作业 1 - SAT

问题 1（简单）：迷宫问题（获得结业证书必做题目 1）

用 SAT 的方法, 求在给定的 $n \times n$ 的迷宫中, 求以 $(1,1)$ 为起点,(n,n) 为终点的一条简单路 (不含环).

1) 请给出编码的方法.

2) 通过利用开源 SAT 求解器, 给出附件例子的结果, 编码源代码和相关的 CNF 文件.

问题 2（困难）：二选一

2.1 编码: SAT 应用

现实生活中有很多 SAT 的应用问题, 例如电子设计自动化 (EDA) 问题, 数学定理证明, 组合优化, AI Planning 问题, 神经网络可解性等.

1) 请描述一个应用场景 (不限于以上问题).

2) 给出该场景中如何应用 SAT 和编码的具体方案.

2.2 求解器: SAT Solver 升级

1) 阅读并了解 CDCL 求解器 microsat 的代码 (地址 https://github.com/marijnheule/microsat).

2) 对 microsat 中的组件 (包括但不限于 branching[Biere and Fröhlich2015b, Liang *et al.*2016], restart[Biere and Fröhlich2015a], 学习子句管理 [Oh2015] 等) 进行升级. 请提交修改后的说明文档和源代码.

测试说明：测试用例节选自 SAT Competition [1]. 每个例子运行时间为 2000 秒, 打分依据的标准为 PAR2 （带惩罚的平均时间：对于时间限制内不能求解的例子，其时间按两倍时间限制，也就是 2*2000=4000 秒计算）.

参考文献

[Biere and Fröhlich2015a] Armin Biere and Andreas Fröhlich. Evaluating cdcl restart schemes. *Proceedings of Pragmatics of SAT*, pages 1–17, 2015.

[1] http://www.satcompetition.org/

[Biere and Fröhlich2015b] Armin Biere and Andreas Fröhlich. Evaluating cdcl variable scoring schemes. In *International conference on theory and applications of satisfiability testing*, pages 405–422. Springer, 2015.

[Liang *et al.*2016] Jia Hui Liang, Vijay Ganesh, Pascal Poupart, and Krzysztof Czarnecki. Learning rate based branching heuristic for sat solvers. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 123–140. Springer, 2016.

[Oh2015] Chanseok Oh. Between sat and unsat: the fundamental difference in cdcl sat. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 307–323. Springer, 2015.

附件 1: 问题 1 例子

例子格式说明:

第一行一个数 n，表示迷宫的大小为 $n \times n$。

接下来一个 $n \times n$ 的矩阵，0 代表无障碍，1 代表有障碍。

例子 1:

```
10
0111100110
0001101010
1000000000
1001110001
0000001100
1111101100
0010000010
0001010110
0110001110
0100110010
```

例子 2:

```
20
01100100001000100100
00000010111101000001
01000111001000000001
00100010011101010001
01100001001111000101
10000001111000000100
00010000001100100101
01100101001000001111
00100101100010111111
01100010000000000001
10011011000101101111
01101100110000100001
00000011111110010100
01111101110000100101
01010000100011000111
11000100101101010000
01100001110000000100
00001011110011000010
01111011110111010100
10000111110110001100
```