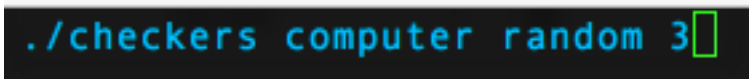
Ryan Hawkins

CS457

Project 1 – Official write-up

Checkers A.I.

I did not work with anyone on this program. My minVal and maxVal methods are coming from what we did in class, and I do not remember making any changes to the code given, that remains unless stated later somewhere in the paper. Since we use minVal in FindBestMove I have done a lot of changes in there (Find Best Move). I also have changed evalBoard to what I thought would be a better heuristic, although it is very struggling to understand what exactly is the correct heuristic to follow so I felt that I changed a lot, then just removed it because It seemed to hurt my game playing. I built my program by ssh'ing into the school onyx system. I had no problems compiling and playing with the following command line prompts.




**evalBoard:**

Right off the bad when you enter the evalBoard and start your loop to check the board and the places, I added in an algorithm that would allow it to skip the places on the board that would not have any checker pieces on them. Creating a faster evalBoard algorithm with less checks on the board. Although I don't know how much this increased my time.

In the evalBoard I decided to add more points to a player when they were located on the outside of the board positions, since those prevent pieces from being jumped, it seemed the less likely you would be jumped by the opponent. At this point I will still lose to random

which I don't understand. The more I put emphasis in having 0-2 points compared to the opponent having 4 – 12 points the more I seemed to lose games more often. I noticed a lot of games where these numbers for the opponent would be near 7, and even though I would setup my eval to give negative numbers that should easily be beat, I was somehow losing to random.

**FindBestMove:**

For my findBestMove Method I started initially with the class given code that just searched through all the best moves and picked the one with the best value. I started with changing how I saved best moves that may be equal to each other, and then randomly grabbing one of those choices. To do so I made an array that was a size equal to the length of our possible moves for our state, (I did this size to ensure I never ran out of room, I could have done it other ways like making a new array with more space each time I needed to add something to my list, but this was not top on my list and other things took over priority), so the only real issue there is the size you are using and the extra space you won't have IF you ever needed it. When we would find an equal size element to our current we would add it in our list, keeping count with an extra counter to ensure if I found a greater size, it would start over in the list and then at the end, I would choose a random number from the size of the counter. So, if counter was 3 we would choose a random number from 0-2 and then grab from that index in the Array we have stored of equal numbers. We start with our currBestValue to be a super large negative number that will be guaranteed to be beat the first time in, checking for larger values and storing them to be used randomly later. For ease if there is only 1 best value ino our array we will just go with the first item in the array.

**Threading & timing:**

I was not successful on any threading or timing before the due date extension and was more focused on trying to get my board to beat random. I attempted the threading but something somewhere was not right and so I stopped focusing on adding that and I just never got back around to getting the threading in.

**Testing my game:**

This part was rather hard as I could not get items to print during the execution of the game. I must have been using an incorrect stderr print. Due to this a lot of my testing came from just running the code against whatever opponent I wanted to test against. To test how my player would do as second player I would just switch the $1^{st}$ and $2^{nd}$ input arguments (starting at 0) and I would run the program to ensure it would perform for the second player in the manner it had been for the first player. It did not crash when I ran and never made an illegal move, leaving me to assume that it was still performing properly.

Another way to test that I used was a website that was an online compiler, through this I could copy and paste my code and then run tests on the code and print things out, making it sure that my algorithms were handling the way I expected them to.

**Conclusion:**

I think one of the most challenging parts of this assignment was the fact that we could discuss what we felt would be a dominant A.I that could compete, and when you go to implement things it was very easy to find yourself having to think deeply about what you wanted, and how to accomplish that. Changing your program was rather tedious, due to facts like when you made a change and it would start causing more chaos, I personally would scrap it if I couldn't find a clear mistake. Given the limited debugging tools that I know and that I could

get to work, I could not always guarantee my program was doing what I wanted without going and finding an online compiler, copying the code, and then running print statements to make sure it was doing what I wanted it to. Specifically for one case, when I was figuring the math for what spots to not check, I wanted to print my x and y value to make sure I was always skipping the right one, to do so I found an online compiler.

I am unfortunately not confident in my program at all, I do not expect to do well in the tournament as I was not always beating random, and for that I worry about my grade. I had trouble understanding the printout of the checkers board, and how to use that visually to see what I was doing on a board during the game. I am still very unsure about how that was working exactly, so I did my best to not use the checkers board visual unless it was to see what my end results were with opponent keys vs my keys. Some of my random losses to the opponent where their score will be 7+ with almost all kings, so I can't help but feel I have just made some really silly mistake in my heuristics as I see no reason why my board should even let a random player get to that point. Needless to say this was a frustrating project but I worked extremely hard so the program definitely does not seem to represent the work I put into it.

Total time I spent on this project was roughly over 8 hours a week for 2 weeks and then about 7 hours within the last few days trying to get things into the best working condition for assignment turn in. Those are rough estimates as I forgot to keep a work log of when I was working and when I stopped, etc.

I think I messed up by not doing a couple of things. The first thing by not finding a partner to work with, the normal person I have partnered with in the past, found another partner and I just did not ever work in a place where I met someone else who wanted or

needed a partner. Working with a partner would have at least allowed me to get through the threading and timing I am assuming, plus 2 brains always seem to be better than one on projects. Another reason I messed up is I seemed to have thought way too deep in the heuristics, I know a lot of people got stuck in there with random but I am not sure if they ever were able to fix their issues. Maybe I spent too much time attempting things on my own rather than speaking with someone who had it, at the same time I was in the lab I would speak with some other teams about just their thoughts and ideas one what sounded good. Nothing seemed to hit a caution with anyone and I think we all went our own ways with how we wanted to add those features. Most of our discussions seemed to be things like, “hey did you notice the timer is funky?”, replied with, “yeah, try (insert C timer class)”. I look forward to redeeming myself in the next program that we have for this class, as I definitely felt defeated in my quest to finish the checkers board A.I. This was a challenging project, and I always say this for classes, but I wish we could do one class at a time for school because when you can focus on one thing for 5 days straight vs 4 classes in 2-3 weeks (I personally), seem to benefit more from that. I am sure during break I will look at this again especially after class discussion and time to think about it.