Sudoku Game

计 64　翁家翌　2016011446

2017.9

1　软件用途

本软件是一个数独小游戏，使用 Qt5 编写，实现了数独游戏的快速生成、以及提供任意题目的解决方案。

2　运行方式

安装 Qtcreater 之后，将源代码拷贝至本机，源代码位于https://git.thusaac.org/trinkle/sudoku-qt5。运行 Qtcreater 直接编译即可。

Ubuntu 下需要额外安装软件包 qtmultimedia5-dev，进入目录之后运行 qmake -makefile;make; 即可得到可执行文件 sudoku。

3　功能介绍

3.1　使用帮助

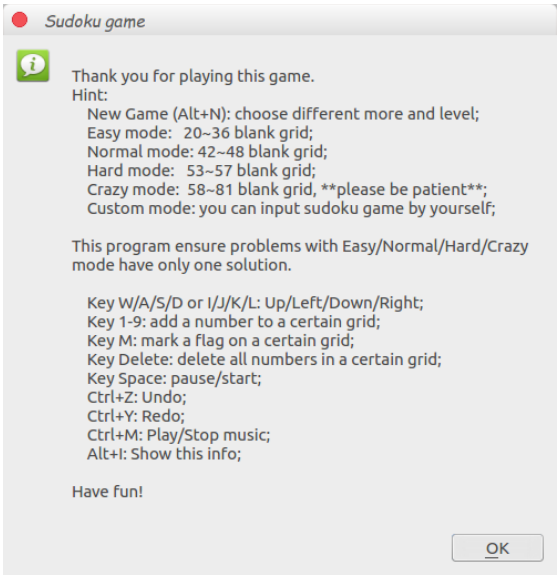

图 1: 提示信息

图 1 显示了关于本软件的使用帮助。它是运行软件时的最初界面，点击 OK 即可进入主游戏界面。

3.2 游戏界面

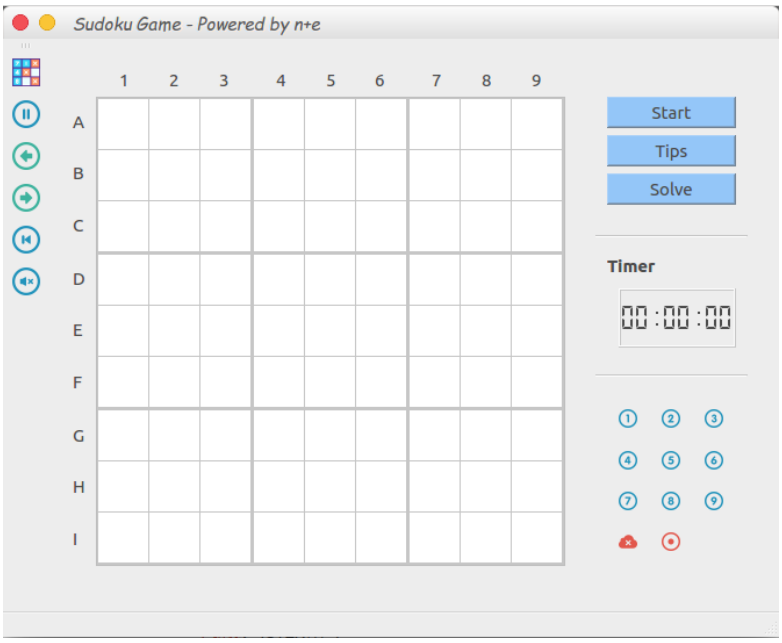

图 2: 开始界面

图 2 显示了软件的开始游戏界面。最上方为选项栏，左侧是菜单栏，中部是数独界面，右上侧是功能栏，右中部是计时器，右下侧是数字栏。

3.3 选项栏

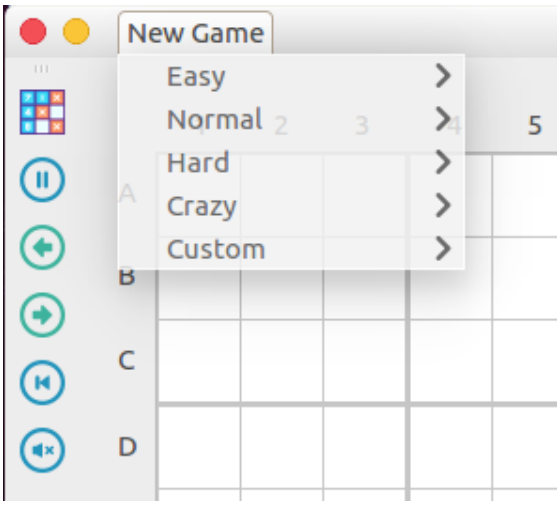

图 3: 标题选项栏

图 3 显示了软件的标题选项栏，**有四种难度可供选择，每种难度下有内置 4 个关卡，并且还有按照该难度随机生成题目的选项。**

最后一个选项 Custom 提供了输入题目的功能，该软件能够给出用户提供题目的解答。

3.4 菜单栏

图 4: 菜单栏

图 4 显示了软件的左侧菜单栏，从上到下依次为：显示信息（快捷键：Alt+I）、**暂停/继续游戏**（快捷键：空格）、**撤销**（快捷键：Ctrl+Z）、**重做**（快捷键：Ctrl+Y）、回到初始题目状态、**音效开关**（快捷键：Ctrl+M）。

3.5 功能栏

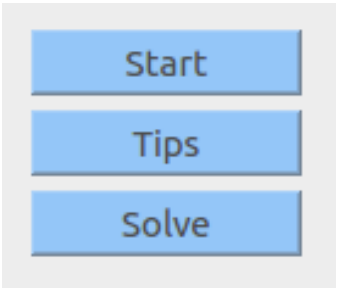

图 5: 功能栏

图 5 显示了软件的右上侧功能栏，从上到下依次为：开始游戏、**显示帮助**、**显示计算机给出的解答**。

显示帮助时，软件会显示一个还未被正确填上数字的格子的答案；显示帮助和解答时，均不会覆盖用户记录，下一步操作会回到点击按钮之前的状态。

3.6 数字栏

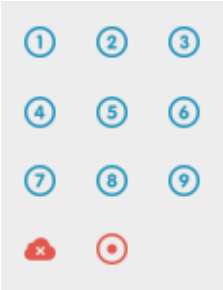

图 6: 数字栏

图 6 显示了软件的数字栏，点击 1～9（或者按下数字 1～9 按键）即可在当前选中的格子上标记/去除一个点击的数字。第四行第一个图标为**清除一个格子内的所有数字**（快捷键：`Delete`），第二个图标为**标记/取消标记一个选中的格子**（快捷键：`M`）。

3.7 数独界面

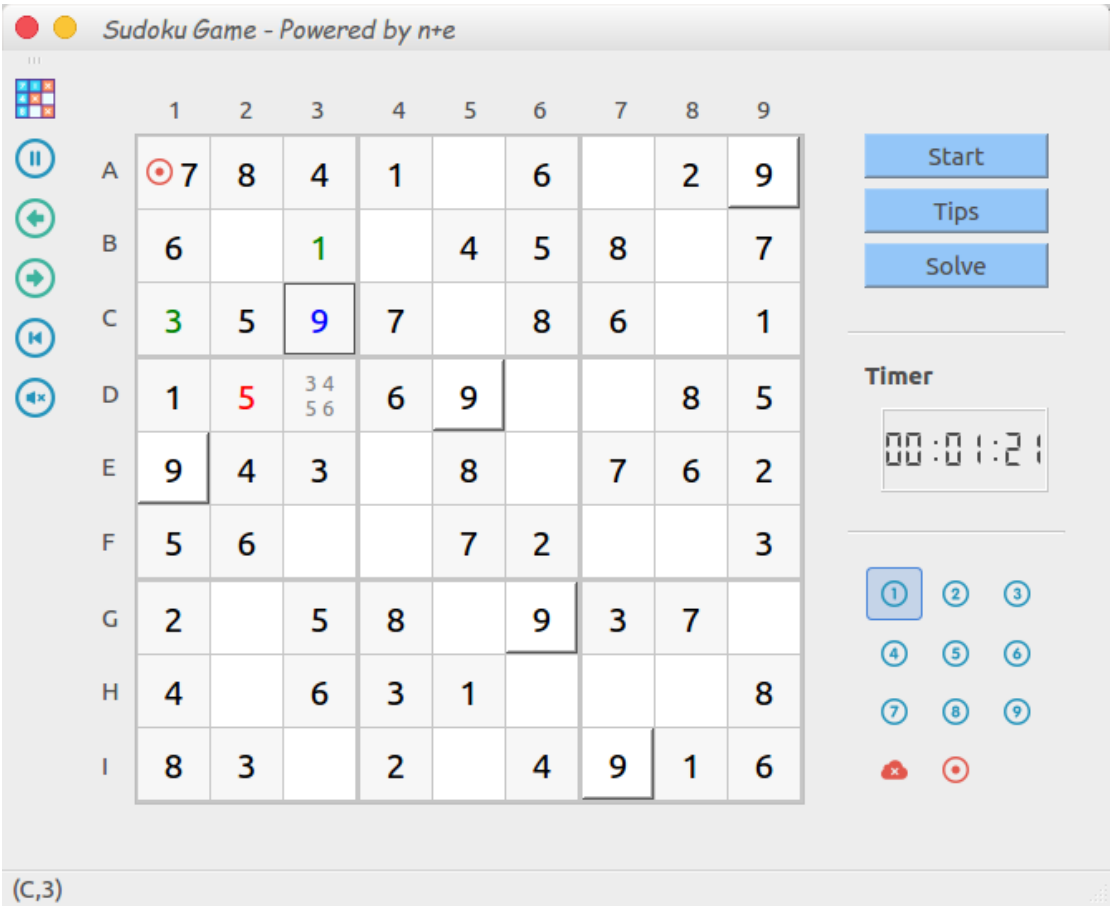

图 7: 运行界面

图 7 显示了软件在游戏过程中的运行界面。在每个时刻，均有一个格子被选中，用黑框蓝色字体高亮显示。当选中的格子已经填上一个数字，那么所有相同数字的格子均会**立体高亮**显示；否则如果选中的是一个空白格子，那么与这个格子同行同列的所有格子都会被**立体高亮**显示。当前选中格子的坐标在左下角会给予显示。

如果一个格子是题目数据，那么它会被标记为黑色字体；如果一个格子里有多个数字，那么程序将会以灰色小字号在格子内显示这些数字；如果一个填写上去的数字和已有数字**发生冲突**，则**标记为红色字体**；否则标记为绿色字体。

图 8 显示了软件在用户完成解题时的界面。

图 9 显示了软件在 Custom 模式中按下 `Solve` 按钮之后的自动解题情况，黑色为用户输入数据，**左下角状态栏会显示是否多解/单解/无解，以及计算机解题的用时**。

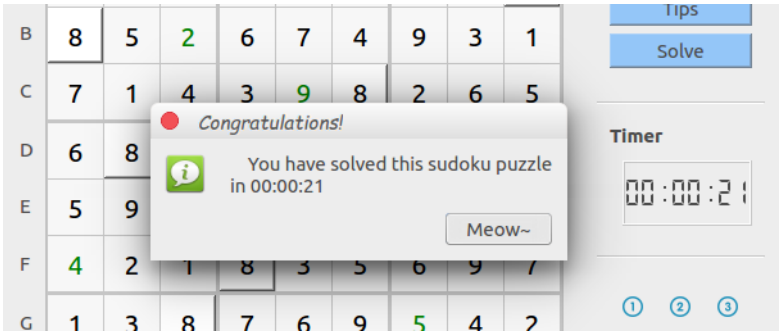

图 8: 完成解题

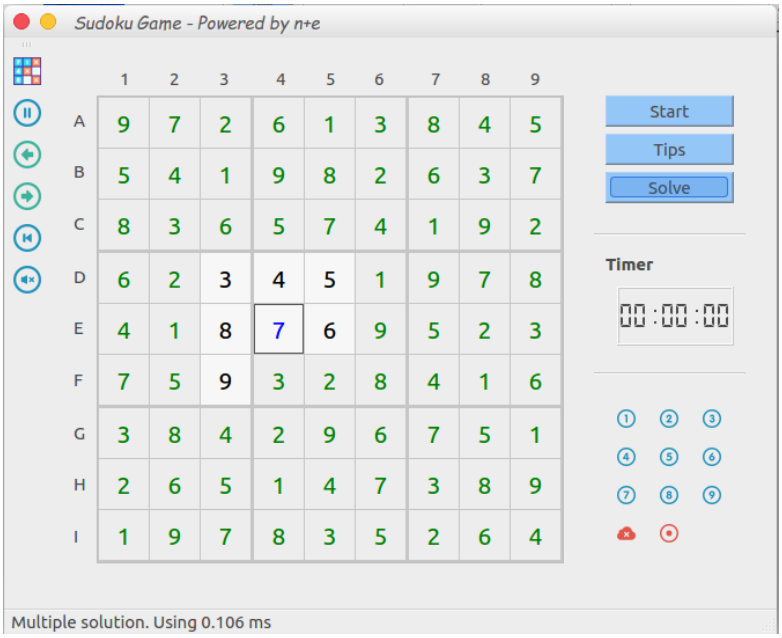

图 9: 自动解题

图 10: 有关数独算法的一个 PDF

4 数独算法实现

高中的时候仔细研究了一下数独的解法，发现 Dancing Link 完全可以被普通的搜索所取代。基本的思路是预处理出来一个搜索序列，然后用二进制表示的状态直接进行搜索。预处理出来的序列整体上保证是限制数目从小到大的，这样的话搜索树就会长得非常瘦，也更容易寻找到解。在 [NOIP2009] 靶型数独一题中，这个方法比 Dancing Link 快一倍以上。并且本机测试，10000 组数独题目，每组平均 56.5 空格，总计算时间为 0.82 秒。

有关这个算法的详细阐述，详见图 10。

5 Qt 界面实现

界面设计的一个原则是，看起来要很清楚，感觉舒服，不要太多颜色花花绿绿的，感觉很浮夸，就像那种垃圾应用一样，玩着玩着给你开个广告。我尽量避免了给使用者留下这样的印象。

我使用了 QTableWidget 创造出一个表单，再将 81 个 QPushButton 嵌到表单里面。在设计的时候，我将所有的 QpushButton 的 setFlat 设置为 True。这样的话扁平效果就能够体现。一个附带的效果就是，setFlat 在 True 和 False 之间切换时，由于表单上的网格线，会使整个界面产生 3D 立体效果（误打误撞 (x)。

有一个问题困扰了我很久：QPushButton 在设置成扁平化之后，在选中的时候会出现蓝色的选中框，并且会影响上下左右键的原始操作。解决方案是将**所有的** QPushButton 的 setFocusPolicy 改成 Qt::NoFocus 即可。

在每次用户产生鼠标/键盘事件时，由于无需考虑界面的实现效率，因此为了方便，每个事件产生之后都会重新刷新一遍整个界面；并且一个好处就是，可以在刷新的时候顺便更新历史版本记录。

界面设计的另一个原则就是用户友好,每种操作既可以用鼠标也可以用键盘,在撤销/恢复/显示提示/显示解答/回到最初的时候都不会丢失之前的数据，所有历史数据都存储在一个 QVector 中，每次要撤销/的时候直接移动指针，不会将原来已经在 QVector 中的数据删除，新一步的操作会 append 到 QVector 中，此时再使用撤销操作会回到上一次撤销前的状态。

在 Qt 程序中，我保存了四个 9×9 的数组，分别为：题目，解答，用户数字数据和用户标记数据。

代码很丑啊我只把数独算法和 ui 隔离开了，一点也不 OO 是吧但是写得爽 233。