Oneshot negotiations in Colored Trails

Harmen de Weerd[1,2], Bart Verheij[1], Rineke Verbrugge[1]

[1] Institute of Artificial Intelligence, Faculty of Science of Engineering, University of Groningen, The Netherlands
[2] Research group User-Centered Design, School of Communication, Media & IT, Hanze University of Applied Sciences, The Netherlands

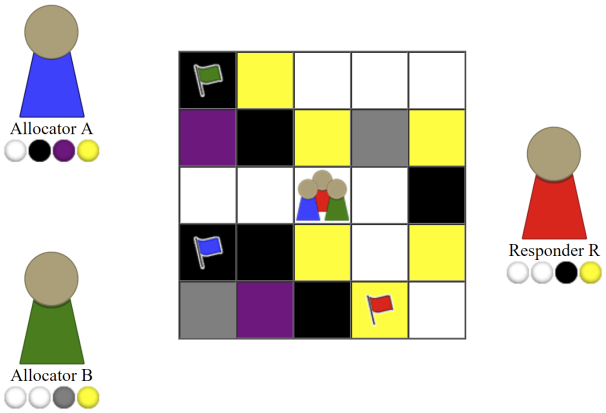

Figure 1: Example setup of the simulation.

1 Short model description

This is a model of agents engages in Colored Trails negotiation [1]. The game is played on a 5 by 5 board of colored tiles, drawn from 5 possible colors, as shown in Figure 1. Three agents each start out on the center tile, and aim to reach a given goal location, indicated by colored flags in Figure 1. To move around on the board, agents have a set of 4 colored chips. Agents may move horizontally and vertically to adjacent tiles by handing in a chip of the same color as the destination tile. For example, the blue Allocator A agent in Figure 1 may move left once by handing in his white chip. However, he would then be unable to move left again, since he no longer holds a white chip.

Before agents start moving, they engage in a single round of negotiations, which is similar to the way described by Ficici & Pfeffer (2008). The two allocator agents (the blue and green agents in Figure 1) propose to exchange some of their own chips against some of the chips of the Responder R (red agent in Figure 1). For example, Allocator B may wish to offer to exchange a gray chip and a white chip against the black chip and the yellow chip of the responder. That way, Allocator B could reach his goal location by moving up, up, left, and left. Of course, Allocator B could also ask the responder to give him all of the responder's chips and offer no exchange.

The two allocator agents make their respective offers simultaneously. Both offers are then revealed to all agents, after which Responder R chooses whether to accept one of the two offers. Note that the responder

[1] Also see `http://coloredtrails.atlassian.net/wiki/display/coloredtrailshome/`.

cannot accept both offers, but may choose to reject both offers. If the responder accepts an offer, the exchange is made. Then negotiations end, the agents move around on the board, and the game ends. Agents are scored on how close they reach their goal location. For each step they take towards the goal location, agents receive 100 points. For reaching their own goal location, an agent receives an additional 500 points. Finally, if an agent owns a chip at the end of the game (i.e. the agent has not used the chip to move on the board), the chip is worth an additional 50 points.

2 Program description

The program code allows for agents of different orders of theory of mind (see De Weerd et al. 2017, for a model example) to play Colored Trails against one another. The sections below give additional information. In the class `tester/Tester1Shot.java`, a sample program is available.

2.1 Setting - oneShot/Setting1Shot.java

The setting describes the board, the initial set of chips, and the goal locations of agents. Note, however, that none of this information is available to the agents. Instead, agent make use of utility functions that specify what actions can be taken by what agents, and what (utility) result these actions would have. To generate a random scenario, use:

```
Setting1Shot setting = new Setting1Shot();
setting.generateSetting(boardSize, nrColors, nrPlayers, nrChipsPerPlayer);
```

The recommended settings are `boardSize = 5, nrColors = 5, nrPlayers = 2` and `nrChipsPerPlayer = 4`. Note that generating a random setting does not generate random goal location for players. In addition, a randomly generated setting does not guarantee a good negotiation setting. For example, Allocator B would not be able to negotiate with Responder R in Figure 1. That is, there is no offer that Allocator B can make to increase the scores of both Allocator B and Responder R (exercise left to the reader).

2.2 Agent

The basic outline of an allocator player is described in the abstract class `oneShot/Player1Shot.java`. It implements everything except `makeOffer` and `selectOffer`, the functions that determine what offers maximize the expected utility, and what offer to make as a result.

Note that agents may have a variety of belief structures, that determine what the basis is of forming beliefs. The belief structures model is not fully fleshed out, and the abstract class `oneShot/BeliefStructure` needs some updating to actually be useful. The model described in (De Weerd et al. 2017) corresponds to `oneShot/Structure25`.

Theory of mind agents are constructed with the class `oneShot/PlayerToM1Shot`, using:

```
int playerID = 0;
PlayerToM1Shot player = new PlayerToM1Shot(playerID, ToMorder, setting, nrColors,
    nrChipsPerPlayer);
```

The `playerID` refers to the ID of the agent within the setting `setting`. Before asking an agent to make an offer, the agent should be told what chips it has through

```
player.init(setting.allocatorChipset[playerID], setting.allocatorChipset[1 - playerID]);
```

This tells the agent what chips it has itself, and the chips owned by the competing allocator. To make an offer, use:

```
playerOffer = player.makeOffer();
```

To reduce the time requirement of finding an offer to make, offers are represented as numbers. To get the number of chips of each color, use

```
Chips.getBins(playerOffer, setting.getBinMax(playerID));
```

Note that making an offer does not result in observing that offer. Similarly, offers made by other agents, as well as the final verdict of the responder, should be communicated to the agent through observation. For example, suppose that a player has made offer `playerOffer` and there is one competitor that has made offer `competitorOffer`. The responder decides to accept the player's offer. The resulting observations are:

```
    player.observeAction(playerID, playerOffer);
    player.observeAction(1 - playerID, competitorOffer);
    player.observeAcceptance(playerID, playerOffer);
    player.observeRejection(1 - playerID, competitorOffer);
```

References

Ficici, S. G. & Pfeffer, A. (2008). Modeling how humans reason about others with partial information. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, (pp. 315–322). IFAAMAS

de Weerd, H., Verbrugge, R. & Verheij, B. (2017). Negotiating with other minds: The role of recursive theory of mind in negotiation with incomplete information. *Autonomous Agents and Multi-Agent Systems*, *31*(2), 250–287. doi:10.1007/s10458-015-9317-1
URL `http://dx.doi.org/10.1007/s10458-015-9317-1`