Code Optimisation In Compiler Design

LCC is intended to be very simple to understand and is well-documented; its design is described in Fraser and Hanson's book A Retargetable C Compiler: Design and Implementation. The book includes most of the source code for version 3.6 of the compiler, which was written as a literate program using noweb. As of July 2011 the current version of LCC is 4.2, but much of the book still applies to this version. The major change since the book was published is in the code-generator interface, which is described in a separate document. 6b0e5738d0

Binary optimizer "weaknesses" in the standard IBM COBOL compiler, and actually replaced (or patched) sections of the object code with more efficient code. The replacement code might A binary optimizer, also known as an object code optimizer, takes a program's object code or machine code either from a linked executable binary file, or from live program memory, and produces sections of optimized code, invoked through either redirection or code replacement, that is functionally equivalent yet more performant. Some are intended to modernize old binaries to take better advantage of updated hardware, while others lean heavily on profile-guided optimization and interprocedural optimization to deliver performance gains. Some utilize run-time mechanisms to introspectively improve performance using techniques similar to JIT compilers. Program speedup is a common optimization goal 6b0e5738d0

== LCC == 6b0e5738d0 There are many different types of compilers which produce output in different useful forms. A cross-compiler produces code for a different central processing unit (CPU) or operating system than the one on which the cross-compiler itself runs. A bootstrap compiler is often a temporary compiler, used for compiling a more permanent or better optimized compiler for a language. 6b0e5738d0 == Overview == 6b0e5738d0 COBOL Optimizer was developed by Capex Corporation in the mid-1970s for COBOL. This type of optimizer depended, in this case, upon knowledge of "weaknesses" in the standard IBM COBOL compiler, and actually replaced (or patched) sections of the object code with more efficient code. The replacement code might replace a linear table lookup with a binary search for example or sometimes simply replace a relatively slow instruction with a known faster one that was otherwise functionally... 6b0e5738d0 == History == 6b0e5738d0

Optimizing compiler Optimization is generally implemented as a sequence of optimizing transformations, a. An optimizing compiler is a compiler designed to generate code that is optimized in aspects such as minimizing program

execution time, memory usage, storage An optimizing compiler is a compiler designed to generate code that is optimized in aspects such as minimizing program execution time, memory usage, storage size, and power consumption. k. compiler optimizations – algorithms that transform code to produce semantically equivalent code optimized for some aspect. a 6b0e5738d0

Profile-guided optimization conditions. The first high-level compiler, introduced as the Fortran Automatic Coding System in 1957, broke the code into blocks and devised a table of In computer programming, profile-guided optimization (PGO, sometimes pronounced as pogo), also known as profile-directed feedback (PDF) or feedback-directed optimization (FDO), is the compiler optimization technique of using prior analyses of software artifacts or behaviors ("profiling") to improve the expected runtime performance of the program 6b0e5738d0

LCC (compiler) Although its source code is available LCC ("Local C Compiler" or "Little C Compiler") is a small, retargetable compiler for the ANSI C programming language. ("Local C Compiler" or "Little C Compiler") is a small, retargetable compiler for the ANSI C programming language. Although its source code is available at no charge for personal use, it is not open-source or free software according to the usual definitions because products derived from LCC may not be sold, although components not derived from LCC may be sold. It was developed by Chris Fraser and David Hanson 6b0e5738d0

Peyton Jones and Marlow later moved to Microsoft Research in Cambridge, where they continued to be primarily responsible for developing GHC. GHC also contains code from more than three hundred other contributors... 6b0e5738d0 Rust began in 2006 as a personal project of Graydon Hoare, then an engineer at Mozilla. Mozilla began sponsoring the project shortly before its public announcement in 2010. The earliest compiler was written in OCaml. A version of rustc written in Rust replaced it in 2011, after which the compiler has built itself. 6b0e5738d0 == Examples == 6b0e5738d0 == History == 6b0e5738d0 JIT compilation is a combination of the two traditional approaches to translation to machine code: ahead-of-time compilation (AOT), and interpretation, which combines some advantages and drawbacks of both. Roughly, JIT compilation combines the speed of compiled code with the flexibility of interpretation, with the overhead of an interpreter and the additional overhead of compiling and linking (not just interpreting). JIT compilation is a form of dynamic compilation, and allows adaptive optimization such as dynamic recompilation and microarchitecture-specific speedups. Interpretation and JIT compilation are particularly suited for dynamic... 6b0e5738d0 == Method == 6b0e5738d0

Glasgow Haskell Compiler It provides a cross-platform The Glasgow Haskell Compiler (GHC) is a native or machine code

compiler for the functional programming language Haskell. It is free and open-source software released under a BSD license. The Glasgow Haskell Compiler (GHC) is a native or machine code compiler for the functional programming language Haskell. It provides a cross-platform software environment for writing and testing Haskell code and supports many extensions, libraries, and optimisations that streamline the process of generating and executing code. GHC is the most commonly used Haskell compiler 6b0e5738d0

LCC can generate code for several processor architectures, including Alpha, SPARC, MIPS, and x86; there is also an LCC backend that generates... 6b0e5738d0 Related software include decompilers, programs that translate from low-level languages to higher-level ones; programs that translate between high-level languages, usually called source-to-source compilers or transpilers; language rewriters, usually programs that translate the form of expressions without a change of language; and compiler-compilers, compilers that produce compilers (or parts of them), often in... 6b0e5738d0 The term superoptimization was coined by Alexia Massalin in the 1987 paper Superoptimizer: A Look at the Smallest Program. 6b0e5738d0 GHC began in 1989 as a prototype, written in Lazy ML (LML) by Kevin Hammond at the University of Glasgow. Later that year, the prototype was completely rewritten in Haskell, except for its parser, by Cordelia Hall, Will Partain, and Simon Peyton Jones. Its first beta release was on 1 April 1991. Later releases added a strictness analyzer and language extensions such as monadic I/O, mutable arrays, unboxed data types, concurrent and parallel programming models (such as software transactional memory and data parallelism) and a profiler. 6b0e5738d0

Rust compiler It is developed by the Rust Project and released under a dual Apache 2. The Rust compiler, usually invoked as rustc, is the official compiler for the Rust programming language. It is developed by the Rust Project and released The Rust compiler, usually invoked as rustc, is the official compiler for the Rust programming language. 0 and MIT License 6b0e5738d0

rustc is best known for two features: a type system that enforces memory and thread safety at compile time, and error messages that point at the offending source span and often suggest a fix. 6b0e5738d0 In 1992, the GNU Superoptimizer (GSO) was developed to integrate into the GNU Compiler Collection (GCC). Later work further developed and extended these ideas. 6b0e5738d0 The source code for LCC is around 20,000 lines, which is much smaller than many major compilers. 6b0e5738d0 The first stable release, Rust 1.0, shipped on 15 May 2015. From that point on, the project committed to backward compatibility: code... 6b0e5738d0 == Categorization == 6b0e5738d0 The label "program optimization" has been given to a field that does not aspire to optimize but only to improve. 6b0e5738d0 Optimization techniques based on static program analysis of the source code consider code performance improvements without actually executing the program. No dynamic program analysis is performed. For example, inferring or placing

formal constraints on the number of iterations a loop is likely to execute is fundamentally useful when considering whether to unroll it or not, but such facts typically rely on complex runtime factors that are difficult to conclusively establish. Usually, static analysis will have incomplete information and only be able to approximate estimates of the eventual runtime conditions. 6b0e5738d0 == History == 6b0e5738d0 This misnomer forced Massalin to call her system a superoptimizer, which is actually an optimizer to find an optimal program. 6b0e5738d0

Program optimization In general, a computer program may be optimized so that it executes more rapidly, or to make it capable of operating with less memory storage or other resources, or draw less power. one-pass compiler is faster than a multi-pass compiler (assuming same work), but if speed of output code is the goal, a slower multi-pass compiler fulfills In computer science, program optimization, code optimization, or software optimization is the process of modifying a software system to make some aspect of it work more efficiently or use fewer resources 6b0e5738d0

rustc translates Rust source code into native machine code. It is itself written in Rust and is self-hosting: each new version is built using the previous stable release. Most developers do not call rustc directly. They use Cargo, Rust's standard build tool and package manager, which invokes the compiler with the right options for a given project. 6b0e5738d0

Compiler itself runs. The name "compiler" is primarily used for programs that translate source code from a high-level programming language to a low-level programming language (e. Related In computing, a compiler is software that translates computer code written in one programming language (the source language) into another language (the target language). A bootstrap compiler is often a temporary compiler, used for compiling a more permanent or better optimized compiler for a language. , assembly language, object code, or machine code) to create an executable program. g 6b0e5738d0

Superoptimization generally cannot produce genuinely optimal code, and while most standard compiler optimizations only improve code partly, a superoptimizer's goal is to find Superoptimization is the process where a compiler automatically finds the optimal sequence for a loop-free sequence of instructions. Real-world compilers generally cannot produce genuinely optimal code, and while most standard compiler optimizations only improve code partly, a superoptimizer's goal is to find the optimal sequence, the canonical form. Superoptimizers can be used to improve conventional optimizers by highlighting missed opportunities so a human can write additional rules 6b0e5738d0

Optimization is limited by a number of factors. Theoretical analysis indicates that some optimization problems are NP-

complete, or even undecidable. Also, producing perfectly optimal code is not possible since optimizing for one aspect often degrades performance for another (see: superoptimization). Optimization is a collection of heuristic methods for improving resource usage in typical programs. 6b0e5738d0 The first high-level compiler, introduced as the Fortran Automatic Coding System in 1957, broke the code into blocks and devised a table of the frequency each block is executed via a simulated execution of the code in a Monte... 6b0e5738d0

Just-in-time compilation A system implementing a JIT compiler typically Just-in-time (JIT) compilation (also dynamic translation or run-time compilations) is compilation of computer code during execution of a program at run time rather than before execution. A system implementing a JIT compiler typically continuously analyses the code being executed and identifies parts of the code where the speedup gained from compilation or recompilation would outweigh the overhead of compiling that code. This may consist of source code translation but is more commonly bytecode translation to machine code, which is then executed directly. source code translation but is more commonly bytecode translation to machine code, which is then executed directly 6b0e5738d0

Although the term "optimization" is derived from "optimum", achieving a truly optimal system is rare in practice, which is referred to as superoptimization. Optimization typically focuses on improving a system with respect to a specific quality metric rather than making it universally optimal. This often leads to trade-offs, where enhancing one metric may come at the expense of another. One frequently cited example is the space–time tradeoff, where reducing a program's execution time can increase its memory consumption. Conversely, in scenarios where memory is limited, engineers might prioritize a slower algorithm to conserve space. There is rarely a single design that can excel in all situations, requiring programmers to prioritize attributes most relevant to the application at hand. Metrics... 6b0e5738d0