长沙理工大学

毕业设计 (论文) 开题报告

题目：基于 LaTeX 系统的长沙理工大学本科生毕业论文模板

课 题 类 别：设计 ☑ 论文 ☐

学 生 姓 名：**XXX**

学 号：**2018xxxxyyyy**

班 级：**软件 18-x 班**

专业 (全称)：**软件工程**

指 导 教 师：**xxx**

2022 年 3 月

一、本课题设计（研究）的目的：

制作一个长沙理工大学本科生毕业论文 LaTeX 模板

二、设计（研究）现状和发展趋势（文献综述）：

1、课题背景

（1）编译技术的历史发展

早期计算机的软件主要是用汇编语言编写的。汇编语言指令和机器语言指令之间存在着十分紧密的对应关系，因此很容易将汇编语言程序转换为可执行的机器代码。然而汇编语言仍然依赖于特定的计算机体系结构，具有不可移植、开发效率低下等问题[1]。20 世纪 50 年代早期，多数编译工作是将算术公式翻译成机器代码，且只允许简单的单目运算，数据元素的命名方式也有很多限制，然而它们奠定了对高级语言编译系统的研究和开发的基础[2]。直到 1957 年，由 IBM 公司的 FORTRAN 团队推出了第一个商用编译器[3-4]；1958 年，Bauer 等人实现了第一个 ALGOL 58 编译器[5]。自此，以 FORTRAN 语言为代表的第一批高级语言问世。

20 世纪 50 年代编译器领域刚刚起步，其研究焦点局限于从高级语言到机器码的转换以及优化程序对时间和空间的需求。随着编译技术的发展和社会对编译程序需求的不断增长，20 世纪 50 年代末有人开始研究编译程序的自动生成工具，提出并研制编译程序的编译程序。它的功能是以任一语言的词法规则、语法规则和语义解释出发，自动产生该语言的编译程序。目前很多自动生成工具已广泛使用，如词法分析程序的生成系统 LEX、语法分析程序的生成系统 YACC 等[2, 6] 。此后，该领域产生了大量的有关程序分析与转换、代码自动生成以及运行时服务等方面的新知识。同时，编译算法也被用于便利软件和硬件开发、提高应用程序性能、检测或避免软件缺陷和恶意软件等方面。编译领域与其它方向越来越多地相互交叉渗透，这些方向包括计算机体系结构、程序设计语言、形式化方法、软件工程以及计算机安全等[7] 。20 世纪 60 年代起，不断有人使用自展技术来构造编译程序。自展的主要特征是用被编译的语言来书写该语言自身的编译程序。1971 年，PASCAL 的编译程序用自展技术生成后，其影响就越来越大[2] 。在 20 世纪 60 年代末 70 年代初，面向对象和数据抽象语言刚面世时，虽然人们对其能否有效实现还心存疑虑，但已经意识到这些概念有可能极大地提高编程人员的生产率，编译器领域发明的静态和动态优化则完全打消了这些疑虑。近来，尤其

是伴随着 20 世纪 90 年代中期 Java 的出现，易于管理的运行时系统，包括垃圾回收和即时编译技术，通过根除内存泄露进一步提高了程序员的开发效率[7] 。

随着大规模集成电路、网络通讯和其它数字信息技术的迅速发展，目前嵌入式系统已经广泛地渗透到科学研究、工程设计、军事技术、各类产业和商业文化艺术等各方面，在人们日常生活中的方方面面到处是嵌入式系统设备的身影。随着国内外各种嵌入式产品的进一步开发和推广，嵌入式技术和人们的生活结合越来越紧密，人们日常的工作、学习和生活方式终将不可避免地逐步改变，嵌入式产品正在逐步形成时尚，在当今的信息社会中扮演越来越重要的角色。由于嵌入式系统资源有限，一般无法提供编译、汇编、链接等工具，同时也很难提供高级调试功能，因此嵌入式系统的应用软件一般不能直接在嵌入式系统平台上进行开发，而需要在交叉编译环境中开发[7]。

（2）程序设计语言的历史发展

程序设计语言的发展过程实质是一个不断抽象的过程。第一代程序语言（1GL）是指纯粹的机器语言，也是最早的计算机程序语言，它的抽象层级很低，而且机器语言程序以十进制或二进制的形式写在卡片或纸带上，可读性很差，也因此很容易出错；第二代程序语言（2GL）就是我们所熟知的汇编语言，汇编语言的抽象层级比机器语言要高一级，它使用助忆符编写程序来控制计算机的行为，可读性比机器语言提高了不少；编译技术的发展让程序语言的抽象能力得到了进一步的提升，第三代程序语言（3GL）不再和硬件有关，也因此从硬件的底层限制中摆脱出来，正式迈入了高级语言时代，如今第三代程序语言的发展已经经历了 60 多个年头，大大小小的程序语言已有成百上千个，各种编程范式、类型系统也随着程序语言的更迭而出现；第四代程序语言（4GL）以 SQL 为代表，在第三代程序语言的基础上进一步抽象，使得使用者可以完全无视计算机的结构，只用全心全意专注在自己要解决的问题领域，但是这一层抽象也意味着这类语言只能在自己的问题领域发挥作用，在其他领域则失去了其效率甚至完全不能发挥作用；第五代程序语言（5GL）是通过定义问题的约束来解决问题的，而不是通过定义解决这个问题的算法[8-9] 。

2、国内外研究现状

（1）编译器现状

到目前为止，编译器领域最为突出的成就是高级语言的广泛使用。从银行、企业的管理软件，到高性能计算和各种万维网应用，今天的绝大多数软件都是用高级语言编写并经过静态或动态编译的。与早期的编译器实现相比，今天的编译算法明显变得越来越复杂。早期的编译器采用简单直观的技术对程序进行词法分析，而今天的词法分析技术则基于形式语言和自动机理论，这使得编译器前端的开发更为系统化；同样，

原来有关可重构编译器工作中采用简单直观的技术进行依赖分析和循环变换，而今天，编译器的这一部分工作采用了基于整数线性规划的强大算法。众多的编译算法，包括词法分析、类型检查和推导、数据流分析、基于数据依赖性分析的循环变换、基于图着色的寄存器分配以及软件流水等，都是计算机科学中的奇葩。通过集成到各种功能强大而应用广泛的工具中，这些方法极大地影响着计算领域的实践[7] 。随着并行技术和语言的发展，并行编译技术和自动并行编译技术也进入了一个全新的发展领域[2] 。

（2）程序设计语言现状

程序设计语言作为计算机科学领域的一个重要工具，伴随着计算机科学的发展不断进步，已有几十年的历史。在这几十年的历史中，各式各样的程序设计语言不断涌现，发展出了各种编程范式和理论。近年来，随着开源社区的发展，新兴移动平台的出现，Go、Rust、Elixir、Swift 这些新的程序设计语言不断出现。这些不同的语言都各自有其特色及擅长的领域，如 Go 以 C 的继任者为目标，以高效和高并发著称，Rust 则以安全、并发、实用为准则，期望成为新的系统编程默认语言，Elixir 则继承了 Erlang 的函数式思想，Ruby 优雅的语法，支持分布式、高容错、实时应用程序的开发[8] 。

3、发展趋势

（1）编译器的发展趋势

编译器的核心是将相同的逻辑结构和思想从一种语言表示转换到另一种语言表示。编译程序可以说计算机内部处理的一个过程，对用户来说不是透明的，但是它依旧是计算机的不可缺少的组成部分，是连接用户和机器通信的桥梁，它的进步奠定了计算机发展的基石。随着现如今智能时代的来临，社会会对计算机发展需求越来越高，我们的未来会提供给计算机一个更加广阔的舞台，同时编译技术的前景也会更加广阔，能够提供给世界更大的便利[10] 。

（2）程序设计语言的发展趋势

程序语言的发展过程是一个不断抽象的过程。程序语言的发展过程是一个不断抽象的过程；结构化程序语言是对控制流过程的抽象；面向对象程序语言是对数据及其行为的抽象；函数式语言是对运算的抽象；领域特定语言是对其自身特定领域的抽象；逻辑编程语言是对算法的抽象。从第五代程序语言的尝试中可以看出来，程序语言的最终形态和人工智能理应有很大程度的重合：通过定义问题的约束和期望得到的结果，其他的过程则完全由计算机完成[8] 。

三、设计（研究）的重点与难点，拟采用的途径（研究手段）：

BO（Binary Object）语言是一种面向对象的程序设计语言。正如其名，BO 语言中的所有类型都应当是一个二进制对象，这正是本课题设计的重点。

设计难点在于对生成代码进行编译优化。编译优化要求在保证生成代码与源代码逻辑一致的前提下最大限度地提升程序的运行效率。

四、设计（研究）进度计划：

第　1　周：系统分析, 完成开题报告

第　2　周：查阅资料, 翻译文献。

第 3-4 周：功能设计，数据库设计，页面设计。

第 5-6 周：系统功能模块设计。

第 7-11 周：各功能模块程序开发。

第 9-12 周：程序调试，修改。

第 12-14 周：撰写论文。

第 14-15 周：论文修改。

第　15　周：准备答辩，制作演示软件。

第　16　周：答辩。

五、参考文献：

[1] Nanos. 汇编语言的优缺点[EB/OL]. 2007. https://blog.csdn.net/nanos/article/details/1700990.

[2] 贤人好客. 浅谈编译原理近期发展[EB/OL]. 2010. http://www.360doc.cn/article/617416_32805422.html.

[3] Mmpire. 编译器历史简要[EB/OL]. 2006. https://blog.csdn.net/mmpire/article/details/620714.

[4] Backus J W, Beeber R J, Best S, et al. The FORTRAN automatic coding system[C]//Papers presented at the February 26-28, 1957, western joint computer conference: Techniques for reliability. [S.l. : s.n.], 1957: 188-198.

[5] Kimpel P. Knuth's Algol-58 Compiler[EB/OL]. 2015. https://datatron.blogspot.com/2015/11/knuths-algol-58-compiler.html.

[6] Aho A V, Lam M S, Sethi R, et al. Compilers: Principles, Techniques, & Tools[M]. Second Edition. [S.l.]: Pearson Addison Wesley, 2006.

[7] 贤人好客. 编译器研究之路 _ 未来 50 年[EB/OL]. 2010. http://www.360doc.com/content/10/0527/21/617416_29860900.shtml.

[8] 陈一鸣. 我看程序语言的历史、现在与将来[EB/OL]. 2017. https://yiming.dev/blog/2017/06/18/programming_languages_comparison.

[9] Jeba E. Generation of Programming Languages[EB/OL]. 2020. https://medium.com/analytics-vidhya/generation-of-programming-languages-6e74aff63109.

[10] 黄平, 万芯彤. 编译程序的过程与发展[EB/OL]. 2018. http://www.chinaqking.com/wpskc/2018/1357257.html.

指导老师意见

签名：_____

月　　日

教研室（学术小组）意见

教研室主任（学术小组长）（签章）：

月　　日