late 70s and 80s
  - Almost every operating system were written in C
  - Many companies wrote their own C compilers for their own machines
- Unfortunately, Ritchie's *The Development of the C Language* is a little sparse on details specifically about the C compiler

The GNU Project

- *The GNU Manifesto* was released in 1983
- At that point, GCC had already started development
- GCC was released in 1987, written from scratch by Richard Stallman and others
- By the 90s, GCC out-performed many vendor C compilers, supported 13 architectures, and was used by several companies

C++

- Started as C with Classes in 1979
- C++ born as a successor to CwC in 1982
  - Virtual functions
  - Overloading
  - References
  - Many other features…
- Developed in a standalone compiler, Cfront
- Cfront translated C++ to C
- Itself written in C++
- Cfront was abandoned in 1993 after failing to add exceptions

C/C++ Compilers in the 90s

C/C++ Compilers in the 90s

- Many companies pooled together funds for the GNU C compiler
- GCC improved drastically throughout the 90s and early 2000s
- Implemented a C++ frontend
- A C/C++ compiler no longer became a constraint of a new system

Early 2000s: LLVM

- Chris Lattner first described LLVM in his 2002 Master's thesis
- While GCC was an amazing compiler, it was not modular / reusable / extensible
- LLVM is a library compiler
- Allowed LLVM to be used for…
  - Databases
  - Shader compilers
  - GPGPU (CUDA is built on LLVM)
  - HLS tools
  - JIT
- GCC is architecturally incapable of doing everything above

Late 2000s and 2010s

- Lots of development in interpreted / bytecode languages / transpilers
- Javascript
  - AJAX and related ideas allowed for dynamic web apps w/o the need to reload the page
  - Google released Chrome 2008, featuring the V8 JavaScript engine, featuring a JIT compiler
  - Transpilers from languages like TypeScript, CoffeeScript, Elm, Dart
- WebAssembly
  - JavaScript turned into a transpiler target / IR code format
  - New idea: create a lower level code format for web browsers to execute
  - LLVM has a WebAssembly backend now!
- Additionally, LLVM became more entrenched w/ usage by Apple, CUDA, OpenCL

Compilers in the modern day

- Increasingly targeting heterogeneous devices
  - GPUs
  - Accelerators
  - SmartNICs
- Increasingly focusing on specific workloads
  - DL
  - DL
  - DL

Compilers in the modern day

So how much does this all cost?

- Compilers are infamously difficult to develop
- Hard to quantify
  - $ / sloc?
  - What about debuggers, linkers, assemblers, or standard libraries (the toolchain)?
- What parts are more expensive (frontend errors vs. optimization vs. code generation vs. testing)?

So how much does this all cost?

Project	Started	Developers	LOC	Estimated Cost
FreePascal	2005	54	3,845,685	$198,619,187
GCC 9.2.0	1988	617	5,591,759	$425,747,278
Glasgow Haskell Compiler 8.8.1	2001	1,119	761,097	$52,449,098
Intel Graphics Compiler 1.0.10	2018	149	684,688	$46,934,626
LLVM 8.0.1	2001	1,210	6,887,489	$529,895,190
OpenJDK 14+10	2007	883	7,955,827	$616,517,786
Roslyn .NET 16.2	2014	496	2,705,092	$198,619,187
Rust 1.37.0	2010	2,737	852,877	$59,109,425
Swift	2010	857	665,238	$45,535,689
v8 7.8.112	2008	736	3,048,793	$225,195,832

Table 1.1: Development history, logical lines of code (LOC), and estimated cost of 10 popular open-source compiler projects. Estimated costs are calculated using a COCOMO model [Dav01] with average 2019 US software developer salaries [Gla19].

So how much does this all cost?

- Intel Graphics Compiler, Rust, and Swift are all LLVM based
- GHC is known for being very small, Haskell is also generally more compact than other languages

Project	Started	Developers	LOC	Estimated Cost
FreePascal	2005	54	3,845,685	$198,619,187
GCC 9.2.0	1988	617	5,591,759	$425,747,278
Glasgow Haskell Compiler 8.8.1	2001	1,119	761,097	$52,449,098
Intel Graphics Compiler 1.0.10	2018	149	684,688	$46,934,626
LLVM 8.0.1	2001	1,210	6,887,489	$529,895,190
OpenJDK 14+10	2007	883	7,955,827	$616,517,786
Roslyn .NET 16.2	2014	496	2,705,092	$198,619,187
Rust 1.37.0	2010	2,737	852,877	$59,109,425
Swift	2010	857	665,238	$45,535,689
v8 7.8.112	2008	736	3,048,793	$225,195,832

Table 1.1: Development history, logical lines of code (LOC), and estimated cost of 10 popular open-source compiler projects. Estimated costs are calculated using a CO-COMO model [Dav01] with average 2019 US software developer salaries [Gla19].

So how much does this all cost?

- Intel Graphics Compiler, Rust, and Swift are all LLVM based
- GHC is known for being very small, Haskell is also generally more compact than other languages
- It depends - 10^6 to 10^8 for a production compiler.

Project	Started	Developers	LOC	Estimated Cost
FreePascal	2005	54	3,845,685	$198,619,187
GCC 9.2.0	1988	617	5,591,759	$425,747,278
Glasgow Haskell Compiler 8.8.1	2001	1,119	761,097	$52,449,098
Intel Graphics Compiler 1.0.10	2018	149	684,688	$46,934,626
LLVM 8.0.1	2001	1,210	6,887,489	$529,895,190
OpenJDK 14+10	2007	883	7,955,827	$616,517,786
Roslyn .NET 16.2	2014	496	2,705,092	$198,619,187
Rust 1.37.0	2010	2,737	852,877	$59,109,425
Swift	2010	857	665,238	$45,535,689
v8 7.8.112	2008	736	3,048,793	$225,195,832

Table 1.1: Development history, logical lines of code (LOC), and estimated cost of 10 popular open-source compiler projects. Estimated costs are calculated using a CO-COMO model [Dav01] with average 2019 US software developer salaries [Gla19].

A smaller case study

- https://www.embecosm.com/2018/02/26/how-much-does-a-compiler-cost/
- Goal: develop an LLVM based compiler to target a custom DSP
  - Basically write a backend for a weird, 16-bit word processor
- Took 120 engineer days from 5 experienced compiler engineers
- 120 engineer days costs less than $1,000,000

A reminder


NVIDIA
CUDA
cadence
plaidML
oneAPI
AMD
ROCm
SYCL
tvm
Mentor
A Siemens Business
nGraph
XILINX
VITIS
AI
VIVADO

A reminder

My guess? Several billion dollars.

“The next LLVM”

- MLIR is the chosen successor, but hasn’t seen as much adoption *yet*
- Projects using MLIR don’t play nicely *with each other*
- There is interest in companies, but LLVM already works very well
- Rust, Swift, and Clang have not transitioned to MLIR based IRs yet

Hopefully not the foreseeable future…