Curriculum Design

Yuzhe SHI

Feburary 2020

摘要

这是课程设计。

目录

第一章　任务书

1.1　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。

1.2　设计要求

1.2.1　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%)

1.2.2　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%)

1.2.3　DPLL 过程

基于 DPLL 算法框架，实现 SAT 算例的求解。(35%)

1.2.4　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%)

1.2.5 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:

$$\frac{t - t_0}{t_0} \times 100\% \tag{1.1}$$

其中 t 为未对 DPLL 优化时求解基准算例的执行时间，t_o 则为优化 DPLL 实现时求解同一算例的执行时间。(15%)

1.2.6 SAT 应用

将二进制数独游戏问题转化为 SAT 问题，并集成到上面的求解器进行问题求解，游戏可玩，具有一定的简单的交互性。

第二章　引言

2.1　背景与研究意义

SAT 问题又称命题逻辑公式的可满足性问题（*satisfiability problem*），是判断对合取范式形式给出的命题逻辑公式是否存在一个真值指派使得该逻辑公式为真。SAT 问题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。看似简单，却可广泛应用于许多实际问题如人工智能、电子设计自动化、自动化推理、硬件设计、安全协议验证等，具有重要理论意义与应用价值。对于 SAT 问题的研究从没有停止过，在 1997 年和 2003 年，H.Kautz 与 B.Selman 两次列举出 SAT 搜索面临的挑战性问题，并于 2011 年和 2007 年，两度对当时的 SAT 问题研究现状进行了全面的综述。黄文奇提出的 Solar 算法在北京第三届 SAT 问题快速算法比赛中获得第一名。对 SAT 问题的求解主要有完备算法和不完备算法两大类。不完备算法主要是局部搜索算法，这种算法不能保证一定找到解，但是求解速度快，对于某些 SAT 问题的求解，局部搜索算法要比很多完备算法更有效。完备算法出现的时间更早，优点是可以正确判断 SAT 问题的可满足性，在算例无解的情况下可以给出完备的证明。对于求解 SAT 问题的优化算法主要有启发式算法、冲突子句学习算法、双文字监视法等。SAT 问题是第一个被证明的 NP 完全问题，而 NP 完全问题由于其极大的理论价值和困难程度，破解后将会在许多领域得到广泛应用，从而在计算复杂性理论中具有非常重要的地位。由于所有的 NP 完全问题都能够在多项式时间内进行转换，那么如果 SAT 问题能够得到高效解决，所有的 NP 完全问题都能够在多项式时间内得到解决。对 SAT 问题的求解，可用于解决计算机和人工智能领域内的 CSP 问题（约束满足问题）、语义信息的处理和逻辑编程等问题，也可用于解决计算机辅助设计领域中的任务规划与设计、三维物体识别等问题。SAT 问题的应用领域非常广泛，还能用于解决数学研究和应用领域中的旅行商问题和逻辑算数问题。许多实际问题，例如数据库检索、积木世界规划、超大规模集成电路设计、人工智能等都可以转换成 SAT 问题进而进行求解。可见对 SAT 问题求解的研究，具有重大意义。

2.2　国内外研究现状

2.3　课程设计的主要研究工作

第三章　系统需求分析与总体设计

3.1　系统需求分析

3.2　系统总体设计

```
#include<stdio.h>
int main()
```

第四章　系统详细设计

4.1　有关数据结构的定义

4.2　主要算法设计

第五章　系统实现与测试

5.1　系统实现

5.2　系统测试

第六章　结论

6.1　讨论与总结

6.2　未来展望

第七章　心得体会