6

Simple state transitions

Concept. To define and develop an archetype simulation environment for simple state transitions. In our classification scheme, this archetype is defined by a trivial state partition graph topology and would make sense for simulations of sequential design problems, sports matches and other simple gameplay domains. We will also discuss the typical ways in which the state of the system may only partially be observed in realistic examples, and analyse how best to deal with each situation. For the mathematically-inclined, this chapter will define the mapping of our formalism to simple state transitions. For the programmers, the software which is designed and described in this chapter can be found in the public Git respository here: https://github.com/worldsoop/worldsoop.

6.1 Defining the archetype

The simple state transition archetype refers to simulation environments where there is no obvious computational benefit to partitioning the state into concurrently updating or acting on separate components. There may even be performance benefits from keeping state information all within the same common data structure in memory, but this can depend on the specific problem of study.

In the interest of completeness with respect to the chapters which follow on from this one, we have illustrated the trivial state partition graph topology for this archetype in Fig. 6.1.

In order to understand what sorts of data might be collected about this archetype in realistic scenarios, let's begin by considering the types of real-world problem domain which have leveraged simulation environments to train control algorithms in the literature. The subset of these which may best suit the simple state transition environment archetype are:

- Event-based simulations of sports matches, e.g., football [1], rugby [2], tennis [3], etc., and other forms of game — all of which typically define a relatively simple global match/gameplay context as their 'state'.

- Sequential design simulations to change the configuration or data collection strategies of, e.g., astronomical telescopes [4, 5], biological experiments [6] and user interfaces [7], which typically define a relatively simple and finite set of possible actions that can be taken.

State
Partition
(Trivial Topology)

Figure 6.1: State partition graph topology for simple state transition archetypes.

Based on the examples given above, what kinds of data may be available to infer the state and parameters of a simulation environment using this archetype?

In the case of team sports matches, a team manager could have access to a large body of data which assesses the capabilities of their players before a game, but they must rely on more subjective or limited data to assess the performance of their team (and the opposition) in the middle of a match. The 'state' of the whole system in this case can be not only the state of play, but also information about, e.g., fatigue or on-the-day performance of each player for both teams and sets of substitutes.

Talk about sequential design data here...

In a general offline learning setting, realistic historical data collected about the state values for this archetype will typically provide an important mechanism for model determination. Users of this archetype in the real world also typically have access to independent datasets that can be used offline to better determine some parameters of the simulation environment which are not observed directly. In the online learning setting, one typically directly observes some portion of the state values, and may only indirectly observe the other state values, if at all.

6.2 Writing the code

Bibliography

[1] M. Pulis and J. Bajada, “Reinforcement learning for football player decision making analysis,” in *StatsBomb Conference*, 2022.

[2] T. Sawczuk, A. Palczewska, and B. Jones, “Markov decision processes with contextual nodes as a method of assessing attacking player performance in rugby league,” in *Advances in Computational Intelligence Systems: Contributions Presented at the 20th UK Workshop on Computational Intelligence, September 8-10, 2021, Aberystwyth, Wales, UK 20.* Springer, 2022, pp. 251–263.

[3] N. Ding, K. Takeda, and K. Fujii, “Deep reinforcement learning in a racket sport for player evaluation with technical and tactical contexts,” *IEEE Access*, vol. 10, pp. 54 764–54 772, 2022.

[4] P. Jia, Q. Jia, T. Jiang, and J. Liu, “Observation strategy optimization for distributed telescope arrays with deep reinforcement learning,” *The Astronomical Journal*, vol. 165, no. 6, p. 233, 2023.

[5] S. Yatawatta and I. M. Avruch, “Deep reinforcement learning for smart calibration of radio telescopes,” *Monthly Notices of the Royal Astronomical Society*, vol. 505, no. 2, pp. 2141–2150, 2021.

[6] N. J. Treloar, N. Braniff, B. Ingalls, and C. P. Barnes, “Deep reinforcement learning for optimal experimental design in biology,” *PLOS Computational Biology*, vol. 18, no. 11, p. e1010695, 2022.

[7] J. D. Lomas, J. Forlizzi, N. Poonwala, N. Patel, S. Shodhan, K. Patel, K. Koedinger, and E. Brunskill, “Interface design optimization as a multi-armed bandit problem,” in *Proceedings of the 2016 CHI conference on human factors in computing systems*, 2016, pp. 4142–4153.