Stevens Institute of Technology

SYS-611 Homework #1

Due Sep. 14 2022

Submit the following using the online submission system: 1) Cover sheet with name, date, and collaborators, 2) Written responses in PDF format, 3) All work (e.g. .xlsx or .py files).

1.1 Systems Modeling and Simulation [5 points]

Imagine Stevens is considering constructing a new academic building on the corner of 5th Street and Sinatra Drive. A new building would alleviate space constraints on research and educational activities; however, it would also add significant debt and interest payments to the budget. You have been contracted to study this problem by the Board of Trustees. Figure 1.1 shows several approaches to study a system that you are considering.

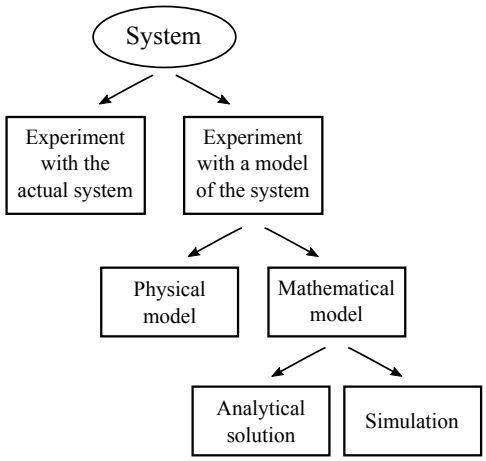

Figure 1.1: Ways to study a system (from Law and Kelton, 2000, p. 4).

(a) 2 pts Identify a part of the problem that could be studied using an example of *non-simulation* analysis (actual system, physical model, or analytical model) and explain why.

(b) 2 pts Identify a part of the problem that could be studied using an example of *simulation* analysis and explain why.

(c) 1 pt For your simulation analysis example given above, describe whether it should be static/dynamic and deterministic/stochastic and explain why.

1.2 Interacting with Model State [8 points]

This question builds on the Tic-Tac-Toe model developed in class.

(a) 2 pts Using words, explain the logic that determines whether a Tic-Tac-Toe model state contains a winner.

(b) 2 pts Implement the Tic-Tac-Toe `get_winner` derived state function in a computational modeling tool (e.g. Python or Excel). Inspect the model state and display the string `"x"` or `"o"` if there is a winner, otherwise display a blank string `""`.

(c) 2 PTS Using words, explain the logic that determines whether a Tic-Tac-Toe model state contains a tie.

(d) 2 PTS Implement the Tic-Tac-Toe `is_tie` derived state function in a modeling tool (e.g. Python or Excel). Inspect the model state and display a boolean `True` or `False`.

1.3 Modeling a Board Game as a Simulation [12 points]

Chess is a very old board game described as[1]:

> a two-player, abstract strategy board game that represents medieval warfare on an 8x8 board with alternating light and dark squares. Opposing pieces, traditionally designated White and Black, are initially lined up on either side. Each type of piece has a unique form of movement and capturing occurs when a piece, via its movement, occupies the square of an opposing piece. Players take turns moving one of their pieces in an attempt to capture, attack, defend, or develop their positions. Chess games can end in checkmate, resignation, or one of several types of draws.

(a) 3 PTS What is the elementary model state for chess? Describe it in sufficient detail to allow someone to create a mathematical (symbolic) model. (*hint: what information is required to "save" a game and later recreate it? It may help to reference an image.*)

(b) 4 PTS Using words, explain the logic that determines the following derived state functions for chess in terms of the elementary state in (a):

(i) `get_num_pieces` which counts the number of remaining pieces owned by a player.

(ii) `get_valid_moves_rook` which lists cells to which a particular rook could move.

(iii) `is_check` which determines if the current player is in "check" (king under attack).

(iv) `is_checkmate` which determines if the current player has lost.

(c) 3 PTS Using words, describe how following state changes modify the elementary state in terms of variables in (a):

(i) `capture_piece`: Captures one of the opponent's pieces

(ii) `move_piece`: Moves one's own piece to a new cell

(iii) `promote_pawn`: Promote a pawn that has reached the last row (you may need to research this rule)

(d) 2 PTS Draw an activity diagram to show the chess game progression from start to end with the following *simplified set* of actions: `choose_valid_move`, `capture_piece`, `move_piece`, `next_player_turn` and derived state variables: `is_move_capture` (checks if a move results in a capture), `is_checkmate`

[1] `https://boardgamegeek.com/boardgame/171/chess`

