1 MODEL DESCRIPTION

This is a model description of a replication of the model described in Janssen (2008) in Netlogo. The original model was implemented in Java in 2002.

The model description follows the ODD protocol for describing individual- and agent-based models (Grimm et al. 2006) and consists of seven elements. The first three elements provide an overview, the fourth element explains general concepts underlying the model's design, and the remaining three elements provide details. Additionally, details of the software implementation are presented.

Purpose

The purpose of this model is to study the conditions under which agents will cooperate in one-shot two-player Prisoner's Dilemma games if they are able to withdraw from playing the game and can learn to recognize the trustworthiness of their opponents. When the agents display a number of symbols and they learn which symbols are important to estimate the trustworthiness of others, agents will evolve who cooperate in games in line with experimental observations.

State variables and scales

There are n agents in the population. Each generation they play a number of one-shot games against randomly drawn others. Each next generation consists a fraction of new agent whoms parents are selected from a pair wise competition.

Process overview and scheduling

In one generation a certain number (g) of games are played. For each of these, two agents are chosen at random. These agents then decide if they trust each other. If they do trust each other, they play the Prisoner's Dilemma once and the resulting payoffs are added to their respective total scores.

The average material payoff an agent has received from all its interactions (games played or games exited) with other agents is used to determine the offspring in the next generation. Each generation 10% of the population is replaced by new agents who are selected from the population (including those who previously left the system) using a tournament selection algorithm. Two contestants are picked at random from the population, and their average payoffs are compared. The one with the higher average payoff becomes a new agent. If both contestants have identical scores, the winner is picked at random.

The genotype of the new agent (α, β, x_i, w_i) is copied from the parent and then may experience mutation. The mutation rates for strategies, symbols, and weights determine the probability or degree that each individual component will be mutated. In the case of the symbol, a mutation consists of flipping the component to the opposite state (0 (off) to 1 (on) or vice versa). In the case of the motivations and weights, there is always a mutation draw from a Gaussian distribution with means equal to the parameter values after crossover.

Note that the model includes two types of adaptation of the weights. Within a generation agents are able to update the weights in order to learn to recognize the agent

types. Between generations agents derive weights from the previous generation, although those weights are somewhat altered by mutations.

Design concepts

Emergence. Emergence of other-regarding preferences of the agents that lead to cooperation

Adaptation. Agents update the weights of their neural network during a generation by learning and between generation from imperfect inherence from their parents. They also inherent the symbols of their parents, as well as the other-regarding preferences. During a generation the population learn to recognize trustworthy others.

Fitness. Payoff per interaction derived during a generation is used as the fitness indicator.

Sensing. Agents can observe symbols of other agents.

Interaction. Agents interact by proposing to play a game, playing a game by deciding to cooperate or defect.

Stochasticity. Stochasticity exists in the probabilistic decision making, and the noise added to the inherence of the parameters of the next generation.

Initialization

The initial values of α are drawn from a uniform distribution between 0 and 1, and for β between 0 and 1, by which only initial conditions are accepted where $\beta \leq \alpha$. The initial values of the symbol x_i, 0 or 1 for each, are chosen randomly, and all initial weights w_i are set to 0.

Input

Parameter	***Value***
Number of agents (n)	100
Number of symbols (s)	20
Learning rate (λ)	1.0
Steepness (γ)	2
Number of games per generation (g)	500
Mutation rate symbols	0.05
Standard deviation mutation w	0.1
Standard deviation mutation α, β	0.025

Submodels

The Game

Each agent has three possible actions: cooperate (C), defect (D), or withdraw (W). If both players cooperate, they each get a payoff of R (reward for cooperation). If both players defect, they each get a payoff of P (punishment for defecting). If player A defects and B cooperates, A gets a payoff of T (temptation to defect), and B gets S (sucker's payoff). If at least one of the players withdraws from the game, both players get a payoff of E (exit payoff). The resulting payoffs are given in Table 1.

Table 1. Pay-off table of the Prisoner's Dilemma with the option to withdraw from the game.

		Player B		
		Cooperate	Defect	Withdraw
Player A	Cooperate	R,R	S,T	E,E
	Defect	T,S	P,P	E,E
	Withdraw	E,E	E,E	E,E

I assume that the costs and benefits of exit are such that the expected payoffs from choosing not to play are higher than those resulting from mutual defection, but lower than those expected from mutual cooperation. The Prisoner's Dilemma is defined when $T > R > E > P > S$ and $2R > T + S$. In this situation the best option for any one move is to withdraw from the game. If one expects that the other agent will cooperate, the best option is to defect. If one expects that the other agent will defect, the best option is to withdraw. Since the game is symmetrical, each player comes to the same conclusion, so they both withdraw and end up with payoffs that are much lower than if they both trust that the other will cooperate. The pay-off matrix for the game is defined using $T = 2$, $R = 1$, $E = 0$, $P = -1$, and $S = -2$.

Subjects always prefer more for themselves and the other person, but are more in favor of getting payoffs for themselves when they are behind than when they are ahead. The strength of such preferences is increasing in the magnitudes of parameters α and β. The utility can then be formulated as

$$u_i = \pi_i - \alpha_i \max(\pi_i - \pi_j, 0) + \beta_i \max(\pi_j - \pi_i, 0) \quad (1)$$

where u_i is utility of agent i, and π_i is the monetary income of agent i. We define $\beta_i \leq \alpha_i$ and $0 \leq \beta \leq 1$. The α value can be regarded as the strength of an individual's aversion to exploiting others, and β can be regarded as an individual's degree of altruistic tendency. These α and β values determine the strategies of the agents to cooperate or not.

To include the heterogeneity of motivations, I formulate the utility function in Table 2, where the material payoffs can be adjusted by individual motivations.

Table 2. Utility pay-off table of the Prisoner's Dilemma with the option to withdraw from the game.

		Player B		
		Cooperate	Defect	Withdraw
Player A	Cooperate	R,R	$S+\beta_A(T-S)$,$T-\alpha_B(T-S)$	E,E
	Defect	$T-\alpha_A(T-S)$,$S+\beta_B(T-S)$	P,P	E,E
	Withdraw	E,E	E,E	E,E

An agent has three elements: (1) the set of symbols that it displays, (2) the strategy that it uses to decide whether to trust or not another agent, and (3) the strategy it uses in Prisoner's Dilemma games.

The symbols are represented in the following way. Each agent has *s* symbols that can have value 0 or 1, where 0 means no display of the symbol and 1 means display of the symbol. The other two elements require more discussion.

Trust

The rule an agent uses to decide to trust the other agent, and thus be willing to play a Prisoner's Dilemma game, is represented as a single-layer neural network. A neural network has *s* inputs, which are the values 0 and 1 of the other agent's symbols. A weighted sum *M* of these inputs is calculated using the following equation:

$$M = w_0 + \sum_{i=1}^{s} w_i x_i, \tag{2}$$

where w_0 is the bias, w_i is the weight of the *i*th input, and x_i is the *i*th input. Initially, all weights are zero, but during the simulation the network is trained, when new information is derived, by updating the weights as described below in equation (4).

Neural networks use a so-called threshold function to translate the inputs into one output. The standard threshold function used for neural networks is a sigmoid function, and it determines trust defined as the probability *Pr[Tr]* that the agent will cooperate with its prospective partner:

$$\Pr[Tr] = \frac{1}{1+e^{-M}}. \tag{3}$$

The higher the value of *M*, the higher the probability will be. The probability of not trusting the other agent is 1 – *Pr[Tr]*. Since the initial weights are assumed to be zero, the initial value of *Pr[Tr]* is 0.5.

If a game is played, each agent receives feedback, *F*, on the experience. This feedback is simply whether the partner cooperated or not. If the partner cooperated (*F* = 1), the agent adjusts the weights associated with the other agent's symbols upward, so that it will be more likely to trust that agent, and others displaying similar symbols, in the future. On the other hand, if the partner defected (*F* = 0), the agent will adjust the same weights downward, so that it will be less likely to trust that agent and others with similar symbols. The equation to adjust the weights is as follows:

$$\Delta w_i = \lambda \cdot (F - \Pr[Tr]) \cdot x_i, \tag{4}$$

where Δw_i is the adjustment to the ith weight, λ is the learning rate, F is the feedback, $F\text{-}Pr[Tr]$ is the difference between the agent's level of trust in the other agent and the observed trustworthiness of the other agent, and x_i is the other agent's ith symbol. In effect, if the other agent displays the ith symbol, the corresponding weight is updated by an amount proportional to the difference between the observed trustworthiness of an agent and the trust placed in that agent. The weights of symbols associated with positive experiences increase, while the weights of those associated with negative experiences decrease, reducing discrepancies between the amount of trust placed in an agent and that agent's trustworthiness.

Strategies

When neither agent withdraws from playing a game, they have to decide to cooperate or to defect. The agents are assumed conditionally to cooperate. I assume that the players will estimate the expected utility for cooperation, E[U(C)], or defection, E[U(D)]. The expected utility is determined by assuming that the level of expected trust of an agent in its opponent, defined in (3), represents the probability that the opponent will cooperate:

$$E[U(C)] = \Pr[Tr] \cdot R + (1 - \Pr[Tr]) \cdot (S + \beta_i \cdot (T - S)) \quad \text{or} \tag{5}$$

$$E[U(D)] = \Pr[Tr] \cdot (T - \alpha_i \cdot (T - S)) + (1 - \Pr[Tr]) \cdot P\,. \tag{6}$$

Given the two estimates of expected utility, the player is confronted with a discrete choice problem which I address with a logit function. The probability to cooperate, $Pr[C]$, depends on the expected utilities and the parameter γ, which represents how sensitive the player is to differences in the estimates. The higher the value of γ, the more sensitive the probability to cooperate is to differences between the estimated utilities:

$$\Pr[C] = \frac{e^{\gamma \cdot E[U(C)]}}{e^{\gamma \cdot E[U(C)]} + e^{\gamma \cdot E[U(D)]}}\,. \tag{7}$$

Model implementation

Based on the model description in the paper, the model is implemented in NetLogo, using version 4.0. NetLogo is not the most appropriate package to implement this computational intensive model. The replication was mainly performed for pedagogic reasons.

References

Grimm, V., U. Berger, F. Bastiansen, S. Eliassen, V. Ginot, J. Giske, J. Goss-Custard, T. Grand, S. Heinz, G. Huse, A. Huth, J.U. Jepsen, C. Jørgensen, W.M. Mooij, B. Müller, G. Pe'er, C. Piou, S.F. Railsback, A.M. Robbins, M.M. Robbins, E. Rossmanith, N. Rüger, E. Strand, S. Souissi, R.A. Stillman, R. Vabø, U. Visser, D.L. DeAngelis (2006) A standard protocol for describing individual-based and agent-based models. *Ecological Modelling* 198:115-126

Janssen, M.A. (2008), Evolution of cooperation in a one-shot prisoner's dilemma based on recognition of trustworthy and untrustworthy agents, *Journal of Economic Behavior and Organization*, 65: 458-471