“Five in a Row” Class Project

CS43600 Principles of Computer Networking

Myson Burch

Nick Baker

[mcburch@iupui.edu](mailto:mcburch@iupui.edu)

[nvbaker@iupui.edu](mailto:nvbaker@iupui.edu)

Myson Burch University ID: 0003198940

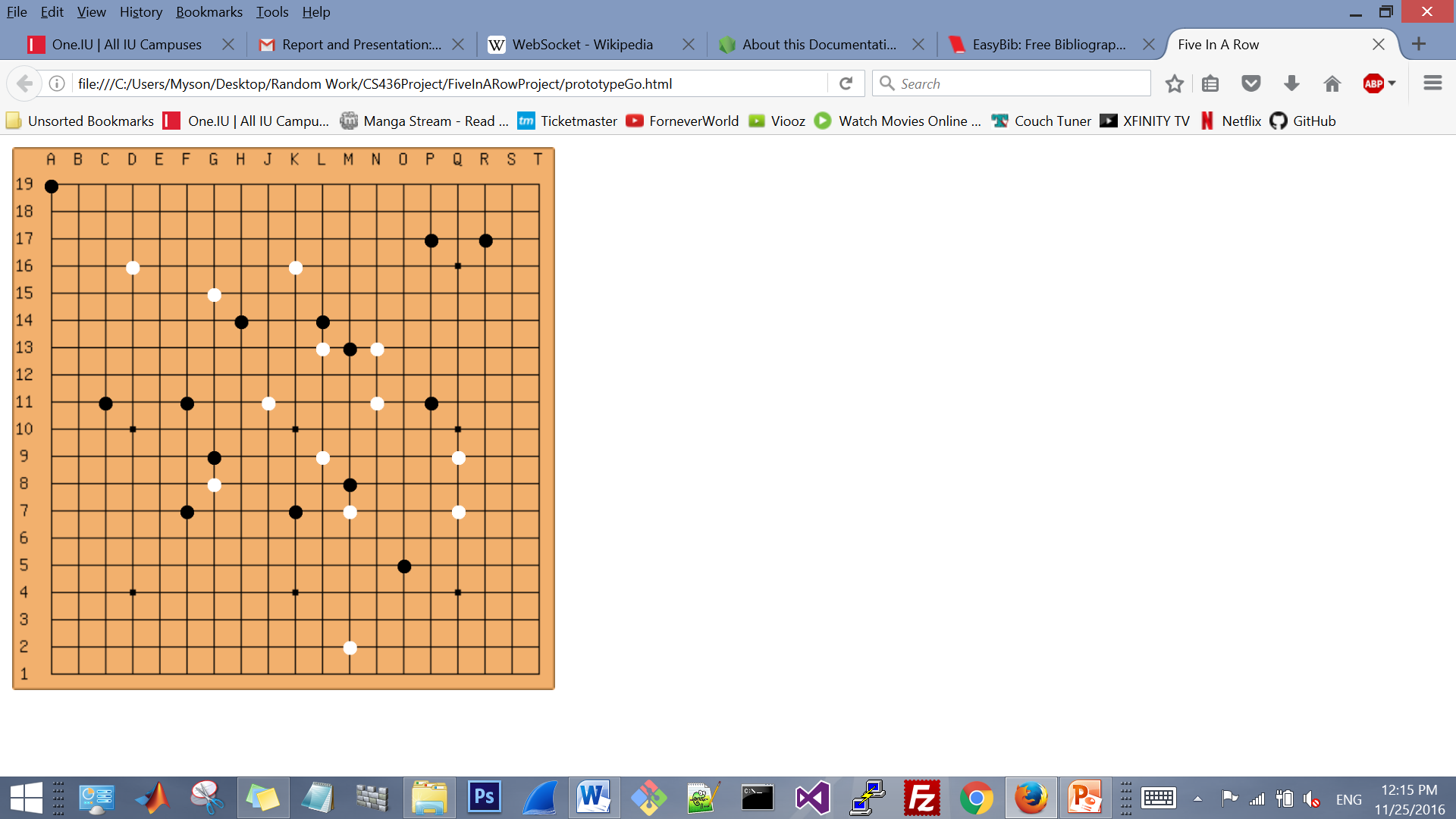
Nick Baker University ID: 0003246911

2 December 2016

The goal of this project is to create a peer-to-peer networking application that two peers can use to play Gomoku or the “Five in a Row” game. The game needs to use HTTP protocol to implement the game in real-time and also Wireshark needs to be used to keep record of the network protocol activities for both players during the game. The time given to complete the project was approximately two weeks.

We chose to use Node.js in our implementation of the game. Node.js is an open source, cross-platform Javascript development environment with many tools and libraries that make this a feasible approach to making this network application. Node.js is event-driven which seems natural to use in a click-based game where there are many input/output operations with real-time communication between the peers. Node.js is compatible across OS X, Linux, Windows and other operating systems. Node.js is also lightweight and efficient which makes it convenient to code in especially with so many open source libraries we have access to. We explored many libraries to try to make the application work. One of the libraries was the WebSocket library. WebSockets are a bi-directional and persistent connection from a web browser to a server that we will use to implement the network side of the application between the peers. This allows us to communicate with any number of open connections at a time since the websocket initiates a single running server, but for our purposes we will only need two connections. Another library we explored that also implements WebSockets was Socket.IO. Socket.IO similarly enables bi-directional event-based communication in real-time and is a more powerful and efficient extension of WebSockets. We are using GitHub to host and manage our project. Git is very useful in terms of version control, feature requests and error tracking. Using this application to host our project allows us to work more seamlessly as a group and is also convenient.

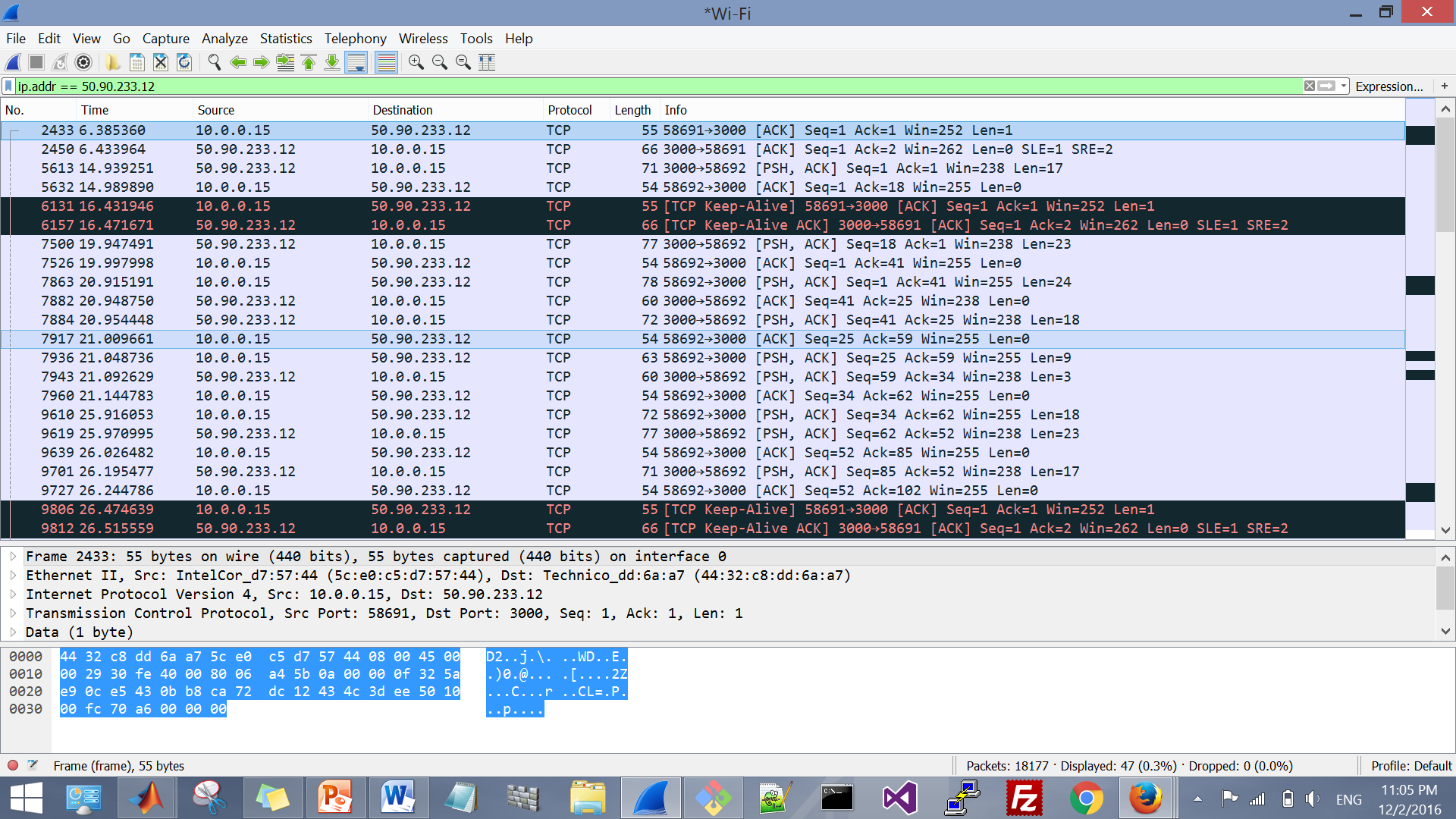
We decided to design the game using a simple game engine developed by Mr. Andy Harris at Indiana University-Purdue University of Indianapolis. As said previously, the peer-to-peer aspect will be handled using Node.js and the node.js websocket library.



**A**

**Figure A**: This is our design for the game board of the “Five in a Row” game. Each player has a different colored piece (black or white) which alternates based on whose turn it is. The board is from a Google image and we used Javascript to make each intersection clickable so that a piece can be placed on the board. Each piece on the board is a Sprite which is an object with position functions, click functions, etc. Myson Burch did have to add a getPosition function to Andy Harris’ game engine for the desired functionality. Each piece is placed into an array based on their respective colors and then after a click occurs, a function to check for wins is run. The algorithm works with each array and basically detects adjacent elements in a certain direction then keeps searching in that direction and if there is 5 elements in that direction then the player wins. An intersection on the board cannot be clicked twice so the issue of pieces changing colors is avoided. The algorithm possibly can be improved but we chose to stick with that functionality in order to work further on the network aspects of the project. Once a winning scenario occurs for either player, an alert will appear announcing the winner and the game will end. The editor used was Notepad++.

**B**



**Figure B**: Wireshark needed to be used in our project to analyze the network protocols in our application and show what exactly is going on between the peers. Wireshark is an open source network analyzer that is a very powerful tool that can be used alongside network applications. The analysis shows us the data communication between the players via TCP protocols. HTTP protocol is used for the handshake. The negotiation of parameters over a communication channel in websocket is interpreted by HTTP servers as a request for an update. As the game progresses, Wireshark will inspect the communication protocols between the players in real-time. Nick implemented port forwarding and port forwarded the server and connected to the public IP address when playing and testing the game. The figure shows one of the clients communicating with the server and passing information back and forth while playing the game via TCP protocols. The game was created on the client-side by Myson Burch and Nick Baker initially had some issues implementing the peer-to-peer aspects using various libraries of Node.js but eventually was able to complete the network aspect of the project using Socket.IO.

A common constraint on any project is time. We did not have as much time as we would have wanted to rigorously outline our design and implementation. Because of this, we tried to make the simplest implementation we could think of so we could get the game working locally on our machines then work on the network elements of the project afterwards. Another roadblock that we had to overcome was the actual design of the game. Obviously once a development stack is picked and certain design choices are made, certain issues are automatically introduced in terms of performance, compatibility, and overall feasibility of implementation of the task. The algorithm for checking wins can possibly be improved versus looping through an array to check for the winning conditions. Lastly, applying the websocket library to make the game a true network application between two peers was a constraint on the development. Since our group had not dealt with many network applications before, this was new territory for us.

Myson Burch was responsible for making the game work client-side before extending the game as a network application. Myson was also responsible for writing the report, making the PowerPoint presentation, and presenting the client-side demo. Nick Baker was responsible for making the game peer-to-peer and presenting any coding implementation issues in the demo. The Wireshark analysis was done as a group.

Despite any roadblocks we had, we were able to successfully implement the game and also take away some lessons and learn from the entire process. There is much room for improvement for our short-term project management as well as project design and we will take things we learned developing this project into other projects, collaborations, and assignments we may be a part of in the future.

References

1. "Nodejs-websocket." *Npm*. N.p., n.d. Web. 27 Nov. 2016.
2. "Node.js V7.2.0 Documentation." *Node.js V7.2.0 Documentation*. N.p., n.d. Web. 25 Nov. 2016.
3. Harris, Andrew. "CSCI 437(00) Game Development I." *CSCI 437(00) Game Development I*. N.p., n.d. Web. 25 Nov. 2016.