2019 年夏季学期　程序设计训练
第二周大作业报告

魏彤

计 86 班　2018011417

2019 年 8 月

摘要

本次大作业基于 Qt 和 socket 实现了网络对战的国际象棋小游戏。用户可以与互联网上的其他玩家建立 TCP 网络连接、设置黑白方和落子时间限、并进行标准的国际象棋游戏（包含兵升变、王车易位、胜负判定等功能）。此外，游戏还支持残局的保存和读入，并能时刻对网络连接的稳定性进行检测。

1　基本图形界面设计

图 1: 主界面

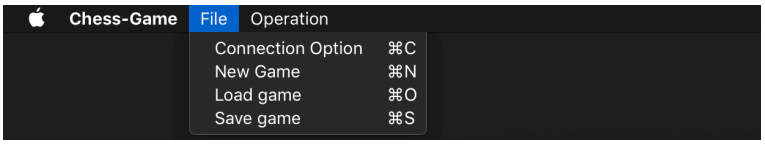


图 2: 菜单栏

以上为程序在运行时的主界面和菜单栏。其中，菜单栏包括了新建连接、新建棋局、载入棋局、保存棋局、投降和求和等功能的入口，并配以合适的快捷键。主界面包括主要的棋盘显示部分、右侧的状态显示栏、倒计时栏、投降求和按键和下方的网络连接状态栏，能够实现对于当前游戏状态的全面描述和方便的操控。

1.1 棋盘界面

考虑到国际象棋游戏的特点和功能需求，对于棋盘的显示和操控采用了 QGraphicsView、QGraphicsScene、QGraphicsItem 类。整个棋盘是一个 QGraphicsView 组件，建立了一个 QGraphicsScene 与之连接，可以完成显示的功能。对于棋盘格，继承 QObject（用于信号槽通信）和 QGraphicsRectItem 设计了一个 MyGraphicsItem 类，每一个棋盘格均为此类的一个对象。

通过在 MyGraphicsItem 类中定义函数和变量，可以统一棋盘格的接口，并很方便地实现棋子设置、边框设置、鼠标点击事件等功能，为之后的开发带来了巨大的便利。棋盘格通过 MainWindow 类中的一个 MyGraphicsItem 指针数组进行管理，可以通过行列的序号直接进行操控。

对于棋盘操控的详细内容参见 **棋子规则和操纵**一节。

1.2 右侧功能栏

右侧功能栏中，上方的状态显示栏在游戏中显示下棋方，在功能键被触发时也能进行对应的相应和显示。状态显示栏和 QMessageBox 共同组成了服务器和客户端操作的显示方式，不同的是，状态显示栏是非阻塞的，且不需要另一方回应的；QMessageBox 则相反，二者适当互补可以实现两端的合理沟通。

数字倒计时器用于每一步棋的计时。如果用户设置了每一步的时限，则会进行计时和显示。对于时间设置的详细内容参见 **时间设置和计时**一节。

1.3 网络状态栏

网络状态栏显示了是否连接、连接中的角色（服务器或客户端）、服务器的 IP 地址和端口，用户可以对网络连接的状态进行查看。对于网络连接的详细内容参见 **网络通信功能设计**部分。

2 网络通信功能设计

2.1 网络连接协议和编程框架

服务器和客户端之间的通信基于 TCP-IP 协议，利用了 Qt 中 QTcpserver 和 QTcpSocket 两个类。服务器输入本机的 IP 地址和端口，开始监听；客户端输入和服务器相同的 IP 地址和端口即可在两端建立稳定的 TCP 连接。

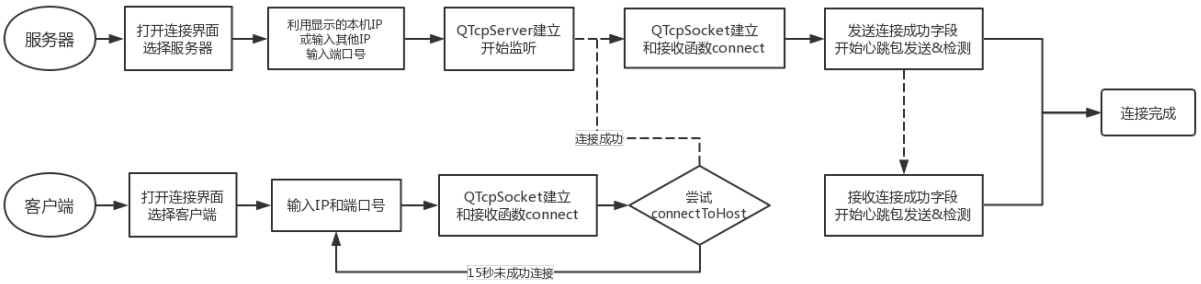


图 3: 服务器、客户端工作模式

上图为服务器和客户端工作模式。第一行表示服务器端的操作，第二行表示客户端之间的操作。实箭头连接的为同一个函数中的步骤，虚箭头连接的为不同函数之间传递信号。

考虑到发送数据的规模较小，服务器和客户端之间发送信息均为字符串。为了避免短时间内多次发送导致的粘包，发送时每一条信息均以 '\n' 结尾，接收时，会先以 '\n' 为标记进行分割，再对每一条指令进行解析、执行。考虑到发送数据的规模，并没有针对一条指令被拆进多个包的情况进行考虑和优化。

2.2 网络连接

图 4 为程序网络连接的设置界面。单选框可以选择以服务器或客户端进行连接，下方可以输入 IP 地址和端口号，最下方是连接和断开连接的按钮。

点击连接后，程序会对于 IP 地址和端口号进行格式检测。IP 地址必须是四段被点隔开的位于 0～255 的数字，端口号必须为 1025～65534 之间的整数。如果不符合格式要求则会弹窗要求用户重新输入。

程序通过 QNetworkInterface 库获得本机的全部 IPv4 的 IP，IP 编辑框默认会显示一个本机 IP，如果选择为客户端，点击连接后会检测输入的 IP 地址是否属于本机，如果不属于则会弹窗要求用户重新输入。

如果选择为客户端，点击连接后如果 15 秒内没有连接到服务器，则会显示连接超时的消息并可以重新输入 IP 地址和端口号。

服务器和客户端在连接尚未建立时均可以通过点击 Disconnect 键取消监听/连接尝

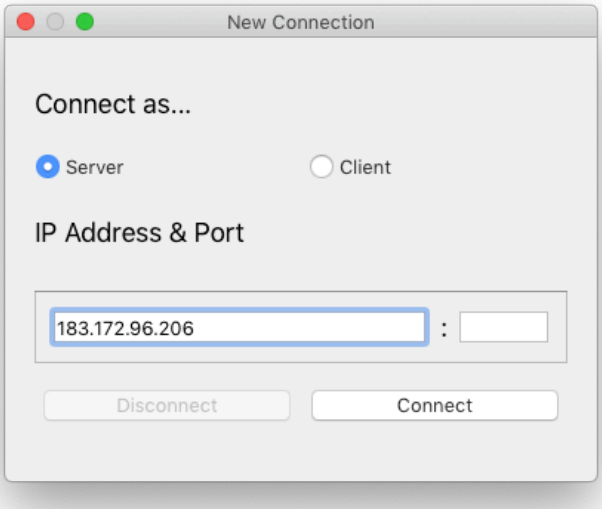
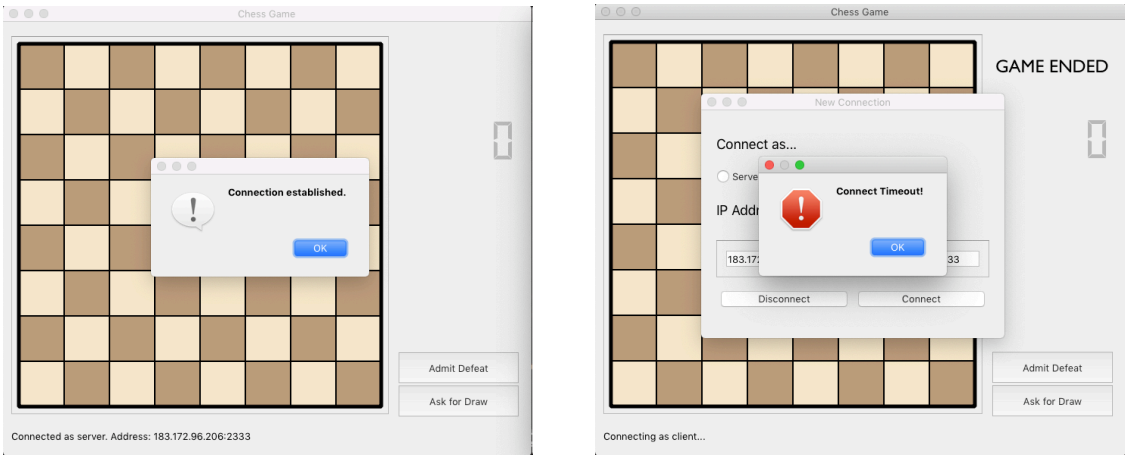


图 4: 网络连接设置界面

试，并可以重新输入地址。

如果连接成功，则会弹窗显示连接成功，并开始进行连接状态检测。同时，网络状态栏的”Disconnected” 会变为”Connected as server/client” 并显示服务器的 IP 地址和端口号。



(a) 连接成功　　(b) 客户端连接超时

图 5: 网络连接界面

2.3 连接状态检测

为了能够实时检测两端之间的连接状态，程序加入了心跳包机制。每一秒，服务器和客户端都会向对方发送一个内容为字符串”Heartbeat” 的数据包，并检测是否收到了

对方发送的心跳包。如果一方连续的 5 秒没有收到对方的心跳包，则判断连接中断，会触发断开连接的机制。这样确保了服务器和客户端的连接稳定可靠且状态被实时检测。

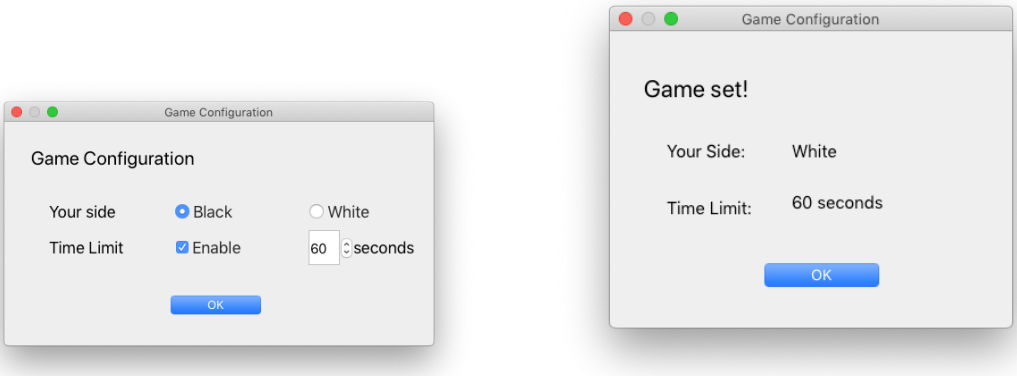
由于有些情况使用弹窗和用户进行交互，会阻塞心跳包的发送，因此在需要弹窗的场合，会暂时关闭心跳包的发送和接收判断，在弹窗被处理后恢复。

3 棋局功能实现

3.1 棋局初始化

在连接完成，输入 New Game 或 Load Game 后，服务器会出现设置棋局的页面，包括持子、是否加入每步的时间限制、时间限制的长度。在服务器完成设置并确定后，会等待客户端的确认。客户端则能够看到相对应的信息。

在客户端确认后，时钟会进行同步，棋盘会进行初始化——己方的棋子在下方。值得注意的是，由于使用了 QGraphicsView，这一操作仅通过每一个棋盘格的 setPos 函数便可完成。



(a) 棋局设置　　(b) 棋局确认

图 6: 棋局初始化界面

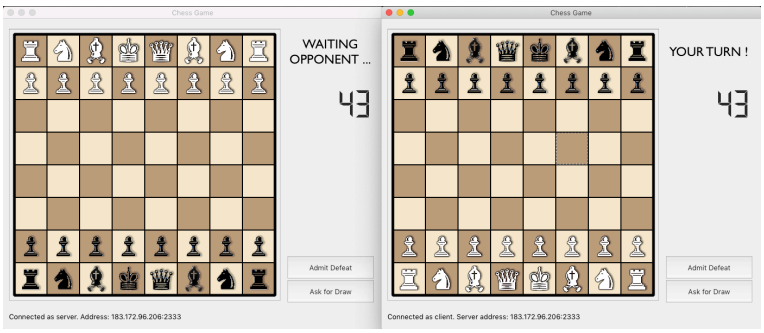


图 7: 棋盘初始化

3.2 棋子规则和操纵

每个棋盘格继承了 QObject，可以发送信号。在鼠标点击格子时，会发出坐标信号，被 MainWindow 中的信号槽接收。从而进行逻辑判断。

由于棋子的移动涉及到两次点击，因此设立一个 bool 变量区分两次点击。但第一次点击被识别，且点击了己方的棋子，则将该棋子所在的方格边框设为蓝色，表示选中。同时，根据棋子的坐标和类型，按照国际象棋的规则，程序计算棋子能够到达的方格坐标。如果方格为空，则会将边框设为绿色；如果方格中有敌方棋子，则会设为红色，表示可以吃子。（具体效果参见**图 1**）如果点击了己方的王，且满足王车易位的条件，王的目的地会被设为紫色。关于王车易位的更多细节，参见 **王车易位和逼和**一节。

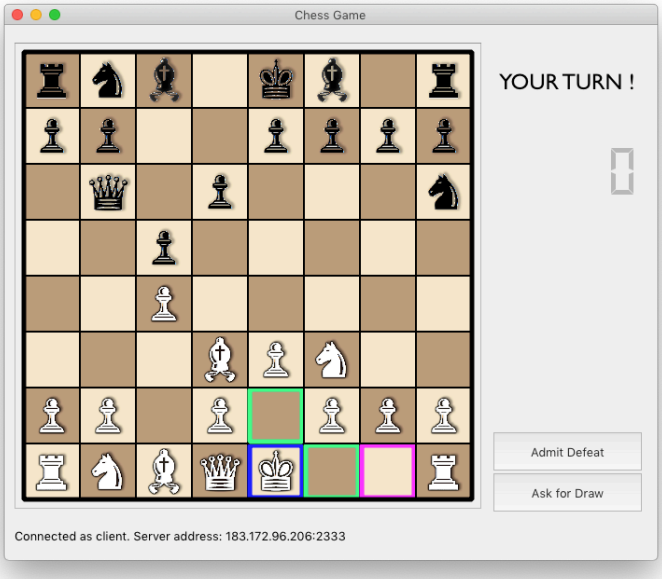
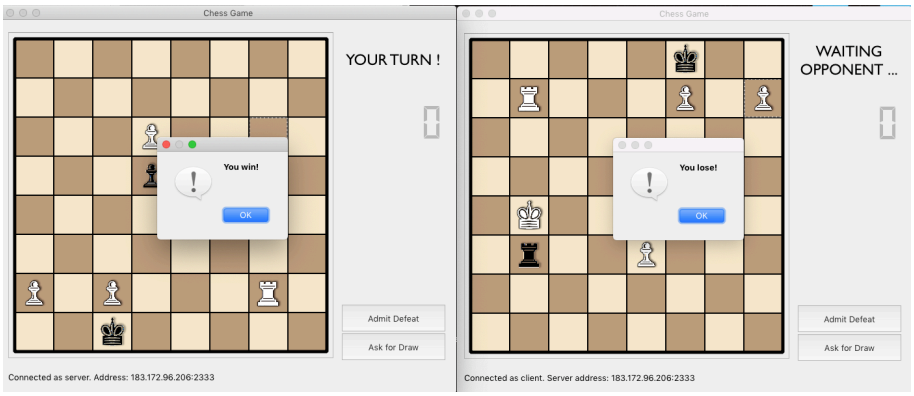


图 8: 王车易位

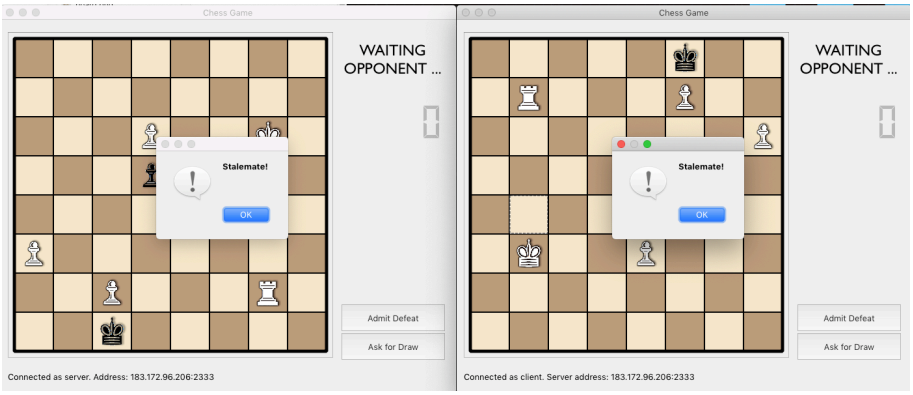
在移动操作完成后，目的地方格的棋子会被设计为和原位置一样，原位置的棋子信息会被设为空。同时将移动信息发送给另一端，另一端解析后进行棋盘同步。

3.3 胜负判定

在每一次移动后会进行判断，如果目标位置是对方的王，则证明对方的王被吃掉，此时判断己方胜利，敌方落败，会进行消息传递和弹窗显示。如果满足逼和条件，则会两方显示逼和。关于逼和判定的更多细节，参见 **王车易位和逼和**一节。



(a) 一方胜利



(b) 逼和判定

图 9: 胜负判定界面

3.4 时间设置和计时

如果棋局设置了每步棋的限时，则数字区域会显示倒计时，两端的时钟在游戏开始时同时被启动，每一次移动后又重新复位，依次保证两端的同步。
如果倒计时停止依然没有完成棋子的移动，则会超时判负，两端都会收到相应的弹窗。(参见**图 10**)

3.5 兵升变

当兵移动到最后一列是，在第二次点击的函数里会进行判断并弹出兵升变的界面，用户选择升变的种类。之后，对于棋盘进行更新，并将相应的信息传给另一端进行棋盘同步。(参见**图 11**)

3.6 王车易位和逼和

王车易位和逼和功能的实现都依赖于对于敌方棋子攻击范围的判断。程序因此设置 getControl() 函数，遍历每个棋盘格，如果有敌方棋子，则将其能够攻击到的区域进行标记。

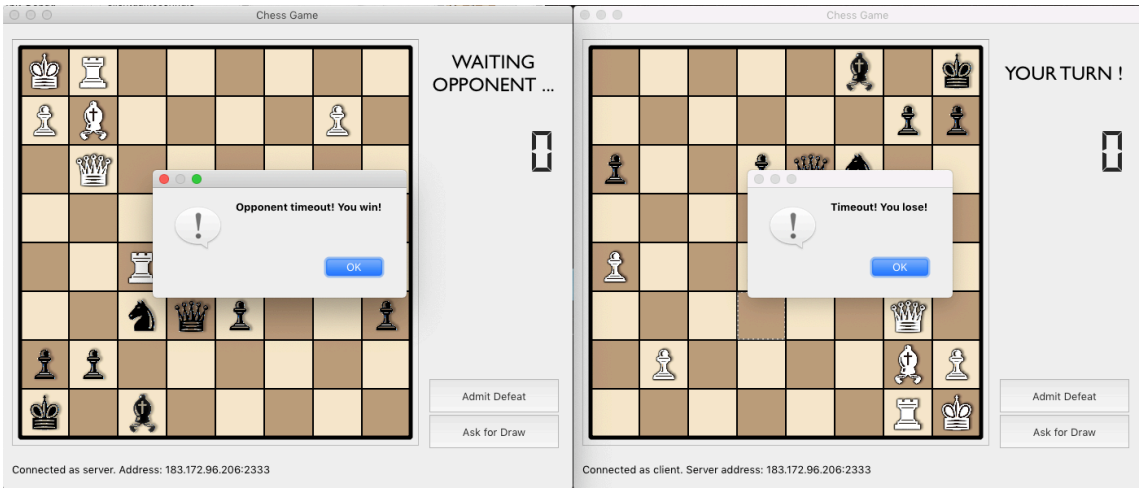


图 10: 超时判定

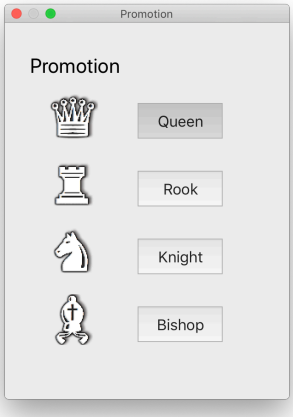


图 11: 兵升变选择界面

当检测到第一次点击到己方的王时，先依次判断王的位置、对应的位置是否有己方的车、中间的格子是否没有棋子。如果都满足则计算敌方控制范围，如果王经过的三格均不在攻击范围中，则判断可以进行王车易位，将相应的格子显示出来。

每一次敌方移动之后，也要获取敌方控制区域，如果己方的王在攻击范围中，则判断不是逼和；在此基础上如果王能够到达的区域均是敌方控制区域，则再一次遍历棋盘，如果所有己方的棋子均没有可以移动的地方或己方棋子的移动会导致王被将死，则判定是逼和。

3.7 认输和求和

通过菜单栏中的入口和主界面上的按钮都能够实现认输和求和的操作。其中，求和会向对方发起询问，如果拒绝则继续棋局，同意则在两方显示弹窗。认输则直接在两边弹窗显示结果。

4 残局存储、读取功能实现

4.1 残局读取

残局读取和开始新游戏类似，由服务器端进行游戏配置，在客户端进行确认，之后由服务器端选择文件并载入。通过 QFile 和 QTextStream，程序将输入的内容按照残局文件的格式进行解析并将棋盘对应位置的棋子更新。

之后，服务器端会生成描述棋盘和回合的字符串并发送给客户端。字符串以当前下棋的一方开始，w 表示白棋、b 表示黑棋。随后按行按列依次将所有棋盘格上棋子的情况写入字符串中。e 表示空白，小写的 g、q、h、k、r、p 依次表示白方的王、后、象、马、车和兵，大写的 G、Q、H、K、R、P 依次表示黑方的王、后、象、马、车和兵。客户端根据发来的字符串，对当前回合和棋盘上棋子的情况进行设置，实现棋盘的同步。

4.2 残局存储

残局存储可以由两方的任意一方完成。程序会遍历棋盘的所有格子，用 vector 数组记录每个类型的棋子的坐标位置。随后将对应的数据按照残局文件的格式写入。通过弹窗，用户可以选择保存的路径和名称，再通过 QFile 和 QTextStream 进行文件写入。

5 心得

本次大作业是我第一次编写可以进行网络通信的程序，在写大作业的过程中通过上课、网络和同学学习到了很多关于 TCP/IP 和网络的基本知识，也在制作小游戏与其他同学对战中获得了成就感。

本次大作业我不但完成了基本要求和附加要求，还在此基础上进行了许多的优化，比如心跳包机制、断开重连机制和服务器客户端之间更多的交互和选择。令我收获颇丰。

大作业程序仍有一些方面可以进一步改进，如用户界面可以更加精美，可以对将军

和将死的情况进行提示，网络传输的数据可以进行恰当的编码和精简，这都是今后可以继续改进的方向。