Optimization and Decision Making

Exercise Sheet 7

Exercise 7.1 (Case Study: Automatic Game Balancing)

Figure 1: Exemplary *Top Trumps* card (taken from [1]).

Top Trumps[1] is a themed card game where each player's goal is winning as many card comparisons (called *tricks*) as possible. An exemplary card is shown in Figure 1. As setup, a deck of K cards is shuffled and evenly divided between the players s.t. each player can only see the top card of their own hand. A random player is chosen as initial starting player. Here, we consider the two-player variant and assume K is even.
Each card contains values for $L \in \mathbb{N}$ different categories, out of which the starting player chooses one. Then, the current cards of all players are compared, and the player with the highest value in the chosen category wins the trick. The winning player adds all cards to the end of their hand and resumes as the starting player. In case of a tie, each player keeps their own card (and moves it to the back of their hand) and the current starting player continues with the next card.
When balancing a game like *Top Trumps*, different objectives may be pursued, such as the *fairness* and *excitement* of the resulting gameplay. Following [1], we model the deck as a real-valued vector $x \in \mathbb{R}^{KL}$ containing

[1] In German, the game is also known as *Autoquartett*.

each card's values and utilize two agents, p_0 (which is less skilled) and p_4 (which is more skilled) to simulate gameplay[2]. We then measure the following objectives by performing R simulations of the first $K/2$ tricks such that each card was played exactly once:

$$\underset{x \in \mathbb{R}^{KL}}{\text{maximize}} \quad F(x) := (f_{Fairness}(x), f_{Excitement}(x)), \tag{1}$$

$$f_{Fairness}(x) = \frac{1}{R} \sum_{r=1}^{R} \mathbb{1}\left(t_4^{(r,x)} > \frac{K}{4}\right), \tag{2}$$

$$f_{Excitement}(x) = \frac{1}{R} \sum_{r=1}^{R} t_c^{(r,x)}, \tag{3}$$

where $t_4^{(r,x)}$ is the number of tricks performed by player p_4 and $t_c^{(r,x)}$ is the number of trick changes (i.e., changes in leading player) in repetition r with deck x. As the game should reward skill, $f_{Fairness}(x)$ measures the win rate of the more skilled player. To express excitement, $f_{Excitement}(x)$ measures the average number of trick changes, i.e., changes in starting player.

An implementation of the game and how to connect it to `pymoo` is provided in a Jupyter notebook in the LearnWeb course. You don't have to use this implementation (though we recommend you do), and you can also use other libraries or your own code to perform the optimization.

It is now your turn to automatically design *Top Trumps* decks that have high fairness and excitement! Your main tasks are:

- **Approximate the Pareto front** on the given problem for $K = 22$ cards and $L = 4$ categories. Use $R = 1\,000$ simulation runs to evaluate each proposed solution / deck.
- Evaluate the quality of your approximation using a suited quality indicator.
- Investigate the resulting solutions in the original game's context: Are there recognizable patterns which kinds of decks do have better scores in one or the other objective?

[2] The notation of the agents was chosen to match the notation in [1].

If you want to dive deeper, you can choose to investigate one or more of the following *optional* tasks:

- Evaluate different optimizers and discuss performance advantages / drawbacks of the individual approaches.
- Investigate the impact of changing the K and L parameters on the attainable Pareto front.
- To what extent does the number of repetitions R influence solution quality?
- Play / test one or more of your automatically designed games and discuss whether the objectives capture your own experiences while playing the game.
- Implement other or additional objectives (the paper [1] already includes a third one) and investigate the influences on the game design.
- Since the properties are modeled as real-valued parameters, ties are unlikely to occur. How can you adapt the game and optimization task to induce some ties?
- Which properties does the problem landscape have? Which landscape properties can you capture for high-dimensional multi-objective problems?

Prepare your solutions in the form of an **A0 poster** and submit it as a PDF document in LearnWeb until **11:59PM, February 2nd, 2026**.

References

[1] Vanessa Volz, Günter Rudolph, and Boris Naujoks. "Demonstrating the Feasibility of Automatic Game Balancing". In: *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. GECCO '16. Denver, Colorado, USA: Association for Computing Machinery, 2016, pp. 269–276. ISBN: 9781450342063. DOI: 10.1145/2908812.2908913.