Advanced Compiler Design And Implementation

History of compiler construction The most common reason for transforming source code is to create an executable program. It was the world's first self-compiling compiler – the compiler In computing, a compiler is a computer program that transforms source code written in a programming language or computer language (the source language), into another computer language (the target language, often having a binary form known as object code or machine code). implemented on the prototype USQ-17 computer (called the Countess) at the laboratory 3be004774d

Copy propagation In compiler theory, copy propagation is the process of replacing the occurrences of targets of direct assignments with their values. Second edition. A direct assignment is an instruction of the form x = y, which simply assigns the value of y to x. Morgan Kaufmann. Pearson/Addison Wesley. Muchnick, Steven S. ISBN 978-0-321-48681-3. Advanced Compiler Design and Implementation. 1997 3be004774d

Steven Muchnick best known as author of the 1997 treatise on compilers, "Advanced Compiler Design and Implementation. " In 1974, Muchnick was awarded a PhD in computer Steven Stanley Muchnick (1945-2020) was a noted computer science researcher, best known as author of the 1997 treatise on compilers, "Advanced Compiler Design and Implementation. " 3be004774d

The name is a joke: "Stalin brutally... 3be004774d

Compiler-compiler computer science, a compiler-compiler or compiler generator is a programming tool that creates a parser, interpreter, or compiler from some form of formal In computer science, a compiler-compiler or compiler generator is a programming tool that creates a parser, interpreter, or compiler from some form of formal description of a

programming language and machine 3be004774d

y = x 3be004774d

Constant folding Constant folding and constant propagation are related compiler optimizations used by many modern compilers. An advanced form of constant propagation known Constant folding and constant propagation are related compiler optimizations used by many modern compilers. An advanced form of constant propagation known as sparse conditional constant propagation can more accurately propagate constants and simultaneously remove dead code 3be004774d

Source code for a parser of the programming language is returned as the parser generator's output. This source code can then be compiled... 3be004774d

Optimizing compiler Optimization is generally implemented as a sequence of optimizing transformations, a. 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 3be004774d

Any program written in a high-level programming language must be translated to object code before it can be executed, so all programmers using such a language use a compiler or an interpreter, sometimes even both. Improvements to a compiler may lead to a large number of improved features in executable programs. 3be004774d Copy propagation often makes use of reaching definitions, use-def chains and def-use chains when computing which occurrences of the target may be safely replaced. If all upwards exposed uses of the target may be safely modified, the assignment operation may be eliminated. 3be004774d The most common type of compiler-compiler is called a parser generator. It handles only

syntactic analysis. 3be004774d Copy propagation is a useful "clean up" optimization frequently used after other compiler passes have already been run. Some optimizations—such as classical implementations of elimination of common sub expressions... 3be004774d A formal description of a language is usually a grammar used as an input to a parser generator. It often resembles Backus–Naur form (BNF), extended Backus–Naur form (EBNF), or has its own syntax. Grammar files describe a syntax of a generated compiler's target programming language and actions that should be taken against its specific constructs. 3be004774d

Common subexpression elimination , they all evaluate to the same value), and analyzes whether it is worthwhile replacing them with a single variable holding the computed value. e. , they In compiler theory, common subexpression elimination (CSE) is a compiler optimization that searches for instances of identical expressions (i. e. In compiler theory, common subexpression elimination (CSE) is a compiler optimization that searches for instances of identical expressions (i 3be004774d

Related software include decompilers,... 3be004774d The compiler runs slowly, with little or no support for debugging or other niceties. Full R4RS Scheme is supported, with a few minor and rarely encountered omissions. Interfacing to external C libraries is straightforward. The compiler does lifetime analysis and hence does not generate as much garbage as might be expected, but global reclamation of storage is done using the Boehm garbage collector. 3be004774d The Production Quality Compiler-Compiler, in the late 1970s, introduced the principles of compiler organization that are still widely... 3be004774d z = 3 + x 3be004774d

Bootstrapping (compilers) An initial core version of the compiler (the bootstrap compiler) is generated in a different language (which could be assembly language); successive expanded versions of the compiler are developed using this minimal subset of the language. An initial In computer science, bootstrapping is the technique for producing a self-compiling compiler – that is, a compiler (or assembler) written in the source programming language that it intends to compile. The problem of compiling a self-compiling compiler has been called the chicken-or-egg problem in compiler

design, and bootstrapping is a solution to this problem. producing a self-compiling compiler – that is, a compiler (or assembler) written in the source programming language that it intends to compile 3be004774d

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. Optimization is a collection of heuristic methods for improving resource usage in typical programs. 3be004774d Bootstrapping is a fairly common practice when creating a programming language. Many compilers for many programming languages are bootstrapped, including compilers for ALGOL, BASIC, C, Common Lisp,... 3be004774d

Compiler g. A bootstrap compiler is often a temporary compiler, used for compiling a more permanent or better optimized compiler for a In computing, a compiler is software that translates computer code written in one programming language (the source language) into another language (the target language). cross-compiler itself runs. assembly language, object code, or machine code) to create an executable program. 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 3be004774d

There are many different types of compilers which produce output in different useful forms. A cross-compiler produces code for a different 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. 3be004774d From the following code: 3be004774d z = 3 + y 3be004774d Copy propagation would yield: 3be004774d

Stalin (Scheme implementation) Stalin is intended for production use in generating an optimized executable. It uses advanced data flow analysis and type inference and a variety of other optimization methods to produce code. Stalin (STAtic Language ImplementatioN) is a programming language, an aggressive optimizing batch whole-program Scheme compiler written by Jeffrey Mark In computing, Stalin (STAtic Language ImplementatioN) is a programming language, an aggressive optimizing batch whole-program Scheme compiler written by Jeffrey Mark Siskind 3be004774d