1 Supplementary Material

This section contains supplementary materials, such as additional figures, technical details, or data.

Table 1: The set of parameters Θ used in the CAB and SChMUSR algorithm. Each parameter can take on an integer value in the range $[0, 100]$. Adjusting the parameters can produce vastly different behavior. The value column refers to the hand-coded value that was used in prior research done with the CAB agents. The * denotes parameters that are not found in the CAB agents, only SChMUSR agents

Symbol	Usage	Value
θ_{α}	Specifies the weight between positive and negative influence when computing	20
θ_{strength}	Specifies the agent's desired community strength as a percentage of the whole	70
θ_{modu}	Weight of modularity in scoring potential communities	100
θ_{target}	Weight of target collective strength in scoring potential communities	80
$\theta_{\text{prominent}}$	Weight of prominence in scoring potential communities	50
θ_{familiar}	Weight of familiarity in scoring potential communities	50
$\theta_{\text{prosocial}}$	Weight of prosocial in scoring potential communities	70
θ_{initialD}	Specifies the percentage of its tokens the agent keeps in round $\tau = 1$	20
θ_{dUpdate}	Specifies how quickly the agent adapts its keeping based on new attacks on itself	50
$\theta_{\text{dPropensity}}$	Specifies the degree to which the agent keeps tokens based on past attacks against it	50
θ_{fearD}	Specifies the degree to which the agent keeps tokens based on attacks on its friends	0
θ_{minD}	Specifies the minimum number of tokens the agent keeps in a round	20
θ_{pFury}	Percentage of tokens to use in a pillage attack	0
θ_{pDelay}	Number of rounds to wait at the start of the game before considering pillage	10
$\theta_{\text{pPriority}}$	The priority of pillage attacks	0
θ_{pMargin}	Determines the gain margin required before a particular pillage attack is considered	0
$\theta_{\text{pCompanion}}$	The amount of help the agent initially believes it will get in a pillage attack over the first 5 rounds	50
θ_{pFriends}	Determines whether an agent will considering pillaging a friend.	0
θ_{vMult}	Specifies how much vengeance to reciprocate	100
θ_{vMax}	Specifies the maximum number of tokens that can be used in a vengeance attack	100
$\theta_{\text{vPriority}}$	The priority of vengeance attacks	100
θ_{dfMult}	Determines how much to reciprocate in defend-friend attacks	100
θ_{dfMax}	Specifies the maximum number of tokens that can be used in a defend-friend attack	100
$\theta_{\text{dfPriority}}$	The priority of defend-friend attacks	90
$\theta_{\text{attackGGuys}}$	Determines whether the agent will consider defend-friend attacks against players that were not the first to attack and did not over-attack	0
$\theta_{\text{groupAware}}$	Determines whether the agent will consider defend-friend attacks against players that belong to communities that have more collective strength than its own.	0
θ_{Safety}	Specifies whether the agent will defend or attack if it cannot do both	0
$\theta_{\text{debtLimits}}$	Defines how much the agent will give to another player without reciprocation	25
$\theta_{\text{limitGive}}$	Sets limits how much the agent can give to a poor player	100
$\theta_{\text{fixedUsage}}$	Determines the percentage of tokens the agent gives uniformly to its group (as opposed to distributing based on past friendship)	50
$*\theta_{\text{trustRate}}$	Determines how quickly the agent builds trust	0
$*\theta_{\text{distrustRate}}$	Determines how quickly the agent loses trust	0
$*\theta_{\text{startingTrust}}$	Specifies the amount of trust the agent initially has for everyone	0
$*\theta_{\text{wChatAgreement}}$	Weight of the agreement of others in scoring potential communities	0
$*\theta_{\text{wTrust}}$	Weight of the trust of group members in scoring potential communities	0
$*\theta_{\text{wAccusations}}$	Weight of accusations in scoring potential communities	0
$*\theta_{\text{fearAggression}}$	How much the agent fears aggression in others	0
$*\theta_{\text{fearGrowth}}$	How much the agent fears growth in others	0
$*\theta_{\text{fearSize}}$	How much the agent fears larger groups	0
$*\theta_{\text{fearContagion}}$	How much the agent fears players that others fear	0
$*\theta_{\text{fearThreshold}}$	The threshold at which the agent's fear triggers a response	0
$*\theta_{\text{fearPriority}}$	The priority of attacking out of fear over other kinds of attacks	0

2 SChMUSR: Algorithmic Details

In this section, we provide algorithmic details for how fear is computed by SChMUSR and how it derives and uses chat messages.

2.1 Fear

SChMUSR's fear (Eq. **??** is based on four variables. In this subsection, we describe how $F_i^{\text{agg}}(t)$, $F_G^{\text{size}}(t)$, and $F_G^{\text{growth}}(t)$ are computed. Computation of $F_G^{\text{contagion}}(t)$ is described in the next subsection.

$F_i^{\text{agg}}(t)$ is calculated from the influence matrix of the JHG [5]. Let $\mathcal{I}_{i,j}(t)$ be the influence of player i on j in round t. To determine the amount of fear from aggression that comes from player i, we take the total negative influence (denoted $\mathcal{I}^-_{i,j}(t) = \max(0, -\mathcal{I}_{i,j}(t))$) player i has on others and divide it by the total influence player i has on others. Formally, the aggression value of player i is

$$F_i^{\text{agg}}(t) = \frac{\sum_j \mathcal{I}^-_{i,j}(t)}{\sum_j |\mathcal{I}_{i,j}(t)|}. \tag{1}$$

$F_G^{\text{agg}}(t)$ is the average of $F_i^{\text{agg}}(t)$ over all $i \in G$.

$F_G^{\text{size}}(t)$, which measures the fear a SChMUSR agent derives due to group G being more powerful than its own group, is calculated by comparing the collective popularity of group G to the agent's own perceived group G_{in}. Let $\mathcal{P}_G(t) = \sum_{j \in G} \mathcal{P}_j(t)$ be the collective popularity of group G in round t, where $\mathcal{P}_j(t)$ is the popularity of player j in round t. Then

$$F_G^{\text{size}}(t) = \max\left(0, \min(1, [\mathcal{P}_G(t)/\mathcal{P}_{G_{\text{in}}}(t)] - 1)\right). \tag{2}$$

Note that SChMUSR derives no fear from any group smaller than its own and treats any group larger than twice its size as if it were exactly twice its size.

$F_G^{\text{growth}}(t)$ measures the fear a SChMUSR agent has when another group G increases in strength more quickly than its own group does. This is calculated as the relative change in popularity over the last Δt rounds between the two groups. Formally,

$$F_G^{\text{growth}}(t) = \max\left(0, \min[1, \frac{\mathcal{P}_G(t) - \mathcal{P}_G(t - \Delta t)}{\mathcal{P}_{G_{\text{in}}}(t) - \mathcal{P}_{G_{\text{in}}}(t - \Delta t)} - 1]\right). \tag{3}$$

Note that SChMUSR derives no fear from groups that are growing in strength slower than its own, and any growth exceeding twice its own group's growth is treated as if it were twice the growth.

2.2 Chat

SChMUSR sends and processes the five types of chat messages listed in Table **??**. In this subsection, we describe how it uses these messages.

Group Formation. SChMUSR uses chat to help in group formation by proposing and agreeing or disagreeing to changes in group composition. Specifically, SChMUSR alters the group formation mechanisms used by CAB agents to track whether each member wants to be in a group with others. Chat agreement is represented as a matrix C. For C_{ij}, a value of 1 means player i wants player j in their group, a value of -1 means player i does not want player j in their group, and a value of 0 means it is unknown. C is updated anytime another player proposes a group or agrees or disagrees to a proposal. Additionally, C is decayed at a rate of 0.9 each round to favor more recent chat messages. C is used to update the community score for each community G that the agent is considering. Formally,

$$\text{Score}'_{\text{G}} = \left(\frac{\sum_{i,j \in G} C_{ij}}{|C|}\right) \cdot \frac{\theta_{\text{wChatAgreement}}}{100} \cdot \text{Score}_{\text{G}}$$

where G is the set of players the agent is considering for a group, and Score_{G} is the score for a group as computed by CAB agents.

Trust-Building. The trust aspect of SChMUSR bridges the gap between what players say and what they actually do. This form of trust has been shown to be important in decision making and in forming relationships [1, 2, 4, 3]. SChMUSR's trust in another player increases when that player consistently follows through on their stated intentions in chat. This trust value then influences how much SChMUSR relies on that player's future statements.

SChMUSR models the trust it has in all players with the vector $T = (T_0, \cdots, T_{|I|})$, where T_i corresponds to the trust SChMUSR has in player i. The trust value for each player can be anywhere between 0, representing no trust, and 1, representing full trust. Initially, $T_i = \theta_{startingTrust}/100$. As the game progresses, T_i is increased when player i does what they say they will do, and decreased when they fail to do so. For example, when player i states that they will give SChMUSR x tokens and then SChMUSR actually receives at least x tokens from player i, then T_i is increased by $\theta_{trustRate}/100$. However, when player i states that they will give SChMUSR x tokens, but SChMUSR does not receive at least x tokens from player i, then T_i is decreased by $\theta_{trustRate}/100$. T_i is also decreased when another player j states that player i (a) did not give a stated amount or (b) stole from them (i.e., accusations). In these cases, it is updated by subtracting $(\theta_{distrustRate}/100)*(T_j)$ from T_i.

When evaluating how much the agent wants to be in a community G, the community score is updated as follows:

$$\text{Score}''_{\text{G}} = \left(\frac{\sum_{i \in G} T_i}{|C|}\right) \cdot \frac{\theta_{\text{wTrust}}}{100} \cdot \text{Score}'_{\text{G}}.$$

Threat Identification. Chat is also used by SChMUSR to enhance threat identification through fear contagion. This mechanism analyzes messages exchanged throughout the game to determine perceived threats. When SChMUSR's fear level (described in the previous subsection) toward an individual or group exceeds its threshold ($\theta_{fearThreshold}$), it will express it in the chat. SChMUSR keeps track of when others express fear and uses this to influence its own fear. Let $F^{other}_{j,i}(t) = 1$ when player j has stated that they fear player i in round t, and 0 otherwise. Then,

$$F_i^{\text{contagion}}(t) = \sum_j F^{other}_{j,i}(t) \tag{4}$$

Threat Response. Once SChMUSR has decided to attack a threat, it will attempt to coordinate the attack with others. First, SChMUSR calculates the amount of strength needed for the attack to be effective. Then, it recruits others to try to achieve that amount of collective strength by issuing a *token allocation* message (e.g., "I will attack ¡player¿ with ¡x¿ tokens").

3 Code and Additional Data

Additional data, along with all code used to run simulations, define agent algorithms, process and analyze the data is included in the following repository: [REDACTED]

References

[1] A. Baier. Trust and antitrust. *Ethics*, 96(2):231–260, 1986.

[2] Russell Hardin. *Trust and trustworthiness.* Russell Sage Foundation, April 2004.

[3] Thomas M. Jones. Ethical decision making by individuals in organizations: An issue-contingent model. *Academy of management review*, 16(2):366, 1991.

[4] Roger C. Mayer, James H. Davis, and F. David Schoorman. An integrative model of organizational trust. *Academy of management review*, 20(3):709, 1995.

[5] Jonathan Skaggs, Michael Richards, Melissa Morris, Michael A. Goodrich, and Jacob W. Crandall. Fostering collective action in complex societies using community-based agents. *International Joint Conference on Artificial Intelligence*, page 211–219, 2024.