Chess Game

计 86　周恩贤　2018011438

2019 年 8 月 30 日

1　程序介绍

本程序是基于 Qt 的国际象棋联机对战软件，允许使用者输入 IP 地址建立或加入连线，进而开启一局刺激的比赛。同时，本软件支持使用者连线后随时投降、读入棋局或储存残局，并判断比赛结束的条件，在结束时给出弹窗提示。可直接运行 *chess.exe*，或用 Qt Creater 编译运行 *chess.pro*。

2　界面介绍

2.1　游戏界面

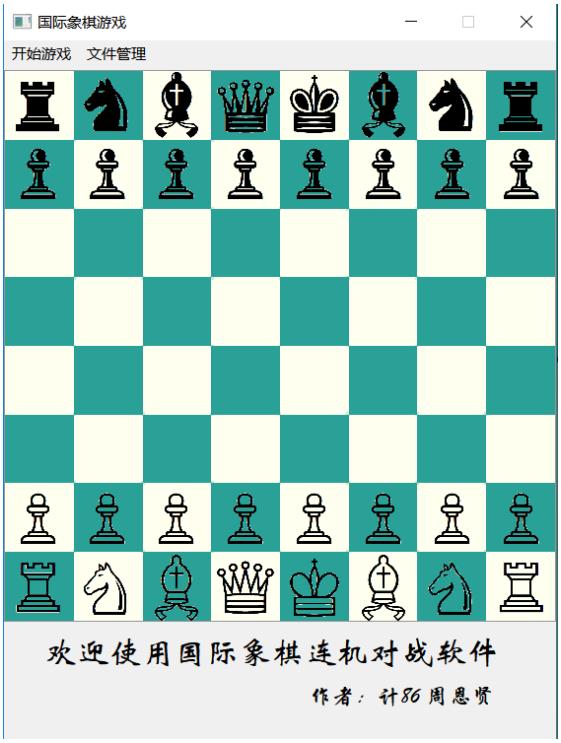

图 1: 游戏界面

图 1 是本程序的开始界面。在未联机时会出现欢迎语与作者。点击棋盘不会触发任何事件，用户只能透过菜单栏（开始游戏 | 建立连接、加入房间）选择建立或加入连接。在未联机时菜单栏（文件管理 | 读入棋局、储存棋局）都设为不可点击。

2.1.1 建立连接（Server）

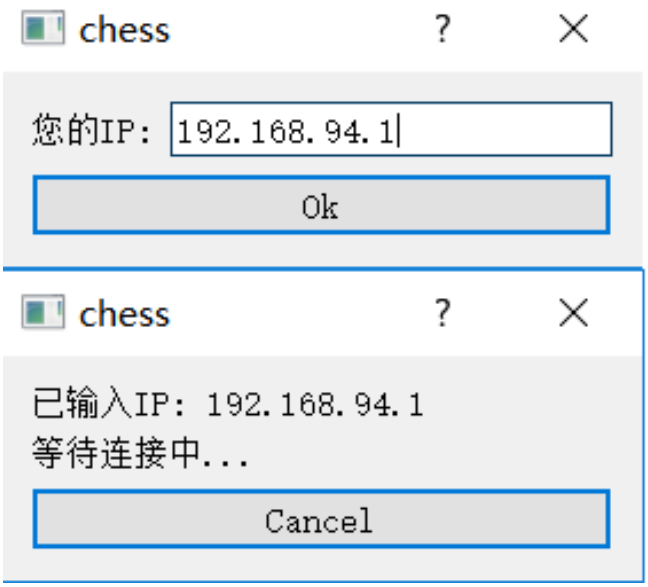

图 2: 服务器端建立连接

图 2 的上半部是服务器端按下菜单栏（开启游戏 | 建立连接）后的界面，会显示出用户当前的 IP 地址。点击 ok 按钮后会调用 QTCPServer::listen(QHostAddress::any, 8888) 开始监听任何到来的连接请求，并转为下半部的等待界面。在等待期间，用户可以随时点击 Cancel 进行取消。

2.1.2 加入连接（Client）

图 3: 用户端加入连接

图 3 为客户端按下菜单栏（开启游戏 | 加入房间）后的界面，可允许使用者输入主机 IP（这里利用一个 QSignalMapper 建立从十个数字按钮到文本框的信号传递）。当使用者按下连接钮时会进行 IP 检查并调用 connectToHost(QHostAddress(IP),8888) 发出连接请求，若 IP 格式错误、无法连接等会出现图中右方的报错提示。

2.1.3 连接成功

图 4 为两端连接成功后的界面，会弹窗告知两端连接成功以及颜色（默认 Server 为白方、Client 为黑方）。同时，界面下半部会改变，计时器开始倒数（默认每一步时限 20 秒），代表游戏正式开始！（本文档中，左方都为 Server 白方，右方皆为 Client 黑方）

图 4: 连接成功

3 基础功能介绍

3.1 移动棋子

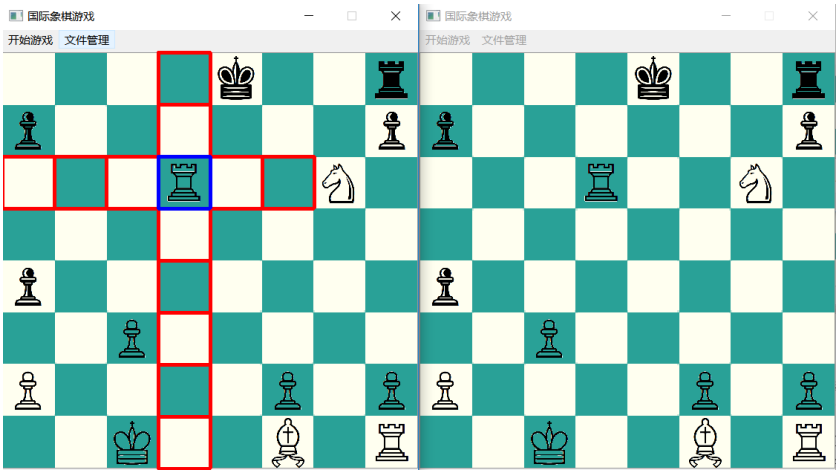

图 5: 白方回合，白方点击棋子的高亮提示

图 5 为用户点击棋子时的变化。当**轮到该用户的回合**且**点击到该用户颜色的棋子**时，被选中的棋子以蓝色高亮提示，可移动的地方以红色高亮提示。当用户再次点击该棋子会取消高亮，点击其他己方棋子会重置提示。若用户点击红色格子则棋子会移动，并调用 `QTCPSocket::write()` 传送数据以同步棋局，同时轮到另一方玩家的回合，计时器重置。

3.1.1 兵升变

当兵走到对方底线时，必须升变为非王非兵以外的棋子，示意图请见章节 9.1

3.2 胜负判定

不考虑和局的情况下，有三种情况下会造成比赛结束。

3.2.1 王被吃掉

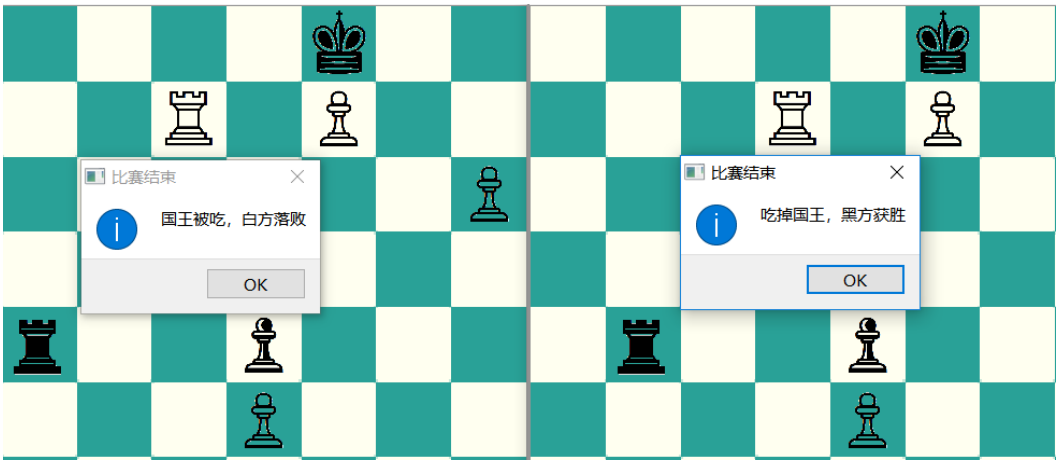

图 6: 王被吃掉

图 6 为白方国王被吃掉（允许送吃）后判负情形，会弹窗提示。

3.2.2 投降

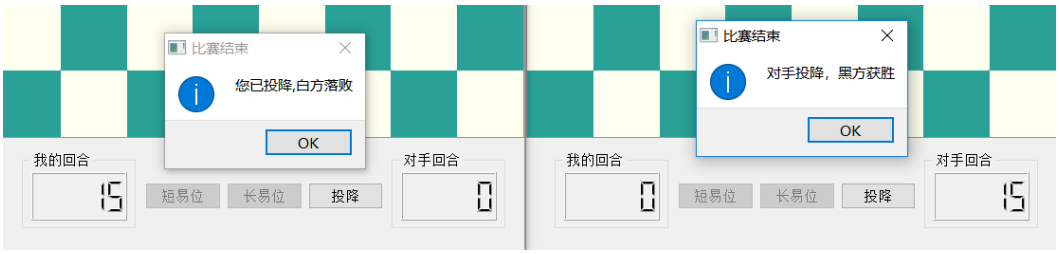

图 7: 投降后的弹窗

图 7 为白方按下投降键后判胜负的情形，会弹窗提示。

3.2.3 超时

图 8: 超时判定的弹窗

图 8 为黑方未在时限内移动后判负的情形，会弹窗提示。

4 进阶功能介绍

4.1 逼和判定

国际象棋中有一个特殊的规定--逼和，当且仅当以下二个条件成立：

- 当前玩家国王未被攻击
- 当前玩家没有任何可以合法移动的棋子。也就是说，任何移动都会导致国王落入敌方攻击范围。

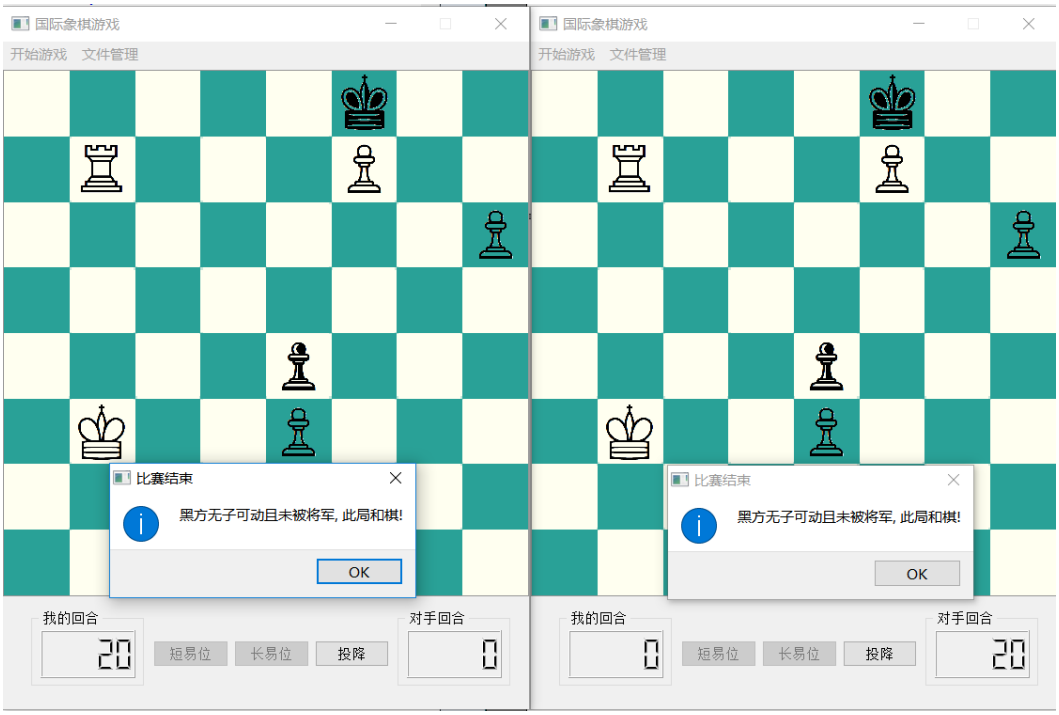

图 9: 黑方王未被将军且没有合法移动的棋子，此局和局

图 9 为逼和的局面，会弹窗提示。具体实现方式请见算法 1。

4.2 王车易位（国王入堡）

正常情况下，用户每回合只能移动一个棋子，但国际象棋有个特殊的步法--王车易位：让国王朝着车走两格，并把车移到国王的后方，当且仅当以下条件成立方可使用：

- 国王在 e 列，车在 a 列（长易位）或 h 列（短易位），且王车在同一行
- 王与车之前没有任何棋子阻隔
- 王当前未被攻击
- 王所经过且到达的位置未被攻击
- 王车未曾移动过（本次作业不要求）

图 10 为黑方回合且满足短易位的示意图（注意到因为 c8 被攻击，故只能短易位不能长易位，黑方短易位按钮变为可点击）。每次移动后，若为该方回合且符合前四条王车易位的条件，底下按钮就会转变为可点击，当用户点击后就会进行王车易位的操作。

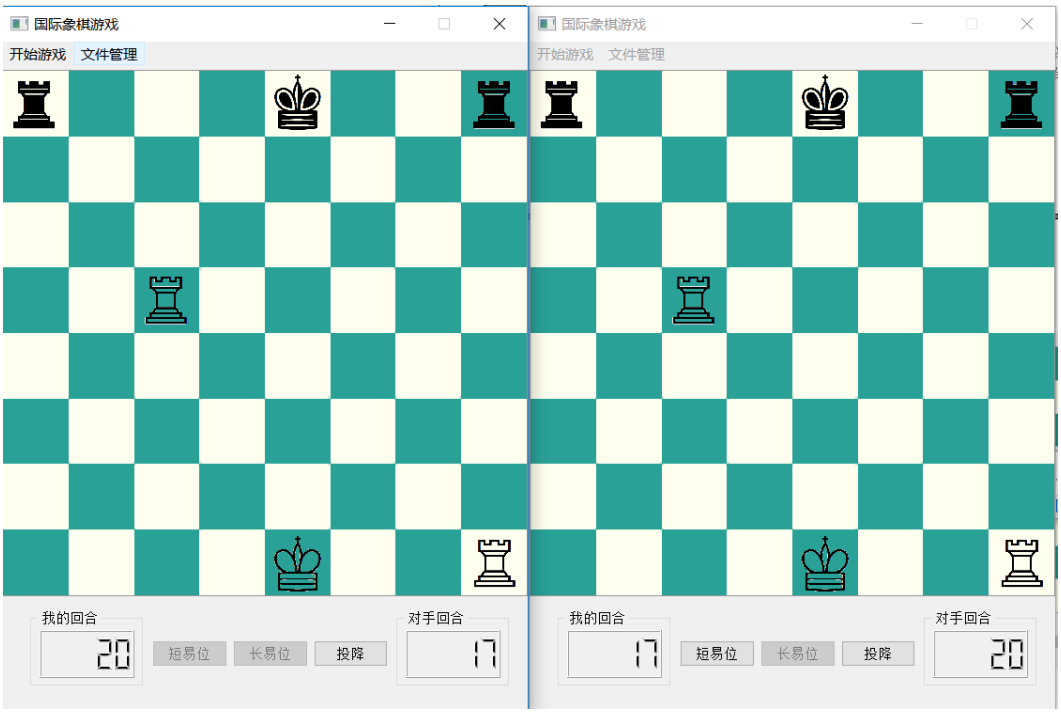

图 10: 触发王车易位条件时，按钮变为可点击

5 文件管理

用户可在连机后任何时间点进行读局/存局操作。

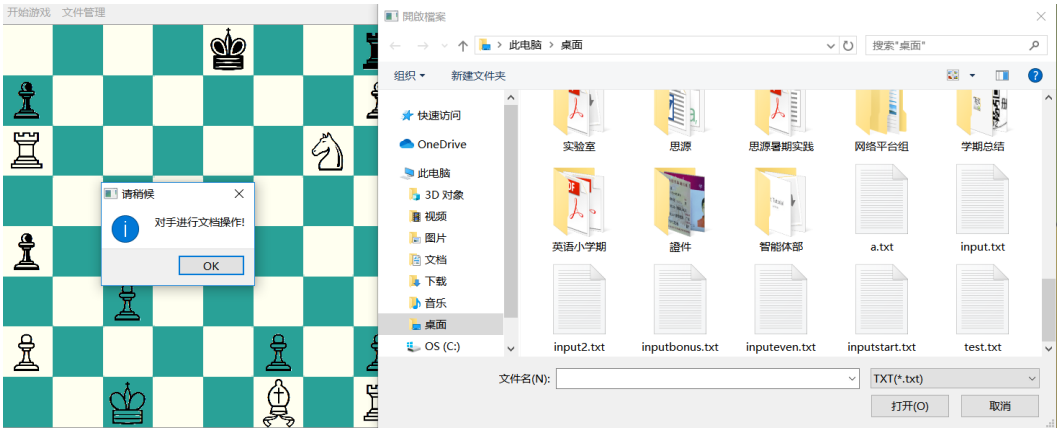

图 11: 黑方进行读入残局操作

图 11 为黑方进行读局操作的示意图，此时另一方会**弹窗给出对手正进行文件操作的提示**，本方则用 `QFileDialog` 选择文挡路径（加入了过滤器让用户只能选择 `TXT(*.txt)` 的文件）。若操作失败会报错，操作成功后双方都会跳出弹窗并提示使用者继续游戏。

6 Qt 界面实现

依照不同功能，调用了许多 `Qt` 库与函数方法：

- 绘出棋盘、棋子：写于 `PaintEvent()`

- 鼠标点击触发移动棋子：写于 `mousePressEvent()`
- 计时器：`QTimer`
- 菜单栏：实现接收 `Action` 触发信号的槽函数
- 按钮是否可触发：`setEnable()` 函数
- 残局读取：`QFileDialog`
- 弹窗警戒：`QMessageBox`
- 网络通信：`<QtNetwork>`

有了上周学习基础，这周写起界面来轻松了许多！

7 网络通信实现

通信是基于 TCP/IP 协议，我仿照了第四讲 chatserver 以及 chatclient 的源代码成功进行连接。接下来使用以下方式进行传输数据：

- 移动：传递 "m, x_1, y_1, x_2, y_2"，表示移动 (x_1, y_1) 的棋子到了 (x_2, y_2) 这一格。若升变，则多传一格整数 '$type$' 表示升变后的棋子种类
- 投降：传递 's'
- 计时器：每秒传递 '$-$'，更新时间
- 文件操作：开始时传递 'f'，结束时传递 'o'
- 读档后更新棋盘：传递 "$l, turn, ...$" 第二个数字 $turn$ 表示先手的玩家颜色，随后 $4n$ 个整数 $(x_i, y_i, col_i, type_i)$ 表示第 i 个棋子位于坐标 (x_i, y_i) 以及其颜色与种类
- 王车易位：传递 "b, i"，若 $i = 1$ 为短易位，$i = 2$ 为长易位

不过，在处理通信时也遇到了一些问题，详见章节 9.2。

8 规则与算法实现

8.1 正确走子

依照所选子的种类，进行不同方式搜索：

- 兵：直接搜前方是否空白、斜前是否是敌方子。若未走过再搜前两格的位置是否空白
- 车：上、下、左、右四个 while 回圈，当走到边界或被其他棋子挡到时跳出回圈
- 主教：与车类似，改为斜的
- 皇后：车主教的判断方式
- 马:利用两个位移向量记录:`vector<int> dx = {-2,-1,-2,-1,2,1,2,1 }`, `vector<int> dy = {-1,-2,1,2,-1,-2,1,2 }`, 并回圈遍历是否可走

- 国王：dx、dy 从-1 遍历到 1

因可以送吃，**只考虑目标位置是否可达，不判断造成王受攻击的不合法情况**。而只要兵走到对面末行，就自动弹窗出现升变提示，供使用者选择升变目标。

8.2 逼和

逼和比较困难，原因在于需要判断王当下是否被攻击，更要判断是否存在棋子能够合法移动（**移动后王仍安全**）。也就是说要遍历棋盘，判断可移动的点，再判断移动的时候是否有让王被攻击的可能性。而所谓的攻击，就是调用 8.1 节的方法进行走子判断与搜索。因此估算了一下，我写的算法的复杂度大概是 n^6，所幸棋盘规模小 $n = 8$，暴力法仍堪用。测了几组测资，目前都能正确运行。具体伪代码如下：

Algorithm 1 逼和算法

遍历棋盘，找到王的位置 (x_{king}, y_{king})
for 敌方棋子 (i, j) in 棋盘 **do**
    if (i, j) 可以攻击到 (x_{king}, y_{king}) **then return**
for 我方棋子 (i, j) in 棋盘 **do**
    遍历所有 (i, j) 可以到达的位置, 加入集合 $List$
    if $List.size() > 0$ **then**
        for (x, y) in $List$ **do**
            if 将 (i, j) 移动到 (x, y) 后, (x_{king}, y_{king}) 仍未被攻击 **then return**
逼和条件达成，弹出消息窗

9 问题分析与解决

9.1 兵升变时意外触发到关闭钮

图 12: 意外按到关闭钮，成为死局

有两种方法可以解决此问题：

- 重载 closeEvent(QCloseEvent*) 函数，拦截非按钮触发的关闭事件

```
void Dialog::closeEvent(QCloseEvent *event)
{
    if (!button_clicked) //拦截非由按钮触发的closeEvent
    {
        QMessageBox* msg = new QMessageBox
                                    (QMessageBox::Warning,
                                   "错误操作","请选择升变目标!");
        msg->show();
        event->ignore();
    }
}
```

- 在构造时向父构造函数 (QDialog) 传参，让其不出现关闭钮

```
QDialog(nullptr,Qt::WindowTitleHint | Qt::CustomizeWindowHint)
```

图 13: 两种方法对应的效果

顺利屏蔽关闭钮后,却发现还有一个问题:按下 Esc 键。查看了 QDialog 的 keyPressedEvent() 函数，发现按下 Esc 键时会调用 reject() 函数。再仔细查看 Qt 文档：

> If the user presses the Escape key in a dialog, QDialog::reject() will be called. **This will cause the window to close, but note that no closeEvent will occur.**

对于此问题也有两种方案：

- 重写 reject() 函数，将其置空
- 安装事件过滤器，筛选掉按下 Esc 键的事件

由于时间原因，我采取了第一个方式。

9.2 通信时所遇到的问题

在原先进行读档操作时，我每解析一句字符串就写入一条信息，但**信息没有及时被读出，而都被塞到了缓冲区**，造成读取信息时误判。解决此问题有两种方法：

- 避免短时间写入太多信息，同个函数中的信息合并後在函数结束前一起写入
- 不同信息利用空白格分隔，读取时使用 `split()` 压入指令栈，再依序剖析信息

10 心得

第二周学习的内容为主要为**网路通信**以及**多线程**编程，不过碍于时间原因，这次大作业我并没有使用到多线程的技术，而网络通信的部份也只是仿照课上代码进行稍为的微调。因此，要在课余时间再多把这周学习的知识点复习。

经过两周 Qt 大作业的洗礼~~（爆肝）~~，让我了解如何写一个与用户交互的程序（Qt 界面编程），以及了解到不同用户间交互传递讯息的方式（网络编程）。虽然还有很多知识点没精通，不过终于做出人生第一个可联网对战的小游戏了！看到自己的棋子在两个窗口中同时动起来的那一瞬间还是很感动的哇！谢谢助教与老师这几周来的指导，我一定会继续加油熬过小学期的 >O<