Final Report: Golf Course Tournament Pace of Play Simulation Model

Ashton Miller
CS 4632 Modeling and Simulation

December 6th, 2024

Abstract

This Golf Course Tournament Pace of Play Simulation Model is a model that studies and manages the time discrepancies that occur on a golf course in a standard tournament setting from the first group all the way to the last. The main objective of this project is to give a user an insight into how their golf course operates based on the time performance of groups and to most efficiently process a tournament based on configuration management. In executing the simulation it will be found that the simulation model itself reflects data of real world data within its own designated results. Upon the reading of this report, the reader will:

- Gain valuable information and knowledge of how the game of golf works and operates.
- Study key findings in outside research reports through the form of literature reviews.
- Be presented with an overview of the model itself to describe how the system works with the support of graphics.
- Have a baseline of simulation results provided to them to better help understand the model at a lower level.
- Be shown a scenario and sensitivity analysis and the results of different scenarios within the projects data studies.
- Be presented with a validation and verification section that emphasizes the testing process and the readiness measures of the project.
- Be given information regarding the results of the analyses that occurred through the project development and how those data points shape the project and its value as a whole.
- Finally, be briefed on the key conceptual and experimental findings and how these findings will contribute to future development work on the project.

Contents

1 Introduction

The game of Golf has been played for centuries around the world and has become one of the most beloved games to ever exist. In golf, the goal of the players is to stay within a certain number of strokes to maintain the lowest score possible before reaching the cup at the end of the hole. Throughout the game's history, one factor that has absolutely never changed has been the factor of time. Time has been and always will be the leading problem with the game of golf. The role that time plays is very important because a lot of times in normalized scenarios, the amount of time taken for a standard round of golf in multiple rounds per day could potentially make or break your business as an owner. However, this report studies the Pace of Play model directly in golf that monitors the pace at which groups on a golf course conduct themselves to maximize course efficiency and round output. By looking into this model in the scope of a tournament setting, the project aims to present the randomness in human nature that exists within the bounds of a group playing the game on the course to further examine how the system of a golf course accurately and effectively works in execution. The motivation of this project is to provide a user, like the management of a golf course, to more accurately and in modular fashion study the time discrepancy trends that exist on their golf course with their own specific course configuration. This will be useful for planning tournaments and deciding which hole to start the golfers on and which groups to start first. The goals of this project are to help teach those who have less of an understanding of how the game of Golf works structurally, display and study how different time discrepancies in pace of play between groups are handled, and observe comparisons of data to real world trends to support the projects validity.

2 Model Overview

This model replicates that of the pace of play model on a golf course and presents the relationship that golf courses have with the never-wavering factor of time. Within this model, there are many classes and functions that that contribute overall to the purpose of the program.
To start in the mod_sim.py file, exists the ModSim class, the brains of the operation. In this class, exists a main function that is utilized to print off introductory print statements, a get_par_wait_time function that configures the holes of the golf course to a designated par number, a get_standard_time function that sets a standard time according to a specific hole and its already determined par type, a play_hole function that handles the timeouts that occur within the simulation to account for the time it takes groups playing golf in the tournament, a send_cart_to_course function that processes groups throughout the golf course from hole 1 to 18, accounts for the groups traversal by way of print statements, and implements a return gap for carts so that no two groups finish the tournament at the exact same time, a cart_recieve function that uses 26 tee times and 52 carts to kickstart each group off of every hole from the very beginning of said holes using an 8 minute time stop gap to stagger group sendoff, and finally a cart_return function that returns carts back to the cart barn by helping the send_carts_to_course function perform its duties.
Next, the course_operator.py file containing the CourseOperator class has two functions that are used to 1, detect when a group is playing slow by calculating their time discrepancy for the hole and using a set of conditionals to determine if they are ahead, behind or within pace, and 2, send and deploy a marshal to the golf course in conjunction of a function call from the detect_slow function. A course operator class is used to set some simple attributes about the clubhouse using constructors and init functions.
Finally, a hole_par file using a HolePar class is used to define a standard random time range that groups operate within when the simulation is executed. These time ranges are drawn up based on the premise of having 2 minutes either ahead or behind pace per hole to better account for the randomness of human nature. Below in figures 1 and 2 can be found two diagrams detailing the relationships of different elements within the project.

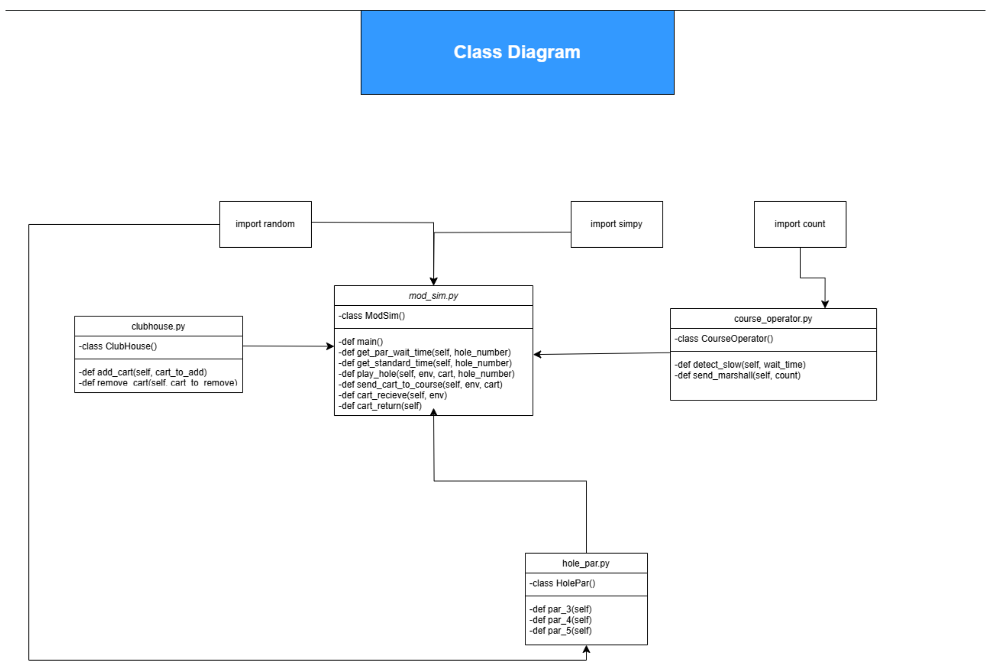

Class Diagram
import random
import simpy
import count
mod_sim.py
-class ModSim()
-def main()
-def get_par_wait_time(self, hole_number)
-def get_standard_time(self, hole_number)
-def play_hole(self, env, cart, hole_number)
-def send_cart_to_course(self, env, cart)
-def cart_recieve(self, env)
-def cart_return(self)
clubhouse.py
-class ClubHouse()
-def add_cart(self, cart_to_add)
-def remove_cart(self, cart_to_remove)
course_operator.py
-class CourseOperator()
-def detect_slow(self, wait_time)
-def send_marshall(self, count)
hole_par.py
-class HolePar()
-def par_3(self)
-def par_4(self)
-def par_5(self)

Figure 1: Updated UML Diagram Displaying Class Relationships Within the Project

3 Simulation Results

Through the semester, there were many different tests that were run to observe the nature of the model. The first of which was the baseline simulation run to test the overall functionality of the model. This leg of the project specifically aimed to represent the program running with no specific changes or alterations to the code to account for scenarios whatsoever.

The way that data was gathered in this baseline run was by the use of the RegEx package that sampled a population of text files for keywords and gathered data on the model based on those specific keywords. below, data will be discussed relating to the simulation runs themselves.

3.1 Overview of Runs:

The three main scenarios that are tested here are the number of times the marshal gets sent out to the course, the amount of time that it takes a group to finish their round from starting time stamp to finish, and how frequently a group falls behind pace, ahead of pace and within pace on the course.

You will find a direct correlation between two separate runs where the times taken to complete a round and the stress on the marshal, being the number of times they are sent out, are directly affected by the occurrences of groups falling behind or ahead of pace.

3.2 Data Collection Methodology:

For the collection of data I took 10 runs of the program and saved the outputs to 10 text files where I then used the re import to gather data based on keywords within each of the files. Such keywords were as follows: ”Marshal sent”, ”ahead of pace”, ”behind pace”, ”stayed”. The count of occurrences of these specific keywords defined the overall numbers within the data.

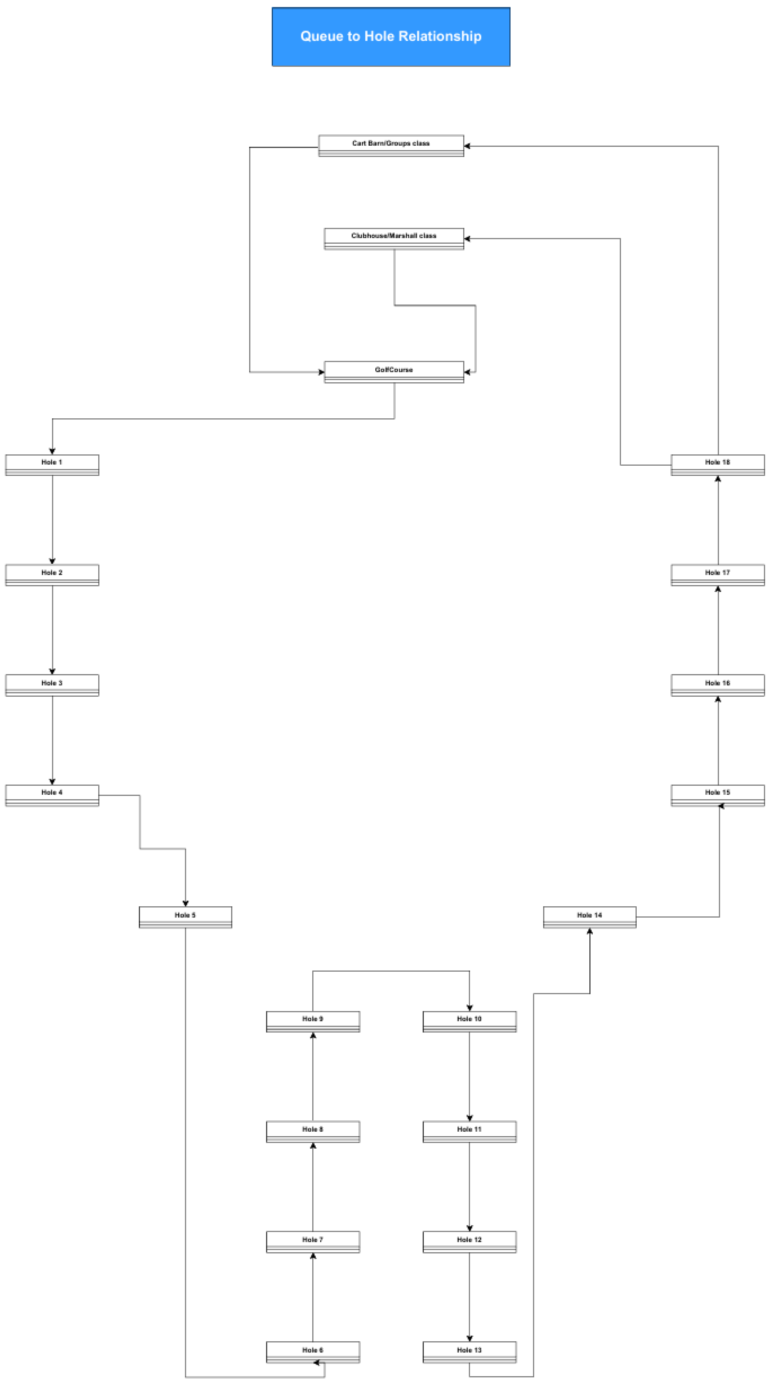
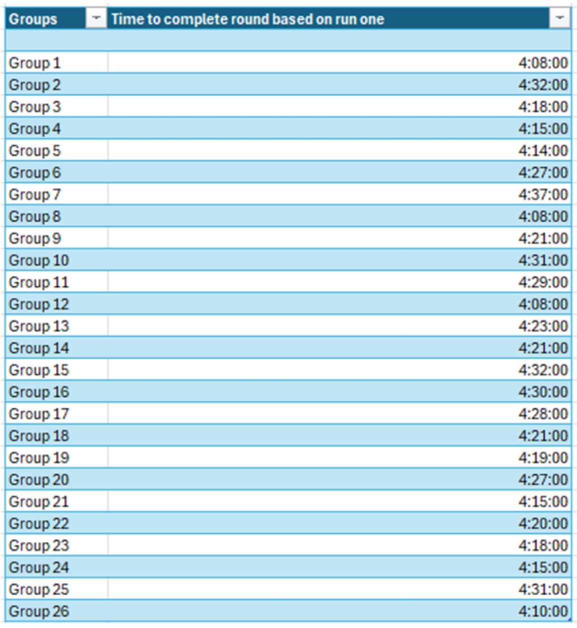


Figure 2: Entity Relationship Diagram to Display the Relationships of Each Hole to the Course Management Queue



Groups	Time to complete round based on run one
Group 1	4:08:00
Group 2	4:32:00
Group 3	4:18:00
Group 4	4:15:00
Group 5	4:14:00
Group 6	4:27:00
Group 7	4:37:00
Group 8	4:08:00
Group 9	4:21:00
Group 10	4:31:00
Group 11	4:29:00
Group 12	4:08:00
Group 13	4:23:00
Group 14	4:21:00
Group 15	4:32:00
Group 16	4:30:00
Group 17	4:28:00
Group 18	4:21:00
Group 19	4:19:00
Group 20	4:27:00
Group 21	4:15:00
Group 22	4:20:00
Group 23	4:18:00
Group 24	4:15:00
Group 25	4:31:00
Group 26	4:10:00

Figure 3: Table documenting the times taken per group in the first run tournament

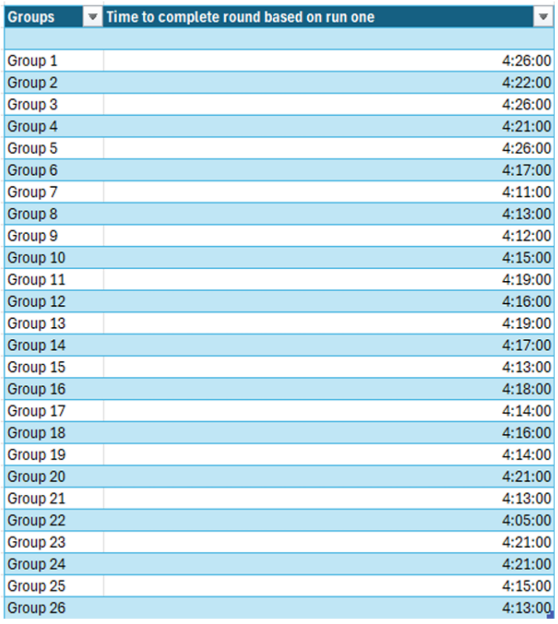
3.3 Data Formats and Storage:

The data was stored as console output in a .txt file 10 times for 10 runs. This data was processed within excel in the graphs and charts below.

3.4 Results

3.4.1 Discussion

The charts and graphs that will be found in figures 3-7 are very significant to the data due to the way that they directly correlate with one another and show the relationship between time taken to complete a full round per group and how often they fall ahead of/ behind / within pace. As you can see for run one in the time chart groups like group 1, group 8, and group 26 contribute to the overall decrease in number of times the marshal is deployed, however there are many data points within this run where groups finished much later and took much longer to complete a round therefore, there will be a result like in run 1 on the marshal deployment occurrences graph shown above. This also explains why in run 1 of the ahead/behind/within graph there is a much larger time discrepancy between the number of times groups got ahead of pace vs behind pace. Groups in this run after the run finished were observationally much slower than they were quick or on pace.
If you take a look at run 7 and the time to complete the rounds within that run, you can correlate the much larger discrepancy between a much larger number of ahead of pace counts vs a lower number of behind pace counts. It is also noticeable that the number of times all the groups within that run fall within pace is larger than the average amongst all 10 runs.
These results that are shown amongst the two charts and the graphs provided are a direct result of how the number of times a marshal on average is sent out for a group being 5 minutes over pace is directly related to how long on average it takes for a round in a run to finish.



Groups	Time to complete round based on run one
Group 1	4:26:00
Group 2	4:22:00
Group 3	4:26:00
Group 4	4:21:00
Group 5	4:26:00
Group 6	4:17:00
Group 7	4:11:00
Group 8	4:13:00
Group 9	4:12:00
Group 10	4:15:00
Group 11	4:19:00
Group 12	4:16:00
Group 13	4:19:00
Group 14	4:17:00
Group 15	4:13:00
Group 16	4:18:00
Group 17	4:14:00
Group 18	4:16:00
Group 19	4:14:00
Group 20	4:21:00
Group 21	4:13:00
Group 22	4:05:00
Group 23	4:21:00
Group 24	4:21:00
Group 25	4:15:00
Group 26	4:13:00

Figure 4: Table documenting the times taken per group in the seventh run tournament

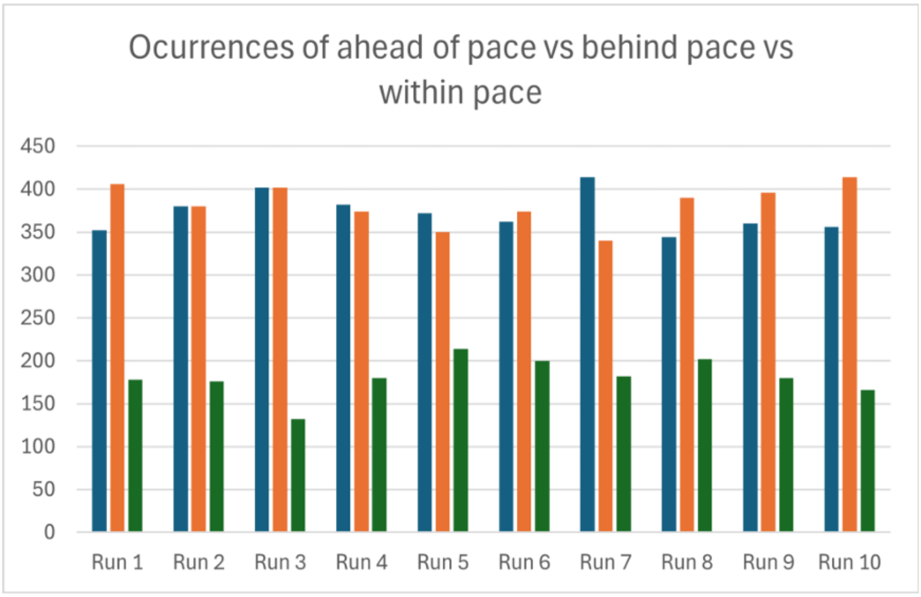


Figure 5: Result bar graph that details the number of times groups per run fell ahead of, behind, or within pace respectively.

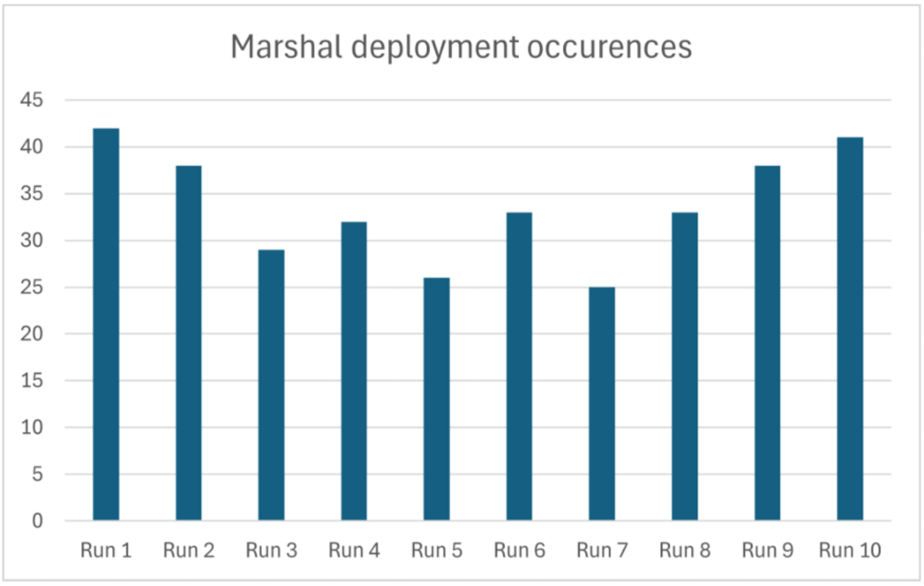
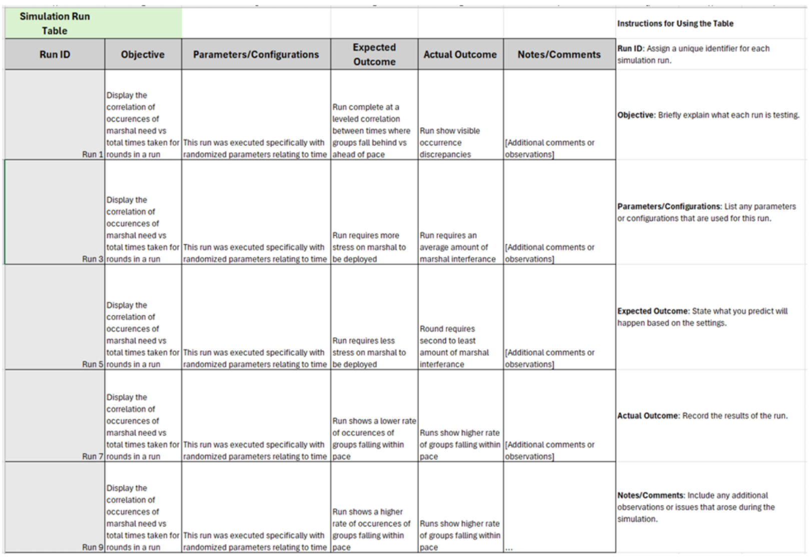


Figure 6: Number of marshal deployment needs where groups surpassed 5 minutes over pace.



Simulation Run Table

Run ID	Objective	Parameters/Configurations	Expected Outcome	Actual Outcome	Notes/Comments
Run 1	Display the correlation of occurences of marshal need vs total times taken for rounds in a run	This run was executed specifically with randomized parameters relating to time	Run complete at a leveled correlation between times where groups fall behind vs ahead of pace	Run show visible occurrence discrepancies	[Additional comments or observations]
Run 3	Display the correlation of occurences of marshal need vs total times taken for rounds in a run	This run was executed specifically with randomized parameters relating to time	Run requires more stress on marshal to be deployed	Run requires an average amount of marshal interferance	[Additional comments or observations]
Run 5	Display the correlation of occurences of marshal need vs total times taken for rounds in a run	This run was executed specifically with randomized parameters relating to time	Run requires less stress on marshal to be deployed	Round requires second to least amount of marshal interferance	[Additional comments or observations]
Run 7	Display the correlation of occurences of marshal need vs total times taken for rounds in a run	This run was executed specifically with randomized parameters relating to time	Run shows a lower rate of occurences of groups falling within pace	Runs show higher rate of groups falling within pace	[Additional comments or observations]
Run 9	Display the correlation of occurences of marshal need vs total times taken for rounds in a run	This run was executed specifically with randomized parameters relating to time	Run shows a higher rate of occurences of groups falling within pace	Runs show higher rate of groups falling within pace	...

Instructions for Using the Table

Run ID: Assign a unique identifier for each simulation run.

Objective: Briefly explain what each run is testing.

Parameters/Configurations: List any parameters or configurations that are used for this run.

Expected Outcome: State what you predict will happen based on the settings.

Actual Outcome: Record the results of the run.

Notes/Comments: Include any additional observations or issues that arose during the simulation.

Figure 7: Simulation Run Table

4 Sensitivity and Scenario Analysis

4.1 Sensitivity Analysis

The sensitivity analysis provides the reader with context of how different variables found within a program can alter the state of the program in such a way that output results change or remain unchanged.
The best ways to analyze the data here would be a combination of group variations. The methodology here would be to take all of the data polls here as groups of variable edits. So, global sensitivity is taken into account above all due to there being many factors within a pace of play system that would prohibit or restrict on pace play.

4.1.1 Parameters

- Env time increased to 700, larger tournament size, behind pace marshal call increased to 10 minutes:– 8 Runs total
- Return gap decreased to 5 minutes, start delay increased to 15 mins, env.timeout increased to 8 minutes:– 8 Runs total
- Hole number rearrangement(start on 10 instead of 1):– 8 Runs total
- Decrease in start delay to 8, time limit to deploy marshal to 8, decreasing env.timeout to 3:– 8 Runs total

4.1.2 Results

One of the common measurements of data that was used throughout the process was the number of times that the marshal was utilized to speed players on the course up who were playing slow. This data point with the new data runs compared to that of the simulation run is far in several runs compared to the several runs that have been conducted with the parameters set to normal values. There were far less needs for the marshal in these runs, most likely due to the fact that the marshal's time threshold had been increased nearly double over what it had been previously. Another aspect of the data that was heavily analyzed was the number of times that players fell ahead of, behind, or within pace. These metrics, in the data tables, were all measured separately, however, they work hand in hand with one another to show the patterns of time on the course. A common observation with this data was that the runs all fell within ballpark of one another in the instance where the return gap was decreased, the start delay was increased, and the timeout was increased and these runs also com pared well to the other simulation data sets as well. One outstanding difference in the data that was found was that in the study done where the holes were rearranged, the number of groups that stayed within pace significantly dropped. This could be a pitfall within the data where the simulation may have malfunctioned.

4.2 Scenario Analysis

In the testing phase of this section, a thorough analysis of 8 runs per scenario was done in order to capture the nature of 8 separate tournament situations across the board. The runs provided display results on the anticipated long tournament, the anticipated short tournament, the shotgun start on hole 10, and the quicker sendoff.

4.2.1 Parameters

- Long tournament day lasting roughly 11.5 hours
  - Env time = 700 in minutes

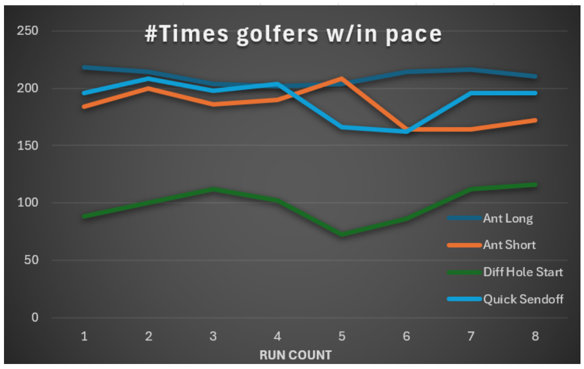


Figure 8: Comparing Scenarios and the number of times golfers fell within pace for 8 tournaments

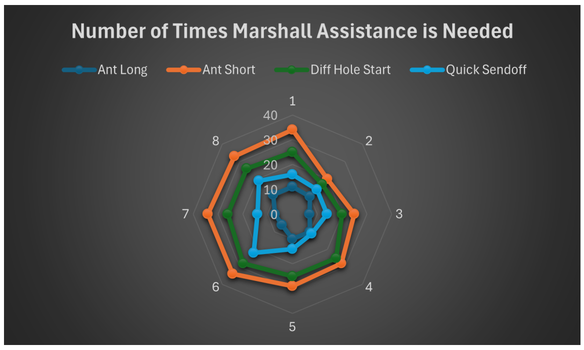


Figure 9: Comparing Scenarios and the number of times the marshals assistance is needed throughout each run

– ”tee times” = list(range(1, 32))- Main value here to look at is the Y value so the top of the range.
– Time limit to deploy marshal set to 10 minutes

- Smaller tournament day lasting roughly 8.5 hours
  – return gap = 5 minutes
  – group start delay = 15 minutes
  – env.timeout = 8 mins

- Tournament starting on hole 10 instead of hole 1
  – Rearrange course array to fit test

- Quicker start day
  – group start delay = 8 minutes
  – time limit to deploy marshal = 8 minutes
  – env.timeout = 3 minutes

4.2.2 Results

As we look at the data and the graphs and study the differences in the data from the first simulation runs and the now tweaked simulation runs to account for the analyses, there are many good points of data research that have come out of this study. For one, the variables that were explained earlier within the document, all cooperate with one another when studying sensitivity and scenarios

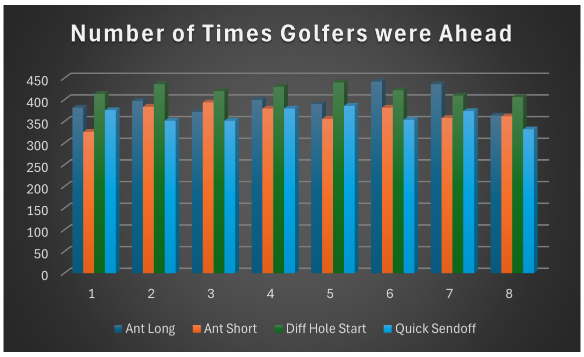


Figure 10: Comparing Scenarios and the number of times the groups were ahead of pace by at least a minute

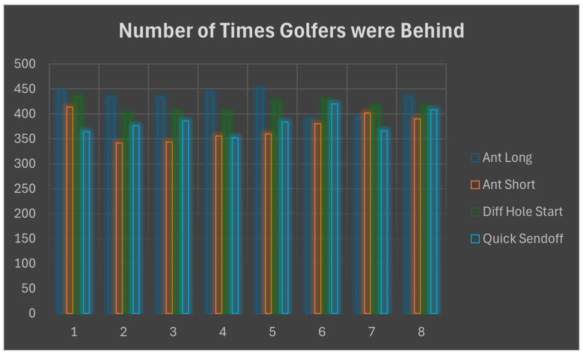


Figure 11: Comparing Scenarios and the number of times the groups were behind pace by at least a minute

specifically, the four scenarios that have been laid out. In many cases, there is a big talk about how to speed people up in an effective manner to where the golf course not only makes their money faster, but the groups that play have an enjoyable experience and so as a result in this program and study there have been many data points that have been picked up on by myself that I can confidently pinpoint as issues that slow down the pace of play on a golf course. For one, the pace of play threshold. This metric of time has been used for years and while not always being at 5 minutes like my own standard metric, it is still quite low for many courses that most definitely should have a higher time threshold. One thing from the data that I my self learned is that just because you rush groups onto the course for the tournament, does not necessarily mean you are going to run less of a risk of burning valuable time (also money) on the golf course for other non tournament playing golfers. Most of the time in cases like these, your cart turnover is going to be significantly slower compared to having a more structured way of sending groups out to play. Another factor that I can confidently say influences the time that it takes a golf course to complete a tournament is the number of groups per tournament. One major factor in a large tournament is always going to be slow play solely because there are just so many people out on the course at a given moment. There are most definitely pitfalls in this study, one being the overall formatting of the visuals needed to display the points, the overall errant behavior in a few outlying pieces of data but overall of the data that has been reliably and easily retrieved, I can firmly say that the objectives that were tackled in the data study itself yielded the results that sought.

5 Validation and Verification Analysis

5.1 Validation Methodology and Results

The validation of this golf course pace of play simulation model is crucial to provide context from the golf world to better reinforce the nature of the model by making direct correlations to real world examples and structural system standards set within industry.

5.1.1 Parameter Validation

One very important parameter to consider among the golf course pace of play model is the time range that is taken per par standard per hole on the golf course. In this model implementation, a set time standard of 12 minutes for a par 3, 14 minutes for a par 4, and 18 minutes for a par 5 are all created as parameters that the simulated groups are aiming to meet in terms of time based on the hole they are playing. The time range specifically granted to each of the pars where the golfers would randomly complete a hole in those times were for a par 3 ten to fourteen minutes, for a par 4 12 to 16 minutes, and for a par 5 sixteen to 20 minutes. These time configurations were all engendered based off of empirical research done by myself in a personal interview with a local golf course owner and PGA Professional Bryan Raines, where a complete list of tee times and time slots were provided. Based on my own knowledge I provided a range in accordance to this hard data provided with a gap of two minutes on either side of an average standard time needed to complete a par type of a hole on course.

In a tournament setting it is no strange concept that groups will take much longer to complete their rounds based on the conditions of play and money at stake. A prime example of this is the findings of Andrew Tiger and Eric Ellerbrook in their literature review titled, " Improving Golf Pace of Play Using Time Study Analysis: Influencing Factors on the Green and Tee Box" where they include stats based on how time is affected by a competitive setting here; "Competition added 68 seconds to the time on each green." and later concluding that, " Of the significant variables, Competition increases the amount of the time on the green the most compared with the other significant variables."

Another example of literature that specifically mentions the use of time standards is an article titled,"How long should it take to play a par 3, par 4 and par 5 during a round of golf?" where a time standard here on average is calculated at about 10 minutes for a par 3, 13 minutes for a par 4, and around 15 minutes for a par 5. This data is a bit more accurate than the data in this simulation model because it accounts for data based upon more then just one golf course and its holes therefore even though my assumptions of a standard time per par type are mighty close to these standards, they are still slightly off.

5.1.2 Cross-Model Validation

A model study done by Qi Fu and Ward Whitt stochastically examines the golf course pace model and how the pace model is directly affected in the nature of a queuing system. If there is a faulty queuing system in place, odds are that the pace of play model will be greatly effected. A large talking point within their research, much like my own, was discussion of a stage based queuing system where a group tees off and its not properly allowed to continue their round until each individual step is completed. An example of this is when group 1 reaches the fairway for their ball and are hitting from fairway to green. If a fairway is open in certain spots, then the ability in the real world to proceed upon open area of land is available. This model would cut that entirely, as does my own. When a group hits the fairway from the tee box, until they make contact with the green the members of the previous group will have no ability to complete their tee shots. Once green contact is made however, the next group may proceed to the fairway of that hole. The main point of partaking in this methodology is to overall add structure to the system in a way that there is not necessary back up in places there should not be.
The authors of this article found much success in shifting focus away from the overall time structure for the course and provided a mathematical queuing structure that dives deeper into the individualistic characteristics of a golf hole.

5.1.3 Face Validation

This models implementation and simulation execution accounted for much of the data gathered by experts in its early conception. Now however, the main accounts that should be made should be that of comparing the model to the real world data that can be provided by an expert themselves.

In research, I was able to reach out to Bryan Raines, a licensed PGA Professional and golf course owner to get some insight into how a normal day on course goes just to observe how the pace of the course takes route and also how the pace for the day is manually manipulated to better fit a courses queue.

5.1.4 Sensitivity Analysis Callback

From open, the courses first tee time on the given sample normal day is at 8am and the final for that day is around 1:54pm this would put the days first groups round end time at around 12:19pm and the last groups end time at around 6:13pm. This would put each groups total time range to complete at 4 to 4 and a half hours. Based on my bounds with 26 groups the start and end time stamps between my own data results and that of the hard data provided to me in the Face Validation, the times largely line up with one another somewhat similarly. If I change the time start gap between tee-off to 8 minutes, the results change significantly down to the mid 400s to high 400s in minutes, therefore bringing the tournament time end closer to that of a much more efficient time. This accurately represents real world data as well as a locla course near me implements this same principle in its time start gaps per group.

Based on an average of three runs in my own simulation model, the margin of error between my own model and that of the real world data was just within plus or minus 2 percent at around plus or minus 1.7 percent. Each run upon changing the sendoff gap changed minutely and remained around the same time gap as the real world data based on 26 groups. I believe that this is an acceptable margin specifically because it only represents an 8 minute differential between real world data and simulation data.

Average Standard Time Calculation based on 3 runs:
(465+460+476) minutes/3 runs = 467 minutes
Margin of Error:
(467-459) minutes/459 minutes = 0.01743 = 1.7% error

5.2 Verification Methodology and Results

Verification is used here to ensure that real world functionalities of a golf course are properly implemented in the given simulation environment. Verification will provide key insights into how the model will react based on an individual component-based level.

5.2.1 Unit Testing

In Unit Testing, I came to a large crossroads in the actual execution of the testing itself. The first scenario that I had planned on testing was the hole classification function titled get_standard_time(). After countless tries at debugging I was unable to get the individual unit working. Above you will find the snippet of code used to test this function alone in figure 4.

5.2.2 Integration Testing

Within integration testing, I was able to test the interactions between separate functions within a run of the entire code base. I chose to initially study the interactions of the detect slow, and send marshal functions in the course operator class to see how the interaction of the two with no outside influence would work with one another. Outside of this with its issues, the approach that I took to do this was to study the interactions between the methods within the output of the overall entire code base within the console. The reason I opted for this is because much like the Unit Testing, I ran into problems where even loading the functions into the unittest environment was nonfunctional no matter what I tweaked or added.

5.2.3 Regression Testing

In rerunning the code base to check for changes in results after modifying the time gap parameter, I can confirm that the results of the program run almost identically to the results of the previous program run with the exception of total time period of a tournament on course.

5.2.4 Code Review

The primary issue that was found within my program were a couple of inefficiencies such as the actual array list of holes itself where the groups are queued. This specifically being a hard coded value restricts the customization and adaptability of the program that essentially would provide the ability for it to be used based on a multitude of different course layouts as opposed to just the one that it is modeled after.

Most other warnings that were found within the code base were warnings relating to the data.py file that gathers data and puts it into a study using the Reg Ex package to gather data based on keywords. These warnings were mainly petty warnings regarding whitespace issues in the list of files that data was gathered from and an issue with the end_time variable regarding type. Although these warnings are present, the code still performs as needed and work efficiently for its designated purpose.

5.2.5 Results

Overall, the results were not entirely fruitful, but also not entirely a failure either. I came to some crossroads in efforts to conduct adequate Unit Tests and Integration Results, however there was fruit in regression testing where I found that the code has not changed for the worst as time has progressed, and I also successfully conducted a code review of all of the code base and was able to bring to light the advantages and inadequacies in the sim model.

Overall in unit testing, I would have farther tested the golf hole classification function for standard par time to check to see what the function would return given a user input of what hole they would like information on. Also within integration testing, I would have tested the contents of the course_operator class by providing alternate parameters for the actual wait time and par standard times variable to more adequately represent the use case of the detect_slow and the send_marshal functions in conjunction with one another.

6 Overall Analysis and Discussion

Through the course of the semester, a significant amount of development and analysis went into this project to contribute to its overall purpose as a whole. In totality, the results of the multiple different runs and tests provided give insight into how this model behaves in nature and how closely it is able to match up to the likeness of real world modeling data. Through the project, there have been baseline run/test/analysis metrics that have been used to describe the overall functionality and practicality of the model.

Namely, these metrics are as follows:

- Simulation and Data Runs:
  Within the Simulation Runs, I thoroughly observed the nature of the model by running the simulation associated to gather data results based on specific given types of data needed to measure the models purpose. In gathering this data I focused on three main scenarios being the number of times that a marshal is sent out to the golf course for deployment, the amount of time that it takes a group to finish their rounds from starting time stamp to finish, and finally the number of times that groups fall ahead of, behind, or within pace. In studying this data, I found that in more adolescent versions of the program, the groups were finishing in a much longer time scope per round and each round was contributing to a long day of golf for those in the tournament. Upon further changes in the semester, I came to the conclusion that the main factor influencing that time stretch was the time start gap per tee time where one group would go off and another would take that time period to wait to tee off on the first hole. Since the time gap was at a higher value and the rest of the models attributes static, there were many more times consequentially that groups went over the pace threshold as opposed to under it. Initially, this value was at 10 minutes per gap and was producing very high run timestamp finishes for the groups. Upon changing this variable, leaving the rest of the simulation to its native values yielded a much quicker result and therefore matched up much more closely to real world data.
- Sensitivity and Scenario Analysis:
  In the Sensitivity and Scenario analysis, I altered much of the parameter and variables within the code to account for group characteristic testing and also scenario testing in the scenario

analyses. In Studying this data, I found that one of the most crucial parts of the simulation in terms of the sensitivity analysis was the start delay being decreased to 8 minutes, the time threshold for a marshal need increasing to 8 minutes from 5 and the hard coded environment timeout being decreased to 3 from 5. These results yielded a much quicker and much more effective run with the compilation of rounds played by the 26 groups. In testing one of the most important things to note was that the groups that fell behind pace by the systems metrics were judged much less harshly compared to those running in the simulation where the marshal interference threshold was around 5 minutes. A correlation that was found was that if you disengage the marshal slightly compared to the base run, you send groups off in a way that is quicker in nature compared to the base run and you utilize a smaller time out period in the environment, the tournament overall, passes in not only a faster manner, but actually a more efficient manner with less marshal stress due to that threshold being higher.

- Validation and Verification:
  For the Validation and Verification Section of the project, I was able to perform validation on the parameter level, the cross-model level and the face level. On the parameter validation section, I was able to validate the standard times per par on a golf course using sample data gathered from a local golf course and a local owner, Bryan Raines where a data sheet was provided to me to accurately analyze and calculate average total times to complete a round. On top of the information that I received here, I was also able to conduct literature reviews that established the points that were needed to be made as well. Another form of validation that was used to test the models usefulness was the cross-model validation where I compared my model to another model created and established by a professional within the field. The model in question is a study done by Qi Fu and Ward Whitt that stochastically examines the golf course pace model and how the pace model is directly affected in the nature of a queuing system. When a group hits the fairway from the tee box and they make contact with the green the members of the previous group will have no ability to complete their tee shots. Once green contact is made however, the next group may proceed to the fairway of that hole. This model implemented by Fu and Whitt closely relates to my model with the exception of expressive medium. Their model resides based in mathematic proofs, whereas my model resides in programmatic execution.
  Throughout the S&S Analysis, I was able to deduce that the given error for my model in comparison to the numbers that I was receiving before shrunk drastically. Before when I had done error calculations I had received close to 4 percent error, after tweaking the start gap time, I was able to get the overall error down to less than 2 percent. Please reference the Sensitivity Analysis Callback section.
  In verification; unit testing, integration testing, regression testing, and code reviews were all conducted to fit the purpose of the project. The main conclusions that I came to had been bleak in regards to the individualized unit testing and integration testing, however, I was able to draw conclusive studies from the regression testing and the code review. In regression testing, I found that there were not any specific structural effects that took place when rerunning the code base with edits, mainly output results in variables such as time. As mentioned previously in the code review section, the main issues that were found there were primarily in the hard coded holes array list and it being static as opposed to dynamic in nature. The rest of the issues that were "holding back" the simulation were simple warnings that were not necessarily affecting anything important in the model. Up to that point I had a good, sturdy model that simulated results that similarly replicated real world pace management efforts.

7 Conclusion

Overall, throughout the semester, this Golf Tournament Pace of Play Simulation model has evolved very nicely with lots of developmental changes throughout the code base itself. There have been many crucial milestones that have been associated with the project. At the beginning of the semester, a project proposal, where I, the student, was responsible for proposing a project idea is submitted to the professor. Then, background research with literature reviews and design development with UML diagrams took place in Milestones 2 and 3. Next came Milestone 4, the model implementation where a bare-bones iteration of a golf course simulation model was developed and kept track of using source control. Milestone 5 consisted of a test run of the program where the base program and its purposeful functionalities were tested. Milestone 6 was a Sensitivity and Scenario Analysis where key parameter variables were altered to better fit other situations as benchmarks for how the program would work outside of the baseline scope attached to the first implementation iteration. The 7th milestone, Validation and Verification served as a checkpoint and baseline to ensure functionality and accuracy. Finally, this Final Report, the 8th milestone, where a conglomeration of elements from all of the different valuable milestones are collectively grouped and presented in a final package.

In this final report and throughout the semester, I have learned and found much value in this simulation model. In developing the model I have been able to make clear connections with the real world by way of studying it via the milestone track that has been provided. Some of the most important findings that I have recognized through the semester in regard to the project itself are that time itself is unwavering and the amount of time that you have to complete functional tasks as time consuming as a golf tournament becomes infinitely more limited as inefficiencies in queuing systems of golf courses become more diminished. One of the main purposes of this project was to create a tool that a user could use to replicate the tournament setting to build back up their diminished systems of queuing golfers in a tournament.

In the future, I plan to further develop this product to fit a more wide range of a population of users and allow the model to study much more than just that of a tournament setting. I am aiming to have this project be capable of calculating time frames after the tournaments are over to account for how many normal, regular paying golfers can go out and enjoy a round afterwards. This is simply going to serve as a tool to study cashflow and how to manage time as efficiently as possible. A goal for the project from here forward is to make further revisions to allow for user input for currently hard coded variables and also to develop a more modular product with many more professionally produced features. A feature that is more standard than anything that I would like to add is a user interface that can be well interacted with by the professional most likely to be developed in HTML and JavaScript. From the user interface, I would like to start adding back-end features such as a customization hole configuration feature to allow for more flexibility in the bounds of what the model can do on a foundational level, a customizable par standard time configuration and many other user input based customization options that will add overall quality to the model.

To conclude, throughout the semester, I have learned a significant amount about how development not only in a static programming sense works, but dynamically in nature as well. I have learned much more about how version control operates, about how the professional workplace in the field operates, how showcases relating to a project in a professional setting operate, and finally how to utilize tools to the best of your ability to come up with a product that at least can serve the purpose of the greater good in some way or another.

References

- W3schools.com. W3Schools Online Web Tutorials. (n.d.). https://www.w3schools.com/python/python_regex.asp
- Ballengee, R. (2024, February 6). How long should it take to play a par 3, par 4 and Par 5 during a round of golf?. Golf News Net. https://thegolfnewsnet.com/ryan_ballengee/2024/02/06/how-long-should-it-take-to-play-a-par-3-par-4-and-par-5-during-a-round-of-golf-125921/
- Models to study the pace of play in golf. (n.d.-a). http://www.columbia.edu/ ww2040/Fu_Whitt_Golf _110613.pdf
- Improving golf pace of play using time study analysis. (n.d.-a). https://www.golfsciencejournal.org/api /v1/articles/4987-improving-golf-pace-of-play-using-time-study-analysis-influencing-factors-on-the-green-and-tee-box.pdf
- Bryan Raines at Ashton Hills Golf Club, Covington, GA
- Leahy, D. O. (n.d.). Pace of Play in Golf. Fisher Digital Publications. https://fisherpub.sjf.edu/sport_undergrad/3
- Fu, Q., Whitt, W. (2015, January 1). Analyzing the pace of play in golf. Journal of Sports Analytics. https://content.iospress.com/articles/journal-of-sports-analytics/jsa0004
- Using simulation to help manage the pace of play in golf. (n.d.). http://www .columbia.edu/ ww2040/GolfManaging042917.pdf
- Miller, A., Raines, B. (n.d.). An Interview with Bryan Raines of AHGC. personal.

7.1 Acknowledgment

I'd like to thank Professor Christopher Regan for believing in this project and committing time to helping me reach my goals in an effective manner throughout the course. This project has been nothing short of enlightening and I would just like to share my appreciation in making it a well rounded and pleasant experience.