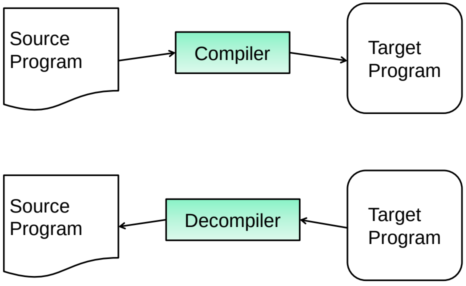
Vignettes

David Notkin
Autumn 2008

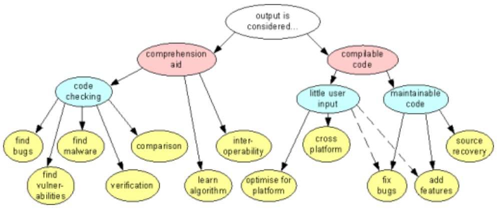
Various compiler-related topics

- Decompiling (reverse engineering)
- Obfuscation
- Syntax-directed editing
- Tools
- IDLs (intermediate description languages)
- JIT
- Multicore
- Attribute grammars
- …

Decompiling (reverse engineering)



Why might you want to do this?



Program-Transformation.Org

Another reason

Y2K: Year 2000 Projects May Be Overlooking Millions of Lines of Missing Computer Code

Edge: Work-Group Computing Report, April 5, 1999

How? (Kind of roughly sort of the reverse of compiling)

- Parse binary (often called loading)
  - Often can infer the architecture (if needed), the main entry point, etc.
  - Goal is to produce assembly language
- Disassembly
  - Convert assembly language to intermediate language
  - May exploit idioms aggressively
- Analysis
  - Program analysis: e.g., combine binary expressions into higher-level constructs
  - Type analysis: infer types of variables
  - Structure analysis: find control structures
- Generate high-level code

http://www.dmoz.org/Computers/Programming/Disassemblers/ 1 December 2008

- …
- AVATAR - A disassembler/patcher/code-explorer for PA-RISC based HP-UX systems, by Allegro Consultants, Inc.
- the bastard disassembler - *NIX disassembler. Written in C on Linux for x86 ELF files; intended to support multiple CPUs, OSes, and file formats. Scriptable.
- The dcc Decompiler - It decompiles small .exe files from the (i286, DOS) platform to C programs.
- DSP5600x - A 5600x disassembler by Miloslaw Smyk.
- FARGDIS - Fargo Disassembler for TI-92 DOS versioin, by John Grafton.
- High Level Assembler and Toolkit (HLASM) - System/390 assembler toolkit for MVS and VM and VSE (HLASM) including disassembler, by IBM.
- IDA Pro - The multi-processor, multi-OS, interactive disassembler, by DataRescue.
- MacNosy - Is a Mac application that disassembles the Macintosh ROM or any 68K or PowerPC codes, by Jasik Designs.
- MELPS7700 Disassembler - By H.Kashima.
- Misosys Disassembler - Tim Mann's TRS-80 Page includes Misosys Disassembler, aka PRO-DUCE.
- Open Reverse Code Engineering - An open community site offering a number of services including blogs, forums, download and reference libraries.
- Re39 - Interactive Disassembler for Rockwell C29/C39 (C40) code by Lewin A.R.W. Edwards.
- Reverse Engineering Compiler - Program that tries to make source coden (C) from binary, multiplatform. There are MIPS disassembler too, by Giampiero Caprino.
- SST Global-Decompilers - Decompilers for IBM midrange systems.
- …

Reverse engineering

- A generalization of decompiling
- Accept a lower-level model as input, produce a higher-level model
  - Common example: inferring UML from source code
- Roughly same motivations, etc.
- Inference may be more difficult

Legality?

- So, is decompiling/reverse engineering legal, illegal, or somewhere in between?
- Why?

It depends

- Copyright law applies to most programs: the owner of the copyright generally has a set of exclusive rights, including making copies (including those in memory)
- Decompilation and reverse engineering usually requires making of copies, so it requires permission of the copyright owner
- However, if decompilation is needed to attain interoperability, US and European copyright laws permit it in some cases
  - One US example allowed a company to decompile to get around a software locking mechanism for a Sega game console

Europe: 1991 Software Directive

- Explicit right to decompile for interoperability only if:
  - The program must be properly licensed
  - Decompilation must be necessary and the burden is on the decompiler to show that manuals, API documents, etc. is insufficient
  - The process must be as confined as much as possible to the parts relevant to interoperability.
  - Decompiled information may only be used for the specific interoperability purpose and may not be shared

Obfuscation

- Making source (or intermediate) code very hard to read, usually intentionally
- Why?
  - IP protection
  - Malicious intent
  - Reduce security exposure
  - Minimization
- Why not?

```
_(__,___,____){___/__<=1?_(__,___+1,____):!(___%__)?_(__,___+1,0):_
__%__==___/
__&&!____?(printf("%d\t",___/__),_(__,___+1,0)):___%__>1&&___%__<__
_/__?_(__,1+
___,____+!(___/__%(___%__))):___<__*__?_(__,___+1,____):0;}main(){_
(100,0,0);}
```

Syntax-directed editing

- Why have programmers take their unambiguous ideas about a program, and then enter text that is then parsed using a potentially ambiguous process?
- Why not provide an editor that is knowledgeable about the abstract syntax (and some semantics) to ensure that programs are entered unambiguously and without syntactic error?

- Research around 1980’s: Gandalf, Cornell Program Synthesizer, Mentor, …

Didn’t work

- Syntax is not generally the problem for experienced programmers
- Integration with other tools (debugger, compiler, etc.) was much harder
- Some new problems appeared: for example, searching for unparsed text that didn’t appear in the AST
- …

- But influence modern environments in several ways