Name: Duc Manh To

SFU ID: 301425895

Discussion Report For Time Limit

1. There could be many reasons why the time limit set in the GUI is surpassed in runtime, but I've narrowed it down to a few hypotheses:
    a. Inaccuracies in time calculations: The use of time.perf.counter() might not be precise enough to control the exact depth I wanted, leading to slight delays.
    b. Function calls overhead: The algorithms I've implemented made multiple function calls, each creating overhead including memory allocations, copying… As the depth of the tree increases, the number of operations increases making the program exceed the time limit.
    c. Performance of the machine: As multiple processes run simultaneously while I complete the work, the program's performance might be affected. It can freeze quite frequently in some cases, namely running the cutoff functions with no time limit.
    d. Non-atomic operations: As each iteration runs, their runtime can vary significantly, resulting in the actual runtime easily exceeding the specified limit.
    e. Branching factors in Minmax and Alpha-Beta: In these algorithms, the branching factor can vary at different levels of the tree. A high branching factor could cause certain nodes to take noticeably more time.
2. Potential fixes:
    a. Add more time checks to the program. This can be added after significant steps or to the recursive calls to make the runtime more precise. However, doing so will lead to higher complexity and overhead.
    b. Modify the code to use threads. This allows the main program to execute and interrupt the processes, enforcing the time limit more strictly.
    c. Dynamic iterations/depth limit: This can be achieved by testing the amount of time iterations and depth levels take to complete and make an estimate of iterations/depth levels that can be reached based on the time limit. This is by far the hardest to estimate using calculations and takes the most time to test by hand, so this way is not recommended.