

**课程设计报告**

**题目：基于SAT的对角线数独游戏求解程序**

**课程名称：程序设计综合课程设计**

**专业班级： 本硕博2301 \_**

**学 号： U202315752\_\_\_\_\_\_**

**姓 名： 陈宇航**

**指导教师： 向文**

**报告日期： 2024年9月4日**

**计算机科学与技术学院**

**任务书**

* **设计内容**

SAT问题即命题逻辑公式的可满足性问题（satisfiability problem），是计算机科学与人工智能基本问题，是一个典型的NP完全问题，可广泛应用于许多实际问题如硬件设计、安全协议验证等，具有重要理论意义与应用价值。本设计要求基于DPLL算法实现一个完备SAT求解器，对输入的CNF范式算例文件，解析并建立其内部表示；精心设计问题中变元、文字、子句、公式等有效的物理存储结构以及一定的分支变元处理策略，使求解器具有优化的执行性能；对一定规模的算例能有效求解，输出与文件保存求解结果，统计求解时间。

* **设计要求**

要求具有如下功能：

**（1）输入输出功能：**包括程序执行参数的输入，SAT算例cnf文件的读取，执行结果的输出与文件保存等。(15%)

**（2）公式解析与验证：**读取cnf算例文件，解析文件，基于一定的物理结构，建立公式的内部表示；并实现对解析正确性的验证功能，即遍历内部结构逐行输出与显示每个子句，与输入算例对比可人工判断解析功能的正确性。数据结构的设计可参考文献[1-3]。(15%)

**（3）DPLL过程：**基于DPLL算法框架，实现SAT算例的求解。(35%)

**（4）时间性能的测量：**基于相应的时间处理函数（参考time.h），记录DPLL过程执行时间（以毫秒为单位），并作为输出信息的一部分。(5%)

**（5）程序优化：**对基本DPLL的实现进行存储结构、分支变元选取策略[1-3]等某一方面进行优化设计与实现，提供较明确的性能优化率结果。优化率的计算公式为：[(t-to)/t]\*100%,其中t 为未对DPLL优化时求解基准算例的执行时间，to则为优化DPLL实现时求解同一算例的执行时间。(15%)

**（6）SAT应用：**将数双独游戏[5]问题转化为SAT问题[6-8]，并集成到上面的求解器进行数独游戏求解，游戏可玩，具有一定的/简单的交互性。应用问题归约为SAT问题的具体方法可参考文献[3]与[6-8]。(15%)

* **参考文献**

[1] 张健著. 逻辑公式的可满足性判定—方法、工具及应用. 科学出版社，2000

[2] Tanbir Ahmed. An Implementation of the DPLL Algorithm. Master thesis, Concordia University,Canada,2009

[3] 陈稳. 基于DPLL的SAT算法的研究与应用.硕士学位论文，电子科技大学，2011

[4] Carsten Sinz. Visualizing SAT Instances and Runs of the DPLL Algorithm. J Autom Reasoning (2007) 39:219–243

[5] 360百科：数独游戏<https://baike.so.com/doc/3390505-3569059.html>

Twodoku： https://en.grandgames.net/multisudoku/twodoku

[6] Tjark Weber. A sat-based sudoku solver. In 12th International Conference on Logic for Programming, Artificial Intelligence and Reasoning, LPAR 2005, pages 11–15, 2005.

[7] Ins Lynce and Jol Ouaknine. Sudoku as a sat problem. In Proceedings of the 9th International Symposium on Artificial Intelligence and Mathematics, AIMATH 2006, Fort Lauderdale. Springer, 2006.

[8] Uwe Pfeiffer, Tomas Karnagel and Guido Scheffler. A Sudoku-Solver for Large Puzzles using SAT. LPAR-17-short (EPiC Series, vol. 13), 52–57

[9] Sudoku Puzzles Generating: from Easy to Evil.

http://zhangroup.aporc.org/images/files/Paper\_3485.pdf

[10] 薛源海，蒋彪彬，李永卓. 基于“挖洞”思想的数独游戏生成算法. 数学的实践与认识,2009,39(21):1-7

[11] 黄祖贤. 数独游戏的问题生成及求解算法优化. 安徽工业大学学报(自然科学版), 2015,32(2):187-191

**目录**

**任务书** I

**1引言** 1

1.1×××××× 1

1.2×××××× 3

1.2.1×××××× 7

1.3×××××× 10

**……**

**……**

**……**

3**××××××** 20

3.1×××××× 20

3.2×××××× 23

3.2.1×××××× 25

3.3×××××× 30

**……**

**……**

**4** 40

**参考文献** 44

**附录□××××××** 45

（章为宋体小4号加粗，其余宋体小4号，字母、阿拉伯数字为Time New Roman小4号）

**1引言**（黑体小2加粗居中）

**1.1课题背景与意义**

计算机科学与技术、大数据与智能数据专业大二学生，在前两个学期已经学习了C语言程序设计、数据结构两门面向编程知识与技术的基础理论课，以及C语言程序设计实验、数据结构实验两门编程实践课程。学生不仅具有较为系统性的C语言、常用数据结构基本知识，而且具有初步的程序设计、数据抽象与建模、问题求解与算法设计的能力，奠定了进行复杂程序设计的知识基础。但两门实验课仍属于对基本编程模型与技术的验证性训练，而“程序设计”综合课程设计正是使大家从简单验证到综合应用，甚至在编程中实现智慧与风格升华的重要实践环节，为后续学习与进行计算机系统编程打下坚实的基础，让综合编程技能成为大家的固有能力与通向未来专业之门的钥匙。

命题逻辑公式的可满足性问题（SAT）是数理逻辑、计算机科学、集成电路

设计与验证和人工智能等领域中的核心问题，并且是第一个被证明出来的 NP问题。 SAT问题在计算复杂性理论中具有非常重要的地位，设计并实现解决该类问题的高效算法意义重大

**1.2国内外研究现状**（黑体4号加粗,字母、阿拉伯数字为Time New Roman小4号加粗）

自 1960 年以来，SAT 问题一直受到全球的广泛关注。世界各地的研究人员为此付出了大量的努力，提出了多种解决算法。每年在可满足性理论和应用方面的国际会议上都会举办一场 SAT 竞赛，旨在寻找一组最快的 SAT 求解器，同时还会详细展示一系列高效求解器的性能。在 2003 年的 SAT 竞赛中，超过三十种的解决方案针对从成千上万的基准问题中挑选出的 SAT 问题实例进行同台竞技。与此同时，国内也经常组织 SAT 竞赛和研讨会，这些活动极大地推动了 SAT 算法的发展。

DPLL 算法被认为是最经典的完备算法，它最早由 Davis 和 Putnam 等人于

1960 年提出。后续的完备算法大多是在 DPLL 算法的基础上进行改进的。然而，

由于 SAT 问题的最坏情况时间复杂度是指数级别，一度使得许多研究者望而却步。 直到 1971 年，S.A.Cook 证明了 SAT 问题是 NP 完全问题，这才重新点燃了研究兴趣。从此，SAT 问题吸引了越来越多的研究者，引发了一系列高效的 SAT 算法的诞生，如 MINISAT、SATO、CHAFF、POSIT 和 GRASP 等。这些算法的发展主要基

于对 DPLL 算法的改进，包括新的数据结构、变量决策策略和快速算法实现方案

等

**1.3课程设计的主要研究工作**（黑体4号加粗,字母、阿拉伯数字为Time New Roman小4号加粗）

××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××××

--------章与章之间插入分页符----------

**2系统需求分析与总体设计**

（黑体小2加粗居中,字母、阿拉伯数字为Time New Roman小2号加粗）

**2.1系统需求分析**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分应该写的是用户需求，明确你做的系统要实现的目标，能处理一些什么样的事务、事务处理流程等。

**2.2系统总体设计**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分可根据用户需求，设计和规划一个系统，说明清楚系统应该有哪些功能模块，每个模块做什么。最后给出完整的系统模块结构图。

以下是文中有图标的形式，要求对图标引用说明。

××××××××××××××××××××××××××××××××××××××××××××××××，其×××××可表示如下：

(3-1）

(3-2）

×××××××××××××××××××××××××××× (如表3-1所示)

表3-1 ××××××××××

|  |  |  |  |
| --- | --- | --- | --- |
| ××××× | ××× | ××× | ××× |
| ××××× | ×× | ×× | ×× |
| ××××× | ×× | ×× | ×× |
| ××××× | ×× | ×× | ×× |
| ××××× | ×× | ×× | ×× |

（表标题：位于表格上方，黑体小4号，字母、阿拉伯数字为Time New Roman小4号，表内容：宋体5号，字母、阿拉伯数字为Time New Roman 5号）

××××××××××××××××××××××××××× (如图3-1所示)



图3-1 ××××××××××

（图标题：位于图下方，黑体小4号，字母、阿拉伯数字为Time New Roman小4号）

.......

--------章与章之间插入分页符----------

**3系统详细设计**

（黑体小2加粗居中,字母、阿拉伯数字为Time New Roman小2号加粗）

**3.1有关数据结构的定义**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分要写的：（1）首先描述系统中要处理那些数据，每种类型的数据包括哪些数据项，每个数据项的数据类型，最后可用一个表格表示出来；（2) 描述这多种数据在系统中如何关联，可通过图直观的说明这多种数据间的关联。

**3.2 主要算法设计**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分主要描述系统中模块实现的流程，可采用文字配合流程图的方式表示各模块的算法思想及流程。

**4系统实现与测试**

（黑体小2加粗居中,字母、阿拉伯数字为Time New Roman小2号加粗）

**4.1系统实现**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分可首先叙述一下你的系统实现的软硬件环境；

根据3.1的设计，用C语言定义各种数据类型；

程序代码部分在这里不需要给出来，只需要叙述清楚在系统中包括哪些函数，各函数的说明，如何利用这些函数实现系统各模块的功能，以及函数间的调用关系（可用图表示出来）。程序详见附录。

**4.2系统测试**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

首先叙述一下常用的软件测试方法，在选择几个主要的功能模块（自行掌握数量，关键要体现你的水平的一些模块）描述测试过程。（1）先明确模块的功能、设计目标等；（2）分析、叙述如何选取测试数据，要求有完整的测试大纲；（3）运行结果（这时可用截图）；（4）分析运行结果、确认程序满足该模块的设计目标。

**5□总结与展望**

（黑体小2加粗居中,字母、阿拉伯数字为Time New Roman小2号加粗）

**5.1全文总结**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

对自己的工作做个总结，主要工作如下：

（1）

（2）

（3）

**5.1工作展望**（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

在今后的研究中，围绕着如下几个方面开展工作。。。。。。。。

（1）

（2）

# 6 体会

这部分就自由发挥了，写具体的项目设计与实现中的经验。

**参考文献**

将报告中引用的参考文献放在此处。

[1]

[2]

[3]

[4]

[5]

……

……

……

**附录**

关键源代码可放在目录中。

……

……

……