final version:
Rino Falcone and Munidar Singh and Yao-Hua Tan (Eds)
Deception, Fraud and Trust in Agent Societies, pp. 133-144,
LNAI 2246, Springer, 2001

# Learning to Trust

Andreas Birk

Vrije Universiteit Brussel, Artificial Intelligence Laboratory,
Pleinlaan 2, 10G725, 1050 Brussels, Belgium
birk@ieee.org

**Abstract.** Evolutionary game-theory is a powerful tool to investigate the development of complex relations between individuals such as the emergence of cooperation and trust. But the propagation of genes is an unrealistic assumption when it comes to model fast-changing social interactions. We show how a transition from evolutionary game-theory to learning can be made. Specifically, we show how cooperation and trust can develop together through social interactions and a suited learning mechanism.

## 1 Introduction

Evolutionary game theory [Axe84,Smi82] is a powerful tool for the investigation of interactions between individuals. It has especially become popular with research on cooperation (see e.g. [AD94] for an overview), but it also has been applied to many other domains.

Evolutionary methods are built upon a transfer of encoded information, i.e., genes, between individuals. This transfer of genes includes two main assumptions. The first one is the "feasibility of breeding" assumption. Evolution includes the generation of new individuals and the deletion or death of others. The second one is the "obey mother nature assumption". When an off-spring is generated, it has no choice whether to incorporate a particular gene or not; the "decision" is made by mother nature or by stochastic operators in simulated evolution.

Let us examine these assumptions from the perspective of trust as a social phenomenon. On the one hand, genetic evolution is very likely to influence animal and especially human behavior also in respect to social interactions. On the other hand, social developments happen on a completely different time-scale than evolution. Therefore, it should be clear that evolution can not serve as the only explanation. The work presented here shows a possible way out of the problems with evolutionary schemes. Instead of evolution, a learning approach somewhat in the spirit of selectionism [Ede87,Ede85] is used here. The main feature of this learning algorithm is that it is based on a pool of potential solutions or so-called hypotheses in each individual. This type of algorithm can also be highly efficient for the learning of individual skills as demonstrated in experiments with learning eye-hand coordination in simulations and real robot systems [BP00,Bir96] and experiments on learning several basic behaviors in a robotic ecosystem [Bir98].

This paper builds on previous results based on evolutionary game theory [Bir00]. It is shown here that cooperation and trust in a continuous-case N-player prisoner's dilemma can not only evolve, but can also be learned. In the general framework of this research, trust is based upon dynamical processes. Other commonly used notions of trust build upon compliance-based approaches, using for example standardized protocols and cryptography. The dynamical notion of trust which is used here guarantess no "absolute" security as trusted systems can cheat. But the process has the important advantage of being completely open and robust. A selection of different approches to trust can be found for example in [CCe00].

In this research framework, the basis for trust is seen as an intrinsic property of each individual in form of the so-called *trustworthiness*. The trustworthiness of an individual $a_A$ is an objective measure for another individual $a_B$ of the desirability of interactions with $a_A$. If the, possibly continuous, trustworthiness of $a_A$ is high, it is highly desirable for $a_B$ to engage in trust-based interactions with $a_A$. The trustworthiness of $a_A$ can be dynamic and it is not directly perceivable by $a_B$. Any process which tries to establish an approximation of the trustworthiness of $a_A$ is denoted as building *trust* in this research framework.

Processes for building trust often include a non-rational component in the sense that decisions on how to deal with another individual are not only based on previous interactions with this individual, but also on other subjective criteria, such as outer appearance, recommendations from others, and so on. Subjective processes for building trust are extremely important as they allow decisions whether to interact or not with unknown individuals, i.e., individuals who have not been encountered in previous interactions. Labels, which do not bear meanings in the beginnings of the experiments, and preferences to interact with individuals carrying certain labels are used here to model the building of trust.

The rest of this paper is structured as follows. In section 2, the basic ideas of the transition from evolutionary game theory to social learning are explained. A continuous case N-players prisoner's dilemma is introduced in section 3 as a basis for the experimental framework. In section 4, the concrete learning algorithms and results are presented. Section 5 concludes the paper.

## 2 Selectionist Learning within Individual Minds

Evolutionary algorithms with their major classes Genetic Algorithms [Gol89,Hol75], Evolutionary Programming [FOW66], Evolutionary Strategies [Sch77,Rec73] and Genetic Programming [Koz94b,Koz92], imitate, or at least are inspired by, the principle of evolution in nature. They use a set of potential solutions (the *population*) to a particular problem or *domain*. Populations are generated in iterations (*generations*) using operations for *selection* and *transformation*. In doing so, the selection and transformation operations focus on good, in respect to a *fitness* function, members of the population. As better members are more likely to be chosen an improvement over time is expected.

For the sake of simplicity, we refer here to any representation of a potential solution as a *gene*, typically a fixed-length binary string or a parse-tree. When using evolutionary algorithms to investigate artificial "living" systems as in the fields of evolutionary game

theory [Axe84,Smi82] or evolutionary robotics [GHF94,WFP99,DBB98,FM94,Koz94a], a single gene determines a crucial aspect of an individual system, such as its morphology [Sim94], its control [FM94], or highlevel behavior like a strategy in social interactions [AH81].

Here we propose a mechanism which is not based on evolution, but which is a learning mechanism inspired by the evolutionary driving forces of selection and the generation of diversity, somewhat in the spirit of selectionism [Ede87,Ede85]. Here a crucial aspect of an individual system is not determined by a fixed gene but by a so-called *hypothesis*. For example, in the domain of robot-control, a certain hypothesis $h$ would for example represent that given a situation $s$ the behavior $b$ would be appropriate. The crucial aspect of a hypothesis is that, unlike in the case of a gene, there is not a single hypothesis for a particular problem. Instead, an individual has *multiple hypotheses* about potential solution for a single instance from the domain. In the case of robot-control, this means that given a situation $s$ there is a set of hypotheses $HS$ linking $s$ to several possible behaviors.

Hypotheses are ranked within the individual by a so-called *preference* function $pref()$. The best hypothesis according to this ranking is most likely to be activated, e.g., to be expressed as an behavior or to serve as a (partial) model of the world. Lower ranking hypotheses also have a chance to become activated. The retrieval of the hypothesis which becomes active can for example be done with the roulette-wheel (RW) principle as follows. Given a hypotheses-set $HS$ and the preferences $pref(h)$ for all $h$ in $HS$, the likelihood $prob$ that a particular hypothesis $h'$ is retrieved from $HS$ for activation is proportional to its preference, i.e.,

$$prob(h' \text{ is activated}) = pref(h') / \sum_{h \in HS} pref(h)$$

Note that it is important not to confuse RW-retrieval with RW-selection from evolutionary algorithms. In the case of evolutionary selection, a gene is transferred into the next generation. If this does not happen, the gene dies out, i.e., it disappears from the population. When a hypothesis $h$ is selected by RW-retrieval, it is applied and tested. This does not necessarily result in a change in the set $HS$ of hypotheses with which $h$ is in concurrency.

Through the retrieval and activation of $h$, information about the usefulness of $h$ is gathered and $pref(h)$ is updated. This in turn can lead to changes of $h$ and even its elimination, but as mentioned above, this is not necessarily the case. With evolution, potential solutions are encoded in genes and transferred among individuals as generations progress. With multiple-hypotheses learning, the potential solutions in form of hypotheses are never transfered between different individuals. Nevertheless, similar hypotheses-sets in different individuals and coordinated usage of hypotheses can emerge through suited social interactions as will be shown later on in experiments.

As already mentioned in the introduction, variations of the learning algorithm presented here have also been applied to learning of individual skills on the level of sensor-motor control and on the level of behaviors for robots in real world environments [BP00,Bir96,Bir98]. In these experiments, it has been shown that a pool of potential solutions in the “mind of a single individual increases the robustness against distortions

from real world noise and it can increase the learning speed through the re-use of partial solutions found in the pool.

## 3 The Experimental Framework

### 3.1 A Continuous-Case N-Player Prisoner's Dilemma

The basis for the experiments described later on is a version of the prisoner's dilemma with $N$ players and continuous cases of investment and payoffs (CN-PD). It is motivated and described in more detail in [Bir00,BW00].

Each agent $a_i$ has a so-called cooperation-level $co_i \in [0.0; 1.0]$. In a game, the cooperation-level determines the agent's investment $I_i$, which serves for the benefit of the group (including $a_i$ itself). Concretely, the investment is determined by:

$$I_i = co_i \cdot 75$$

Let $\bar{co}$ denote the average cooperation-level of the group, i.e.:

$$\bar{co} = \sum_{1 \leq i \leq N} co_i / N$$

The so-called gain $G_i$ for an agent $a_i$ is determined by:

$$G_i = \bar{co} \cdot 100$$

Roughly, all investments are collected, some profit is generated with the investments, and finally investments and profit are distributed among the investors. The dilemma arises as investments and profit are equally shared among all. Thus there is the temptation to invest less than the others and to exploit their contribution to the profit. This becomes even clearer when we look at the netgain or payoff for each agent. This payoff $po_i$ for an agent $a_i$ is the difference between gain and investment, i.e.:

$$po_i = G_i - I_i = \sum_{1 \leq j \leq N} co_j / N \cdot 100 - co_i \cdot 75$$

So, on the one hand, it is in the interest of each agent that there is a high overall investment. On the other hand, there is the temptation to leave the task of investing to others, as the overall gain is distributed among all, independent of the individual investment. Note, that the payoff for an agent depends on its own cooperation level $co_i$ and on the average cooperation level $\bar{co}$. Its profit function $f_p : [0, 1] \times [0, 1] \rightarrow I\!R$ is thus

$$f_p(co_i, \bar{co}) = co_i \cdot -75 + \bar{co} \cdot 100$$

Based on this, we can extend the terminology for payoff values in the standard prisoner's dilemma, with payoff types for cooperation (C), punishment (P), temptation (T), and sucking (S), as follows:

- Full cooperation as all fully invest: $C_{all} = f_p(1.0, 1.0) = 25$

– All punished as nobody invests: $P_{all} = f_p(0.0, 0.0) = 0$
– Maximum temptation: $T_{max} = f_p(0.0, \frac{N-1}{N}) \geq 50$
– Maximum sucking: $S_{max} = f_p(0.0, \frac{1}{N}) \leq -25$

For $co, \bar{co} \neq 0.0, 1.0$, we get the following additional types of payoffs, the so-called partial temptation, the weak cooperation, the single punishment, and the partial sucking. They are not constants (for a fixed $N$) like the previous ones, but actual functions in $(co, \bar{co})$. Concretely, they are sub-functions of $f_p(co, \bar{co})$, operating on sub-spaces defined by relations of $co$ in respect to $\bar{co}$.

### 3.2 Strategies for Iterated Games

When playing iterated games, the concept of a strategy [Axe84] can be used to determine the behavior of an agent. This means, the outcome of previous games is used to compute whether to cooperate or not in the recent game, or to compute the degree of cooperation in the continuous case [RS98].

In [BW00] it is shown that the so-called justified snobism (JS) is a successful strategy for the continuous case N-player prisoner's dilemma. JS cooperates slightly more than the average cooperation level of the group of $N$ players if a non-negative payoff was achieved in the previous iteration, and it cooperates exactly at the previous average cooperation level of the group otherwise.

**Justified-Snobism (JS):**

$$po_i(t-1) \geq 0 : co_i(t) = \bar{co}(t-1) + c_{JS}$$
$$po_i(t-1) < 0 : co_i(t) = \bar{co}(t-1)$$

So, JS tries to be slightly more cooperative than the average. This leads to the name for this strategy as the snobbish belief to be "better" (in terms of altruism) than the average of the group is somehow justified for players which use this strategy.

In addition, following strategies are used in the experiments described lateron to challenge JS:

**Follow-the-masses (FTM)** : match the average cooperation level from the previous iteration, i.e., $co_i[t] = \bar{co}[t-1]$

**Hide-in-the-masses (HIM)** : subtract a small constant $c$ from the average cooperation level, i.e., $co_i[t] = \bar{co}[t-1] - c$

**Occasional-short-changed-JS (OSC-JS)** : a slight variation of JS, where occasionally a small constant $c$ is subtracted from the JS investment

**Occasional-cheating-JS (OC-JS)** : an other slight variation of JS, where occasionally nothing is invested

**Challenge-the-masses (CTM)** : Zero cooperation when the previous average cooperation is below one's cooperation level, a constant cooperation level $c'$ otherwise, i.e.,
  – $co_i[t-1] \geq \bar{co} : co_i[t] = c'$
  – $co_i[t-1] < \bar{co} : co_i[t] = 0$

**Non-altruism (NA)** : always completely defect, i.e., $co_i[t] = 0$
**Anything-will-do (AWD)** : always cooperate at a fixed level, i.e., $co_i[t] = c'$

In evolutionary game theory, each agent has exactly one strategy, which is encoded in a single gene. The survival of the agent and the number of its off springs depend on the performance of this strategy. Here, each agent has a set of strategies from which he can choose. This means strategies are encoded as multiple-hypotheses. The set of strategy-hypotheses for an agent $a_i$ is denoted with $HS_i^s$.

When playing games, an agent must first retrieve a strategy $s$ from $HS_i^s$. This is done using roulette-wheel retrieval as introduced above. The outcome of the game is then used to update the preference $pref(h)$ for the hypothesis that strategy $s$ is useful for getting a high payoff in a game.

### 3.3 The Basis of Trust

Much like in [Bir00], trust here is expressed by preferences of an agent to be grouped together with certain other agents in a game. Again, subjective criteria in the form of labels, as a kind of outer appearance of agents, is used for this. The two major differences with previous work are that here

- the emergence of cooperation and trust is grounded in learning instead of using evolution, and
- agents can change their labels, i.e., there is no a priori assignments of labels to individuals.

The basic principle of the so-called trust-function stays the same as in previous work. So, the function $trust_i : L \rightarrow [0.0, 1.0]$ of an agent $a_i$ maps a weight $w$ to each possible label $l_j$, such that $trust_i(l_j) = w$. The weight $w$ represents $a_i$'s preference to interact with an agent with label $l_j$. If $w$ is high, i.e., close to $1.0$, $a_i$ prefers to interact with agents with label $l_j$, or it simply trusts them. If $w$ is low, i.e., close to $0.0$, $a_i$ prefers not to interact with agents with label $l_j$, or it simply does not trust them.

As mentioned above, individuals do not have a fixed label in the experiments reported here. Instead, the labels are represented within each agent as multiple-hypotheses. The set of label-hypotheses for an agent $a_i$ is denoted with $HS_i^l$. Before each game, an agent must decide which label it signals to the other agents. This will influence the formation of groups and the outcome of the games. So choosing a label is a hypothesis-retrieval and hypothesis-activation for an agent. The hypothesis is that when $a_i$ signals this particular label, the outcome of the next game will be beneficial for $a_i$.

## 4 Learning Strategies, Signals, and Trust

### 4.1 The Structure of the Iterated Games

Before the algorithm running within an agent is presented in more detail, let us first have another look at the overall game. There is a set of agents with a fixed cardinality $n_S$, the so-called *society* $S$. The society plays iterated CN-PDs in time-steps $t$. At the

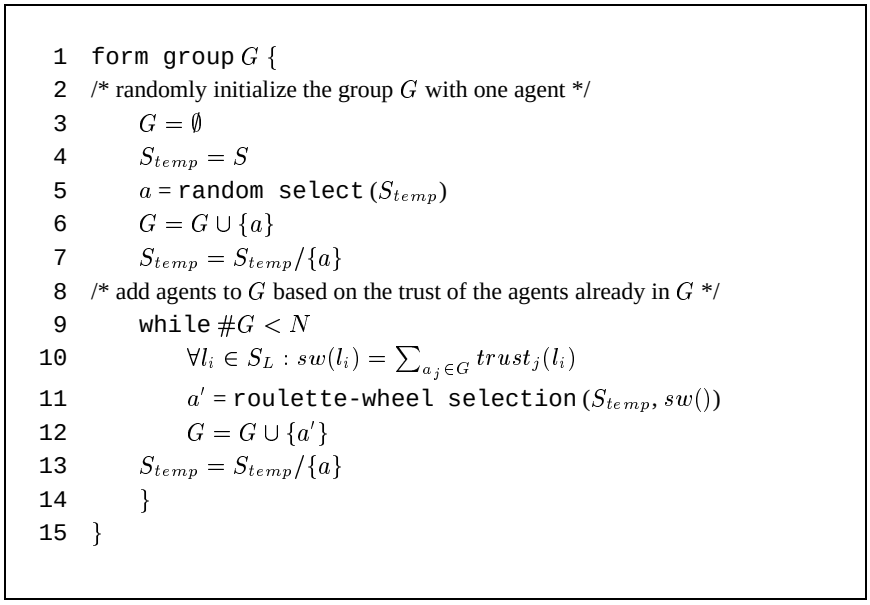


**Fig. 1.** Group formation based on the trust-functions.

beginning of each time-step, the society is split into groups of size $N$ biased by the individual trust-functions.

The concrete algorithm for doing this is shown in Figure 1. First, the group is randomly initialized with one agent. Then, additional agents are put into the group. In doing so, the likelihood of placing an agent $a_i$ who signals label $l$ into the group is proportional to the summed trust in $l$ of all agents which are already in the group. Note that agents are true individuals in the sense that each agent can only be once in one particular group during a game. In evolutionary games in contrast, agents can multiply by generating offspring and thus be represented several times in several groups.

After the groups are formed, several CN-PD games are played and payoffs for each agent are generated. The payoffs are used to update the preferences of the different hypotheses as will be shown in the next section. Afterwards, the groups are mixed together into a uniform society again and the overall process proceeds to the next time-step, $t + 1$.

### 4.2 The Algorithm within an Agent

Figure 2 shows the algorithm running within an individual agent. Most of it has been explained and motivated above. What remains to be defined is the update of the preferences of different hypotheses (lines `7` to `10` for labels and lines `12` to `15` for strategies).

The main idea for the update is simply that the running average of payoffs is used as indication of how preferrable a certain strategy or label is. As a minor problem, negative payoffs have to be taken care of. To ensure that the preferences never become negative

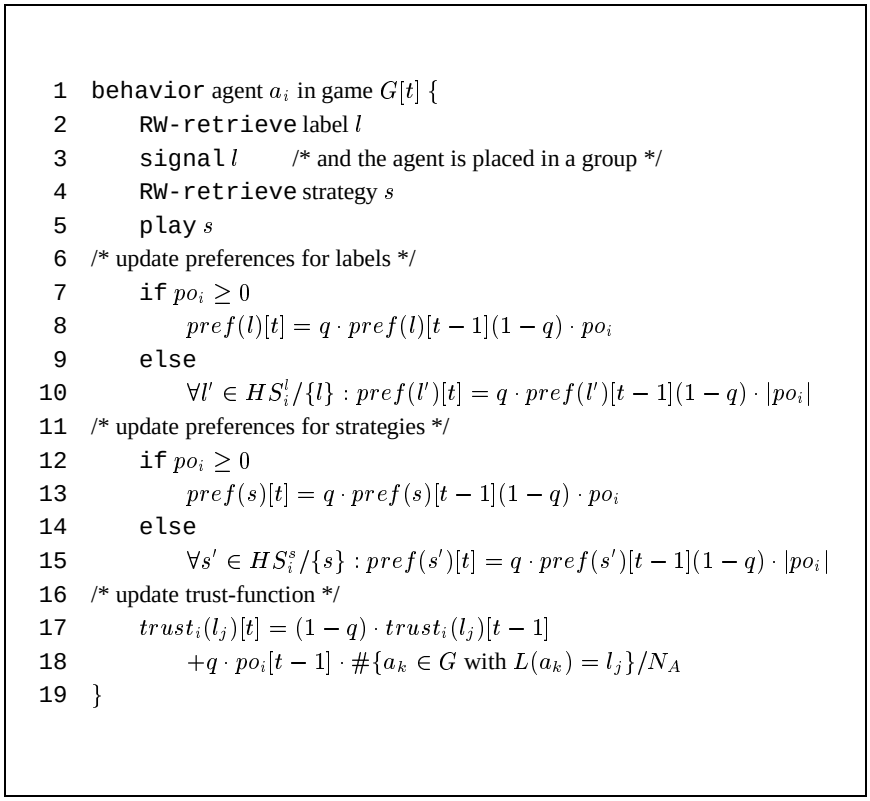


**Fig. 2.** The behavior of an individual agent in a single game in a Pseudo-Code.

as the standard roulette-wheel principle is only applicable with positive weights. Therefore, negative payoffs do not decrease the preference for a strategy $s$, which was active in the last time-step, but they lead to an increase in the preference of all other strategies except $s$ (line `10`). The same holds in respect to labels (line `15`).

### 4.3 Results

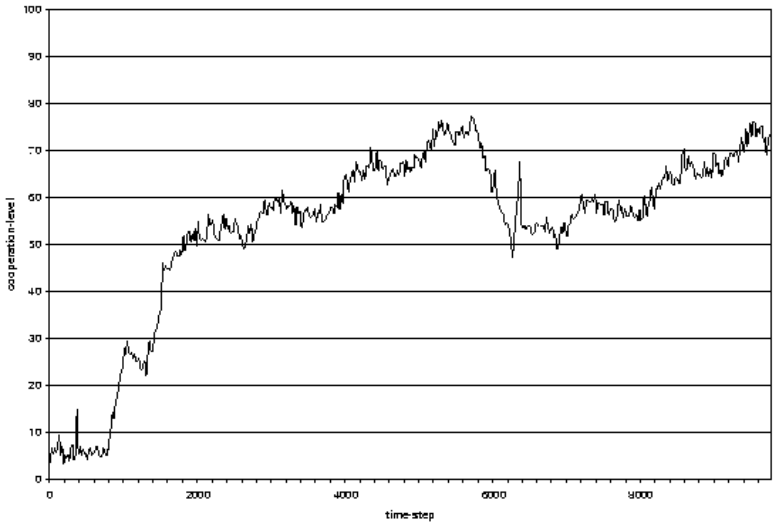


**Fig. 3.** The simultaneous learning of suited strategies, signaled labels, and trust-functions leads to an emergent cooperation.

In the experiments reported here, the size $n_S$ of the society is 100, the group sizes are always 10 agents. After grouping, the agents always play 50 games together to collect payoffs before they proceed to the next time-step and new groups are formed. The preferences for the different hypotheses are randomly initialized for both types, i.e., for strategies as well as for labels.

As shown in figure 3, the simultaneous learning of strategies, signaled labels, and trust-functions leads to an emergent cooperation, i.e., an increasing average cooperation level. This cooperation is nevertheless very vulnerable. Unlike evolutionary experiments where the JS as strategy and the trust into a particular label become dominant and stable, the multiple-hypotheses learning leads to dynamic scenarios.

Especially, when the preferences for JS and certain labels become very high and stable, some agents will (re-)discover in their hypotheses-sets $HS^s$ and $HS^l$ non-altruistic

strategies and how to use deceptive labels. Note, that unlike in the genetic case, these "bad behaviors" can never die out. They are always present, even if they are sometimes rarely observered due to small preferences.

So, when the preferences for JS and certain labels become very high and stable in the majority of the society, activating non-altruistic behaviors and deceptive signals is very profitable as a high amount of general goodwill can be exploited. Agents which retrieve these hypotheses by chance receive high payoffs and therefore increase their preference for them substantially. Therefore, they are likely to retrieve and activate them in the next round again, and so on.

A substantial break in the general cooperativeness and trust can hence occasionally be observed. In a snowball reaction, more and more agents discover "bad behaviors". But the more agents do so, the less profitable this becomes. Therefore, the society is then able to recover again.

## 5 Conclusion

The work presented in this paper is set in a research context where trust is modeled as dynamic preferences of whether to engage in social interactions with others. The basis of this model is an intrinsic property called trustworthiness in every individual $a$. Trustworthiness of $a$ is an objective measure for other individuals regarding whether it is desirable to engage in an interaction with $a$. But trustworthiness cannot directly be perceived. Building trust therefore relates in this model to the estimation of trustworthiness. Subjective criteria like the outer appearance are important for building trust as they allow to handle unknown agents for whom data from previous interactions does not exist. Hence, trust is represented as preference to be grouped together with agents with a certain label to play a game.

In previous work, it was shown that stable relations of trust can evolve and that the co-evolution of trust can boost the evolution of cooperation. In general, evolutionary game-theory is a well known tool for investigating basic properties of interactions between individuals. But the transfer of encoded information, i.e., genes, is unsuited as main basis for models of social interactions. Social interactions do not operate on the time-scale of natural evolution nor do they provide such powerful means of information exchange as the transfer of genes. Here, we show how learning can be used to overcome this severe drawback. The so-called multiple-hypotheses approach is used to successfully develop cooperation and trust simultaneously in scenarios modeled by a continuous-case N-player prisoner's dilemma.

**Acknowledgments**

Andreas Birk is a research fellow (OZM-980252) of the Flemish Institution for Applied Research (IWT).

## References


[AD94] Robert Axelrod and Lisa D'Ambrosio. An annotated bibliography on the evolution of cooperation. *http://www.ipps.lsa.umich.edu/ipps/papers/coop/Evol_of_Coop_Bibliography.txt*, October 1994.

[AH81] R. Axelrod and W. D. Hamilton. The evolution of cooperation. *Science*, 211:1390–1396, 1981.

[Axe84] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.

[Bir96] Andreas Birk. Learning geometric concepts with an evolutionary algorithm. In *Proc. of The Fifth Annual Conference on Evolutionary Programming*. The MIT Press, Cambridge, 1996.

[Bir98] Andreas Birk. Robot learning and self-sufficiency: What the energy-level can tell us about a robot's performance. In *Proceedings of the Sixth European Workshop on Learning Robots*, LNAI 1545. Springer, 1998.

[Bir00] Andreas Birk. Boosting cooperation by evolving trust. *Applied Artificial Intelligence Journal*, 14(8), September 2000.

[BP00] Andreas Birk and Wolfgang J. Paul. Schemas and genetic programming. In Ritter, Cruse, and Dean, editors, *Prerational Intelligence*, volume 2. Kluwer, 2000.

[BW00] Andreas Birk and Julie Wiernik. A successful strategy for a real-world instance of the n-player prisoner's dilemma with continuous degrees of cooperation. Technical report, Vrije Universiteit Brussel, AI-Laboratory, 2000.

[CCe00] Babak Sadighi Firozabadi Cristiano Castelfranchi, Rino Falcone and Yao Hua Tan (eds). Special issue: Trust in agents. *Applied Artificial Intelligence Journal*, 14(8), September 2000.

[DBB98] Peter Dittrich, Andreas Burgel, and Wolfgang Banzhaf. Random morphology robot - A test platform for online evolution. *Robots and Autonomous Systems*, 1998.

[Ede85] Gerald M. Edelman. Neural darwinism: Population thinking and higher brain function. In Michael Shafto, editor, *How We Know*, pages 1–30. Harper and Row, 1985.

[Ede87] Gerald M. Edelman. *Neural Darwinism: The Theory of Neuronal Group Selection*. Basic Books, New York, 1987.

[FM94] D. Floreano and F. Mondada. Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. In *Proceedings of the Conference on Simulation of Adaptive Behavior*, 1994.

[FOW66] L.J. Fogel, A.J. Owens, and M.J. Walsh. *Artificial Intelligence through Simulated Evolution*. Wiley, New York, 1966.

[GHF94] R. Ghanea-Hercock and A. P Fraser. Evolution of autonomous robot control architectures. In T. C. Fogarty, editor, *Evolutionary Computing*, Lecture Notes in Computer Science. Springer-Verlag, 1994.

[Gol89] David Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley, Reading, 1989.

[Hol75] John H. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, 1975.

[Koz92] John R. Koza. *Genetic programming*. The MIT Press, Cambridge, 1992.

[Koz94a] John R. Koza. *Evolution of a subsumption architecture that performs a wall following task for an autonomous mobile robot*, volume II: Intersections Between Theory and Experiment, chapter 19, pages 321–346. MIT Press, 1994.

[Koz94b] John R. Koza. *Genetic programming II*. The MIT Press, Cambridge, 1994.

[Rec73] Ingo Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Fromman-Holzboog, Stuttgart, 1973.

[RS98] Gilbert Roberts and Thomas N. Sherratt. Development of cooperative relationships through increasing investment. *Nature*, 394 (July):175–179, 1998.

[Sch77] Hans Paul Schwefel. *Numerische Optimierung von Computer-Modellen mittels der Evolutions-Strategie*. Birkhäuser, Basel, 1977.

[Sim94] Karl Sims. Evolving 3D morphologgy and behavior by competition. In R. Brooks and P. Maes, editors, *Artificial Life IV*, pages 28–39, Cambridge, MA, 1994. MIT Press.

[Smi82] J. Maynard Smith. *Evolution and the Theory of Games*. Cambridge University Press, Cambridge, 1982.

[WFP99] Richard A. Watson, Sevan G. Ficici, and Jordan B. Pollack. Embodied evolution: Embodying an evolutionary algorithm in a population of robots. In Peter J. Angeline, Zbyszek Michalewicz, Marc Schoenauer, Xin Yao, and Ali Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 1, pages 335–342. IEEE Press, 1999.