Genetic Action Trees

A New Concept for Social and Economic Simulation

Thomas Pitz[a], *Thorsten Chmura*[a]

[a] Laboratory of Experimental Economics, Adenauerallee 24-42, 53113 Bonn, Germany

Abstract
Multi-Agent Based Simulation is a branch of Distributed Artificial Intelligence that builds the base for computer simulations which connect the micro and macro level of social and economic scenarios. This paper presents a new method of modelling the formation and change of patterns of action in social systems with the help of Multi-Agent Simulations. The approach is based on two scientific concepts: Genetic Algorithms [Goldberg 1989, Holland 1975] and the theory of Action Trees [Goldman 1971]. Genetic Algorithms were developed following the biological mechanisms of evolution. Action Trees are used in analytic philosophy for the structural description of actions. The theory of Action Trees makes use of the observation of linguistic analysis that through the preposition *by* a semi-order is induced on a set of actions. Through the application of Genetic Algorithms on the attributes of the actions of an Action Tree an intuitively simple algorithm can be developed with which one can describe the learning behaviour of agents and the changes in action spaces.
Using the extremely simplified economic action space, in this paper called "SMALLWORLD", it is shown with the aid of this method how simulated agents react to the qualities and changes of their environment. Thus, one manages to endogenously evoke intuitively comprehensible changes in the agents' actions. This way, one can observe in these simulations that the agents move from a barter to a monetary economy because of the higher effectiveness or that they change their behaviour towards actions of fraud.

1. Introduction

Computer simulations open up a new way of analysing complex social and economic systems. The simulation stands out against the classical differentiation of deductive and inductive methods in economic theory because in simulations, explicit conditions are formulated in a formal language as it is done in deductive procedures. However, simulation results are not interpreted through proofs but through the inductive evaluation of the data [Axelrod 1971]. Like an experiment, the modularity of a computer programme allows that data are achieved under controlled conditions and precisely presuppositions.

An example of social simulations which has become classical in recent times is the finite cellular automat which was developed by v. Neumann and Ulam [v. Neumann 1966]. At first, it was used for an attempt to model the biological process of self-reproduction (Conway's game of life [Gardner 1970, 1971]). Another field of application of the cellular automat is the analysis of complex technical and physical processes.

In computer models of social sciences, cellular automats are often implemented in order to demonstrate social interdependencies such as segregation effects in populations [Schelling 1969, 1971]. Such emergent phenomena evolve through the fact that through their individual actions, agents generate circumstances which were not intended by them. Other examples for that are the origin of traffic jams [Nagel, Schreckenberg 1992] and the collapse of finance markets. Such paradoxical relationships between the individual actions on the micro level and the collective effects on the macro level describe a fundamental class of problems in social sciences which can very well be modelled through computer simulations.

Applications of computer simulations in economic sciences can often be found in the context of game theory. An important example is the theory of reinforcement learning developed by the biologist Harley, which was intensely analysed by [Roth, Erev 1995; Erev, Roth 1998]. Recent learning models such as Experience Weighted Attraction by Camerer also belong in this field [Camerer 1998]. In simple game situations, experimental data could well be prognosed with the aid of such learning algorithms [Roth, Erev 1995; Selten, Pitz, Chmura 2003].

However, in the approaches of game theoretically motivated learning algorithms, many characteristics of social agents are not taken into account. Thus, many intentional attitudes (beliefs, desires, intentions), the alterations of preferences or the social context of agents is only rudimentarily represented. Communication, adaptive abilities or a representation of the agents' knowledge are nearly completely missing. Apart from that, agents with heterogeneous behaviour patterns depending on the specific situation are important for the adequate depiction of complex social systems. These cannot be recorded by simple learning algorithms.
Due to this criticism, multi-agent systems have developed out of the field of distributed artificial intelligence.

Software entities which differ from the usual learning algorithms in that way they autonomously develop their own strategies for the coping with tasks are called agents. Agents do not own a merely trivial control of their actions. They use and administer knowledge, interact with other agents and possess the ability to cooperate and communicate. Agents perceive their environment and react to changes in their surroundings. They own adaptive abilities and learn from experience and they do not only react to changes in their environment but take the initiative and affect their environment through their actions.

Among the first and probably best analysed architectures of multi-agent systems are the BDI systems (belief-desire-intention) whose development was started by Bratman [Bratman 1980] in the middle of the 1980s. BDI systems are examples of top-down models. This means that the agents' intentional states, beliefs, desires and intentions, are explicitly represented through the BDI structure. Epistemic, modal or temporal logics are used for the manipulation of the BDI structure. Nevertheless, the implementation of these systems is usually very laborious due to the complex logical calculus.

A possible solution is offered by bottom-up modelling on the basis of neural networks or, as in this paper, by Genetic Algorithms [J. H. HOLLAND 1975]. This means that the entities relevant to the model, such as the characteristics of actions or intentional attitudes, are implicitly represented by a codification like binary sequences. On the basis of the codification, fundamental mathematical operations substitute the logical calculi.

Although widely unnoticed by social and economic scientists, multi-agent systems have primarily become established as solutions for problems in engineering and information sciences, for instance in process management, traffic sciences and logistics, as well as on the internet as a virtual service provider and in computer games [Weiss 1999]. Thus, a new approach for the modelling of actions in multi-agent systems shall now be exemplarily analysed in an economic context in this paper. In contrast to this, game

theory normally deals with the ontological structure of actions in a reductionism way. Through strategies, actions are mainly represented as atomic mathematical entities in economic models without any operational semantics. At best, characteristics of actions manifest themselves in their evaluation through the abstract preference relations or utility functions. We take the view that a linguistically motivated analysis of actions offers a deeper insight in the formal structure of actions in order to model those operational and procedural qualities of actions which are important for computer simulations. The terminology of analytical philosophy, especially analytical theories of action [J. L. Austin 1975, J. R. Searle 1971, Chisholm 1964, Davidson 1980], serves as a basis of the concept formation.

Multi-Agent Based Simulation is a branch of Distributed Artificial Intelligence that builds the base for computer simulations which connect the micro and macro level of social and economic phenomena like cooperation, competition, markets and social networks dynamics. The dynamics of social and economic systems manifest themselves in the formation, change and disappearance of actions. For this reason, actions of many diverse forms appear in the application of computer simulations on economic and social processes. That creates the question how actions can be represented in computer simulations. In this paper a general and uniform approach of modelling an as extensive class of actions as possible is analysed. On the one hand, this concept is supposed to take into consideration the diversity of actions, on the other to be operationally manageable nevertheless for the modelling of simulation models such the multi agents system.

Searching for an integral model of actions, one inevitably encounters the question which qualities of actions separate these from different entities. This can be expressed the most forcefully with the ontological question “What are actions?”. Due to the heterogeneity and fundamental fuzziness of actions, one can give only incomplete answers to this. Therefore, we will attempt to approximate the ontological question through an epistemic and an evolutionary translation:

1. How can actions be described through a general model that is suitable for Multi-Agent simulations?
2. How can the formation and disappearance of actions be described in this model algorithmically?

We will now develop a suitable general data format and an algorithm for the generation of modifications of actions and thus give a possible answer to the epistemic and evolutionary question. Afterwards, we will explain the method with the help of an economically motivated example. For this economic example a Multi-Agent Simulation SMALLWORLD was implemented. In these situations one can observe how a set of agents develops from a barter to a monetary economy or how they change their behaviour towards actions of fraud. SMALLWORLD is, as the name suggests, a strongly simplified model of an economic action space. Nevertheless, this simplification is not fundamental in nature and can easily be rectified in a more complex but methodically analogous model.

It was not the aim of this paper to compare quantitative results with empirical data. Rather, the statistical evaluation of the simulations serves to prove endogenous qualitative changes in the behaviour of the agents in the simulations. The main aim was

to exemplify the method of Genetic Action Trees with intuitively self-evident and deliberaty almost trivial examples.

Of course, we do not claim that there is a bijection of our model to the ontological structure of social processes. But as physical elementary particles do not solve differential equations; their behaviour could be described by differential equations. In this sense, the concept of Genetic Action Trees is understood as a pragmatic vehicle for modelling the dynamic of social and economic systems.

2. The epistemic question

2.1 Action types

In order to find answers to the epistemic question concerning a uniform system of describing actions, it is helpful to make use of some of the concepts used in analytical theories of action. For a definition of these fundamental notions in theories of action, it is worthwhile to examine the way in which actions are represented in common speech. Descriptions of actions such as "Last week Peter drank 2 bottles of wine with his two colleagues in his favourite pub." first of all suggest understanding actions as concrete singular objects to which a context of limited expansion in space and time can be attached. But if we consider actions to be singular entities in this way, we are confronted with the problem of their repeatability and countability. It is therefore advisable to take a further step of abstraction and to subsume similar concrete actions as equivalence classes, so-called action types. Hence, the repetition of an action should be understood as the carrying out of an equivalent but indeed numerically different action. It is exactly the analysis of iterations of similar actions which plays an essential role in computer simulations. A certain quantity of action types is called action space.

2.2 Action attributes

Just as the ontological question of the nature of actions cannot be solved generally, likewise the question of the characterizability of the equivalence relation cannot finally be answered but only in regard to the respective model. Instead of a formal definition of the equivalence relation for action types, we use quasi-deictic denotations of action types taken from common speech: action types can be denoted by using the infinitive form of verbs “to walk, to buy, to produce, to kill” etc. and a set of free variables. The free variables serve as a parameter for references of space and time as well as for the actors and objects affected by the processes when the action is performed. Likewise, we will treat quantitative and qualitative action attributes, such as “5 m/s” for “to walk” and “maliciously” for “to kill” etc. as free variables. In the following, we will call these free variables action attributes. Finally, we will understand certain preferences agents might have with regard to actions, such as “with pleasure” for “to travel” and “reluctantly” for “to drive a car”, as action attributes in our model.

Talking about actions, it is naturally impossible to specify all free variables through an explicit allocation. Therefore, we will restrict ourselves in the notation to those variables which are essential, i.e. relevant for the model. The number and sort of free variables of an action type thus depends on the model. The more detailed the model, the higher the number of parameters and the more differentiated the range of the variables. The execution of an action belonging to an action type formally means the instantiation of all free variables.

Notation

- If $H([i_1,...,i_n])$ is an action type with a list of free variables, let $H([i_1,...,i_n]:[k_1,...,k_n])$ or in short $H(k_1,...,k_n)$ be the execution of an action of type H with the allocation $[k_1,...,k_n]$ of the free variables $[i_1,...,i_n]$. Due to better legibility, the square brackets [ ] are sometimes left out.
- We use the meta-variable "_" for a set of variables which is not completely specified.

Example 1 (action types)

- *to eat*$([i]:[Peter])$; Possible semantics: Peter is eating.
- *to drink*$([i,w]:[Peter,wine])$; Possible semantics: Peter is drinking some wine.
- *to buy from*$([i,j,b,g]:[Peter,Paula,10€,book])$; Possible semantics: Peter buys a book from Berta for 10 €.
- *to offer*$([i,k,J]:[Paula,wine,consumer])$; Possible semantics: Paula offers the product wine to a set *J* of consumers.

2.3 Action Trees

On certain action spaces the preposition *by* induces a semi-order ($<_{by}$) [Goldman 1971].

Example 2 (Unbranched Action Tree)

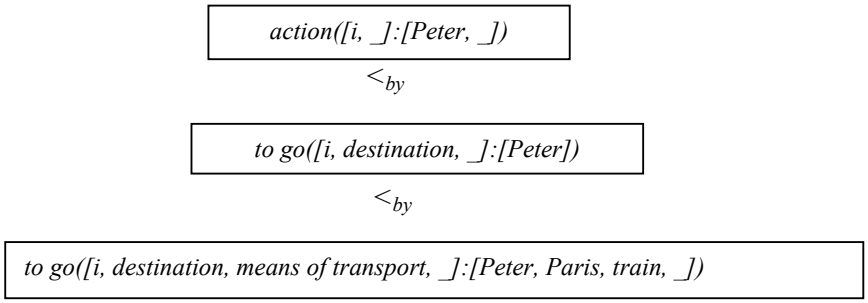


Figure 1: Unbranched Action Tree

Such action spaces can be arranged as a tree diagram using the most general action type *action(i,_)* and a set of free variables "_" (see Figure 1). Here the semantics of *action(i,_)* be that "an agent *i* carries out any kind of action". This action is specified from the root to a leaf by the allocation of more and more of the free variables.

2.4. Binary decision-making actions – degree of intention

The degree of intention in SMALLWORLD

Decisions can be understood as special forms of action. We thus suggestively call them decision-making actions. In an Action Tree, decision-making actions possess at least two followers with regard to the $<_{by}$-relation.

In the simulations of the model SMALLWORLD described in this paper, we restrict ourselves to binary decisions, i.e. decisions with two alternatives (see Figure 2). *H* is a binary decision-making action with the nodes H_1 and H_2 which can be carried out by

the agent in principle. In order to depict the agent's preference regarding the two alternatives, we use a degree of intention of H_1. The degree of intention of H_1 is represented by a real number $d_{H_1} \in [0,1] \cup \{2\}$. The selection algorithm at the node H can be described as follows: One chooses a random number c from a finite subset $C \subseteq [0,1]$. Then $H1$ is chosen exactly then if $c < d_{H_1}$ is valid. Thus, changes in behaviour result from a modification of the set C. With $d_{H_1} = 0$ ($d_{H_1} = 2$), the node H_1 (H_2) can be eliminated in the model. The degree of intention d_{H_1} implicitly also determines the intensity of the wish to carry out H_2. The degree of intention describes what the agent *wants* to do. We understand d_{H_1} as the action attribute belonging to the decision-making action H.

Example 3 (branched Action Tree)

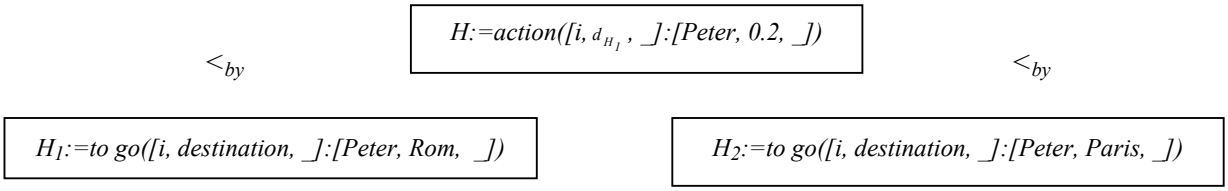


Figure 2: Action Tree with binary decision-making action

In Figure 2, let d_{H_1} denote the degree of intention of the left node, i.e. the intensity of the wish to carry out the left node of the tree.

2.5 The selection of an action type

The selection of an action type by an agent can be seen as the walk through the Action Tree from the root to a leaf. In doing so, the agent chooses one of the two subsequent nodes depending on the degree of intention, as described under 2.4. Occasionally, the agents get instructions from other agents which they are forced to follow. For instance, Peter can have order to go to Paris. The free variable "destination" is then already allocated with "Paris". In this case, Peter's selection process starts at the node where free variables appear for the first time. In the example of Figure 2, no decisions remain to be taken by the agent in this case. In the following, we will restrict ourselves to Action Trees with the following qualities:

1. The Action Tree let be completely defined with regard to the actual model, i.e. each node exhaustively describes one action regarding to the given model. If a leaf of an Action Tree is reached, all variables which are necessary for the performance of the action are allocated.
2. Due to better legibility, we will only analyse Action Trees with two nodes at maximum in this paper.
3. If an agent is unable to carry out any node, no action is conducted. The process of selecting an action will be restarted later on. In the example of SMALLWORLD, the Action Tree was determined in a way which always allows the agent to choose an action.

3. The evolutionary question – an algorithm for the modification of action attributes

3.1. Genetic Action Trees

The dynamics of Action Trees can be described through the alteration of the allocation of action attributes. In order to conduct these alterations as methodically coherently as possible, one can represent allocations of action attributes as sets of binary series. In the following, the allocation of an action attribute coded through the binary series will also be called the gene of an action attribute. A set of genes is called a gene pool. The alteration of the allocations results from the genetic algorithms operating on the gene pool of the action attribute (see 3.5).

Definition 1 (Genetic Action Trees**)**
A Genetic Action Tree *G(T)* for a set of agents *I* has the following structure:

- *T* is an Action Tree.
- $\forall i \in I \,.\, \forall H[\mathrm{A}] \in T \,.\, \forall\ a \in A$ let $C(i,a) \subseteq \{0,1\}^n$ be the gene pool of an action attribute *a* of the action type *H* and of the agent *i*.
- $\forall H \in T$ of a decision-making action and the subsequent nodes H_1 and H_2, there is a degree of intention $d \in [0,1] \cup \{2\}$ and a potentially empty set of exogenous conditions Δ_{H_1} and Δ_{H_2}, which have to be fulfilled for the specific agent i so that H_1 or H_2 can be chosen by *i*.
- $\forall\ c(i,a) \in C(i,a)$ let $\varphi(c(i,a)) \in \mathrm{IR}$ be the fitness of $c(i,a)$.
  After the execution of an action, the fitness of $c(i,a)$ is modified with regard to the results of the action.
- $\forall\ c(i,a) \in C(i,a)$ let $\delta(c(i,a))$ be the semantics of $c(i,a)$.
  The semantics describes how the action has to be carried out in the computer simulation depending on the gene.

In the next section, the meaning of the terms introduced just now shall be exemplified on the basis of how they are used. First, we will describe the choice algorithm of the actions through the agents.

3.2 The choice of the actions through the agents

1. After having been activated, each agent $i \in I$ checks if the list of instructions for actions the agent might have received contains any elements. In SMALLWORLD, all instructions for actions are carried out successively first in first out. If no instructions for actions have been received by the agent *i, i* determines which action has to be carried out by following a path $[H_1,...,H_n]$ of an Action Tree from the root H_1 to a leaf H_n. While doing so, all attributes of the action types $[H_1,...,H_n]$ are successively randomly allocated with coded values $c(i,a) \in C(i,a)$ on the basis of an equal distribution. Let $H_{\otimes}(i,d_{H_1})$ be a decision-making action and $d_{H_1} \in [0,1] \cup \{2\}$ the degree of intention for the left node. If the agent is confronted with a decision-making action $H_{\otimes}(i,d_{H_1})$, the subsequent nodes H_1

and H_2, for which Δ_{H_1} or rather Δ_{H_2} have been violated, are eliminated. Δ_{H_1} and Δ_{H_2} are exogenous conditions specific to the model which guarantee the consistency and coherence regarding the agent i's abilities in the agent's environment. One example of a condition in SMALLWORLD is for instance that *i* is only able to buy a commodity if *i* owns enough money. Thus, the exogenous conditions define which actions *can* be carried out by *i*.

2. If it is the case that two nodes remain for *i* at the node $H_{\otimes}(i,d_{H_1})$, the agent is confronted with tow alternative actions. We describe the semantics of the decision-making algorithm:

   Let $\beta : \{0,...,2^n-1\} \to \{0,1\}^n$, $n \in IN$, be the natural bijection which assigns each binary series $[x_1,...,x_n] \in \{0,1\}^n$ with $\beta(k) = [x_1,...,x_n] \leftrightarrow k = \sum_{i=1}^{n} x_i 2^{i-1}$.

   Let $\delta : \{0,...,2^n-1\} \to \{0,1\}^n$ be defined by $\delta(k) = \frac{k}{2^n-1}$, for each $k \in \{0,...,2^n-1\}$.

   Let $C(i,d_{H1}) \subseteq \{0,1\}^n$ be the gene pool of the agent *i* with regard to the degree of intention d_{H_1}. The left node is chosen exactly then if $\delta(c(i,d_{H_1})) < d_{H_1}$ holds, while $c(i,d_{H_1})$ is chosen randomly from the set $C(i,d_{H_1})$.

3. It is conceivable that an agent receives instructions for actions from another agent. That means precisely that action attributes are determined through another agent's instruction. The decision-making process then starts at a node H_t, $1 < t \le n$ at which free variables appear for the first time, if all the conditions Δ_H of all preceding nodes *H* have been fulfilled. If a condition is violated, the instruction will not be carried out.

4. When the agent *i* has reached the root H_n of the Action Tree, all attributes are allocated with coded values. The action type is carried out in accordance with theses allocations and each allocation $c(i,a) \in C(i,a)$ with which the action type was carried out is judged through the change in its fitness value $\varphi(c(i,a))$ due to the results of the action.

The simulations in SMALLWORLD were conducted with 100 agents and 1000 periods. In each period all agents are activated in a random sequence. In an active phase an agent chooses exactly one action, as described above, and carries it out.

3.3 The evaluation of an agent's action

After the agent's run through an Action Tree, all attributes which are necessary for the specification of an action within the context of the model are clearly defined. Each gene has been assigned a concrete specification through the semantics δ which belongs to the actual model. After the determination of the necessary parameters, the action can now be carried out in the model according to the semantics δ. Actions change the agent's environment. Those changes are called the results of an action. The results of actions form the basis of the evaluation of actions or more precisely the evaluation of

the genes of action attributes through the alteration of the fitness value φ. We will describe this process in the case of SMALLWORLD in detail in chapter 4.

3.4 Changes in the characteristic of action attributes

Genetic algorithms are used for the simulation of changes in action spaces. In the following, specific genetic algorithms will be discussed which have proved useful in the case of the action spaces.

If a gene pool $C(i,a)$ of an agent *i* and an action attribute *a* is changed, this can evoke changes in the behaviour of agent *i*.

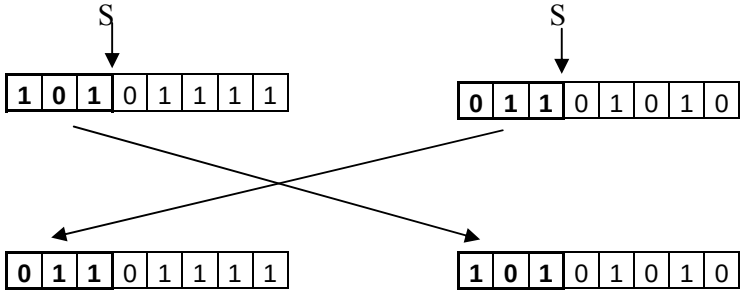
Let $J \subset I$ be a subset of agents and *a* an action attribute. Let $C_J(a) := \bigcup_{i \in J} C(i,a)$ be the common gene pool of the agents $i \in J$ of a. If *J* contains more than one agent, the intersubjective changes in behaviour can thus be simulated. For those application examples in SMALLWORLD analysed in this paper, the gene pool $C_I(a)$ was used. Here was valid: For all agents *i* and action attributes $C(i,a)$ contained exactly one element.

3.5 The Genetic Algorithms for attributes of the action types

The Genetic Algorithms consist of mutation, selection and cross over. In SMALLWORLD, the Genetic Algorithms were used on the sets $C_J(a)$ when the actions belonging to the attribute *a* had been carried out 10 times.

Selection and mutation
Each gene $c(i,a) \in C_J(a)$ is changed with the probability $p_{mut}(c(i,a))$ at n_{mut} pairwise different places which have been chosen randomly. If the gene $c(i,a)$ of the agent *i* is changed through mutation, the oringinal gene is substituted with its mutant. $p_{mut}(c(i,a))$ is here proportional to the fitness $\varphi(c(i,a))$. Thus, the substitution of a gene with its mutant is the more likely the smaller the fitness of the gene is.

Cross Over:
On the sets $C_J(a)$ a cross over was carried out on 5% of the elements of $C_J(a)$. A cross over is presented as an example in the following figure. The cut *S* is determined randomly on the basis of uniform distribution.



4. SMALLWORLD

We will now describe the application of this technique on a simplified economic Action Tree called SMALLWORLD in an exemplary manner. In the model, we will restrict ourselves to the analysis of a subset of the action attributes: to the degree of intention of decision-making actions. While doing so, we will compare simulations having constant fitness functions with simulations having non-constant fitness functions.

First, the course of the agent's selection of an action will be described in detail. All occurring action types and all action attributes relevant to the simulating model will be listed. Furthermore, exogenous conditions for the execution of an action type and for the evaluation of the results of an action will be presented.

An outline of the Action Tree can be found in Appendix 1, a detailed description of the semantics of the nodes in Appendix 2.

4.1 The agents' environment

4.1.1 Communication in SMALLWORLD

The agents in SMALLWORLD have the possibility to write, retrieve and delete messages to other agents in a communication database. There are mainly two different types of messages:

1. An agent can direct his message at the whole population. For instance, an agent *i* in SMALLWORLD can offer a good *k* to the other agents. The notation for that is *mail(offer(i,k))*.

2. It should be mentioned that in SMALLWORLD an agent *i* can give an instruction for performing an action *act* an agent *j*. We use *do(i, act(j))* as a notation for that. The agent *j* adds this instruction to a list of instructions which are worked through first in first out in SMALLWORLD.

4.1.2. Attributes of the agents and the environment in SMALLWORLD

There is a list of attributes for each agent. In SMALLWORLD, these are the cash resources and the number of goods which are assigned to each agent. Apart form that, certain values of attributes in the agents' environment are used. In the examples dealt with in this paper, the unit price of a good in SMALLWORLD will be given exogenously, for instance.
I denotes the set of agents who act in an economic context in SMALLWORLD. A set of goods $G = \{1,...,m\}$ exists in SMALLWORLD. At the beginning of the simulation, each agent $i \in I$ owns a non-negative amount of money $S_i \in \mathrm{IR}_+$ and a vector of goods $V_i := (V_{i1},...,V_{im}) \in \mathrm{IR}_+^m$. Here, V_{ik} denotes the quantity of a good *k* which the agent *i* has at it's disposal. To simplify the depiction of the action types, we will use in this paper goods which are different in form but equally priced, i.e. for all $k \in G$ a uniform price $P_k := 1$ will be given exogenously for the duration of the simulation. If goods are exchanged, it will be done one by one.

4.2. Selection of an action in SMALLWORLD

When an agent $i \in I$ is activated in the simulation model, the process of choosing an action starts at the root [*action(i,*$d_{consume}$*,_)*] of the Action Tree in the simulations described in this paper (see Appendix 1). The agent *i* can decide at this node weather to carry out an action by consuming the goods the agent owns or by trying to get hold of new goods. As described in section 3.1, he chooses one of the two subsequent nodes *consume(i,k)* or *get(i,k)* in consideration of the degree of intention $d_{consume}=\frac{1}{8}$. Beforehand, the consistency is checked for both of the subsequent nodes to find out if they can be conducted (see Appendix 2). The action type *consume(i,k)* cannot be carried out if $V_{ik} = 0$ applies for all goods. In this case, *i* does not own any good and has to choose *get(i,_)*. If $V_{ik} > 0$ is valid for at least one good and *i* chooses the action type *consume(i,_)*, then this means in SMALLWORLD that *i* chooses randomly form the set of goods *i* owns one good *k* on the basis of uniform distribution and that *i*'s number of pieces of the chosen good is reduced by one unit. If the agent *i* reaches the inner node *get(i,*d_{cheat}*,_)*, *i* has to decide weather to take a good by cheating [*get_cheat(i,_)*] with regard to the degree of intention d_{cheat} or by not cheating [*get_not_cheat(i,_)*]. Two different types of cases haven been analysed:

CASE I: Without actions of fraud, [with barter (s.b.)],
$d_{cheat} = 0$ is valid for the degree of intention, i.e. action of fraud are not possible.

CASE II: With actions of fraud, [without barter (s.b.)]
$d_{cheat} = \frac{1}{2}$ is valid for the degree of intention, i.e. actions of fraud are possible.

We will first exemplify CASE I:
The agent has to choose *get_not_cheat(i,_)*. In this case, an agent *i* can get hold of a good either by producing it [*produce(i,k)*] or by trading [*trade(i,_)*]. With the degree of intention $d_{produce}=\frac{1}{4}$, *i* chooses *produce(i,k)*. This means in SMALLWORLD that *i* randomly chooses a good *k* on the basis of uniform distribution and that this possession of this good *k* increases by one unit. The agent can only choose *trade(i,_)* if at least one of the following two conditions is fulfilled: 1. The agent owns a sufficient amount of money to buy a good (here $S_i > 0$). 2. The agent owns a sufficient number of pieces of at least one good to practise barter (here: $\{k \in G.V_{ik} > 0\} \neq \varnothing$). If none of these conditions is fulfilled, the agent has to choose *produce(i,k)*. When an agent *i* reaches the node *trade(i,*d_{aquire}*,_)*, *i* chooses one of the two nodes *aquire(i,_)* or *dispose(i,_)* with regard to the degree of intention $d_{aquire} = \frac{1}{2}$. In order to acquire a good [*aquire(i,_)*], the agent has to possess an amount of money sufficient for the purchase or a quantity of at least one good which is sufficient for an exchange. In order to dispose of a good [*dispose(i,_)*], *i* has to own at least one good.

If an agent *i* at the node *trade(i,*d_{aquire}*,_)* chooses the subsequent node *aquire(i,*$d_{buy,}$*_)*, the agent has again two alternatives: either to buy a good [*buy(i,_)*] or to acquire it via an exchange [*a_exchange(i,_)*]. If the agent decides in favour of [*buy(i,_)*] at the node *aquire(i,*$d_{buy,}$*_)* with regard to the necessary conditions and the degree of intention $d_{buy}=\frac{1}{2}$, then the agent randomly chooses a good *k* form a set of goods he can buy on the bases of uniform distribution. If an agent $j \neq i$ already offers the good *k* for sale, the transaction is completed: *i* receives one unit of the good *k* and *j* one unit of money. If all

exogenous conditions of the subsequent nodes of *aquire(i,$d_{buy,}$_)* are fulfilled, then the node *buy(i,_)* is chosen if $\delta(c(i,d_{buy}))< d_{buy}$ is valid. If the successful purchase shall be rewarded, this can be done by increasing $\varphi(c(i,d_{buy}))$ by 1.

If no agent offers the good *k* for sale, *i* writes a request $mail(b\_search([i,k]))$ into the communication database provided that that has not already been done.

If the agent decides in favour of *exchange(i,_)* at the node *aquire(i,$d_{buy,}$_)* with regard to the necessary exogenous conditions, the agent randomly chooses from the set of goods a good *k(i,in)* which the agent wants to acquire by exchanging it for a randomly chosen good *k(i,out)* which has to be in the agent's possession. If an agent $j \neq i$ already offers to exchange the good *k(i,out)* for *k(i,in)*, the transaction is completed: *i* receives one unit of the good *k(i,in)* and *j* one unit of the good *k(i,out)*. If all exogenous conditions of the subsequent nodes of *acquire(i,$d_{buy,}$_)* are fulfilled, then the node *a_exchange(i,_)* is chosen if $\delta(c(i,d_{buy})) \geq d_{buy}$ applies. If the successful exchange shall be rewarded, this can be done by increasing $\varphi(c(i,d_{buy}))$ by 1.

If no agent $j \neq i$ demands this good, then *i* writes the message $mail(a\_search(i,k(i,in),k(i,out)))$ into the communication database.

If an agent *i* at the node *trade(i,d,_)* chooses the subsequent node *dispose(i,$d_{sell,}$_)*, the agent has again two alternatives: either to sell a good [*sell(i,_)*] or to get hold of it via an exchange [*d_exchange(i,_)*]. If the agent at the node *dispose(i,c(i,d_{sell}),_)* decides in favour of *sell(i,_)* with regard to the necessary conditions and the degree of intention $d_{sell}=\frac{1}{2}$, then the agent randomly chooses a good *k* from the set of goods he can sell on the basis of uniform distribution. If an agent *j* already demands the good *k*, the transaction is completed: *i* gives one unit of the good to *j* and *j* pays one unit of money for it.

If no agent wants to buy the good *k*, *i* writes an offer into the communication database provided that that has not already been done.

If the agent at the node *dispose(i,$d_{sell,}$_)* decides in favour of *d_exchange(i,_)* with regard to the necessary conditions, the agent randomly chooses a good *k(i,out)* form the set of all goods which he wants to dispose of by exchanging it for a randomly chosen good *k(i,in)*. If an agent already wants to exchange the good *k(i,out)* for the good *k(i,in)*, the transaction is completed: *i* gives one unit of the good *k(i,out)*, *j* one unit of the good *k(i,in)*.

If this is not the case, the agent writes the message $mail(d\_search(i,k(i,out),k(i,in)))$ into the communication database.

The evaluation of the gene *c(i,d_{sell})* after the execution of the action type *sell(i,_)* or *d_exchange(i,_)* can take place in accordance to the evaluation of the gene *c(i,d_{buy})* after the execution of the action type *buy(i,_)* or *a_exchange(i,_)*.

We will now discuss CASE II:

If the agent at the node *get(i,_)* chooses the subsequent node *get_not_cheat(i,_)*, the selective process is continued as in CASE I. But in the simulations of CASE II the possibility of barter was excluded, i.e. it applied $d_{buy} = d_{sell} = 2$. If the agent chooses at

the node *get(i,_)* the subsequent node *cheat(i,_)*, he has two alternatives in SMALLWORLD: either to pretend to buy something [*feign_buy(i,_)*] or to pretend to sell something [*feign_sell(i,_)*]. If the agent chooses feign_*buy(i,k)* with regard to the degree of intention $d_{feign\_buy} = \frac{1}{2}$, the agent randomly determines a good *k*. If an agent *j* offers a good *k* for sale, *i* takes one unit of the good and *j* does not receive anything from *i*. A case of fraud has taken place. If no agent offers *k*, *i* starts a search. The relevant message is inserted into the communication database.
If *i* chooses the node *feign_sell(i,k)*, *i* randomly chooses a good *k*. If an agent *j* wants to buy the good *k, i* takes one unit of money and does not receive anything of *k*. A case of fraud has taken place. If no agent wants to buy *k, i* starts to offer *k*. The relevant message is inserted into the communication database. The "guilty conscience" of a swindler can be represented by reducing the fitness of $\varphi(c(i,d_{cheat}))$ by 1.

5. The results of the simulations

In the following, the results of the simulations will be described. On the basis of the model described under 4, SMALLWORLD CASE I and CASE II were carried out with constant as well as non-constant evaluation functions. For each case, *N*=1000 simulations with 100 agents and 1000 periods were conducted. For a definition of the value *N* see Appendix 3.

CASE I: Barter – no actions of fraud

Subcase A

Constant evaluation functions:
The evaluation function was constant for the duration of the simulations, i.e. successful trade was not rewarded. At the beginning of the simulations, sufficient resources (money, goods) were made available to the agents so that the exogenous conditions of the action types were always fulfilled. The action types *buy(i,_), a_exchange(i,_), sell(i,_), d_exchange(i,_)* were chosen with almost the same frequency.

Subcase B

The effectiveness of monetary acting – non-constant evaluation functions:
The evaluation function was changed as described in section 4, i.e. successful trade was rewarded. In comparison with Subcase A one can observe that the action types *buy(i,_)* and *sell(i,_)* were chosen more often than *a_exchange(i,_)* and *d_exchange(i,_)*.
Figure 3 shows the simulation results of Subcase A and B.

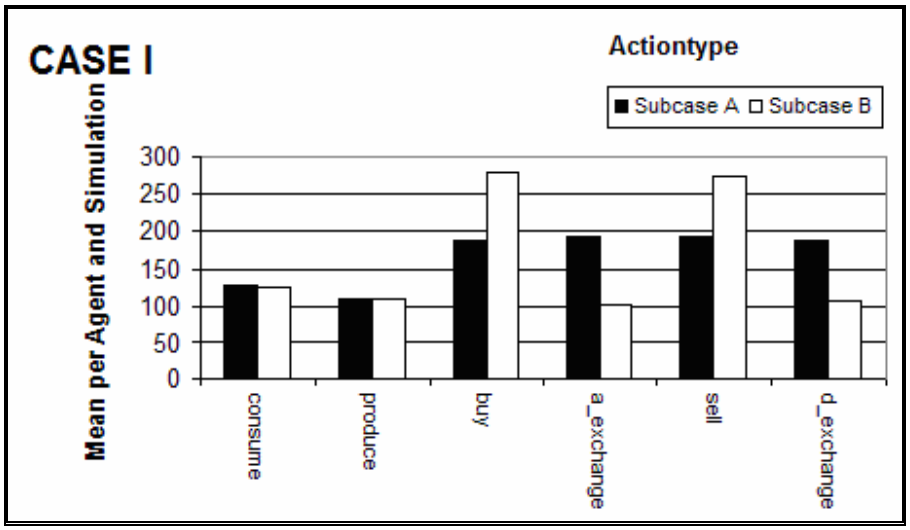


Figure 3: CASE I

The main reason for those results is that agents are more often able to complete a transaction when they pay with money or receive money as means of payment. That means that genes with $\delta(c(i,d_{buy}))< d_{buy}$ or rather $\delta(c(i,d_{sell}))< d_{sell}$ are rewarded more often and thus able to assert themselves in the simulations. In Figure 4 the black lines represent possible trading actions the dotted lines represent impossible trading actions. This shows that the change from barter to a monetary economy is more successful for the agents.

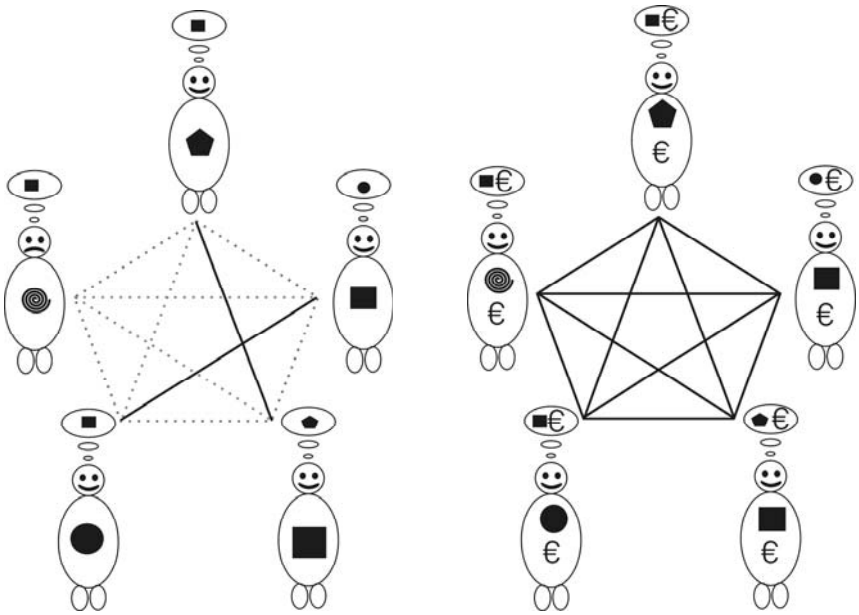


Figure 4: Barter and monetary economy.

CASE II: Actions of fraud

Subcase A

Constant evaluation functions:
The evaluation function was constant for the duration of the simulations, i.e. actions of fraud were not punished. The agents chooses the type of action *get_cheat(i,_)* with the same frequency as *get_not_cheat(i,_)*.

Subcase B

The swindlers' "Guilty conscience" – non-constant evaluation functions:
The evaluation function was changed as described in section 4, i.e. actions of fraud are punished. Since the gene $\delta(c(i,d_{cheat}))$ is punished in the case of an action of fraud, the agents now choose the action type *get_not_cheat(i,_)* considerably more often than the action type *get_cheat(i,_)*. Figure 4 shows the Simulation results of Subcase A and B.

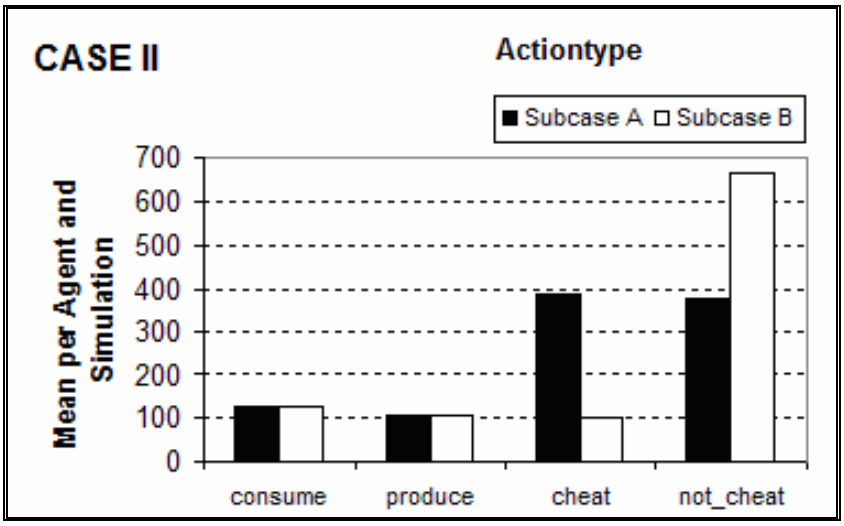


Figure 5: CASE II

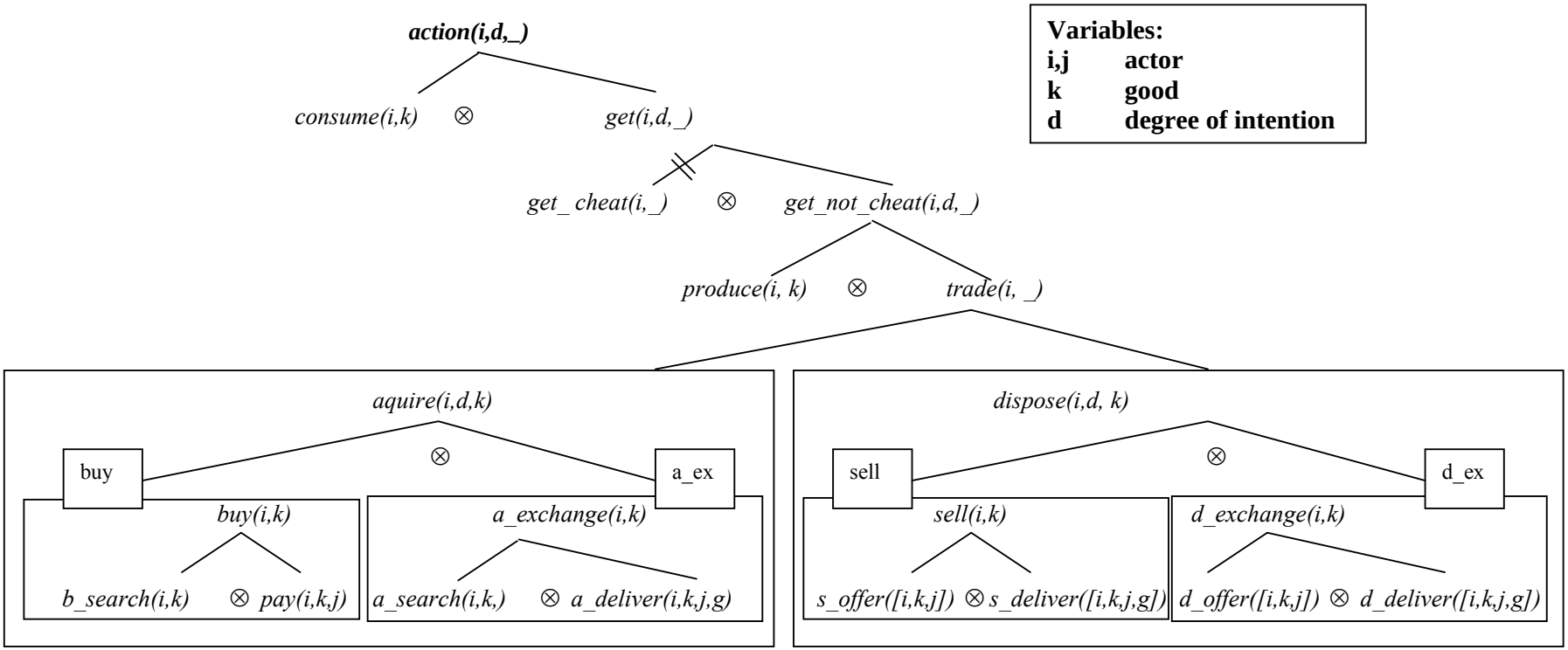
6. Outlook

In this paper, a general concept of the modelling of action spaces has been presented. The epistemic question "How can actions be described through a general model suitable for computer simulations?" was answered by a concept used in analytic theories of action, the Action Trees. For the evolutionary question "How can the emergence and disappearance of actions be described through a uniform algorithm within this model?", it was made use of Genetic Algorithms. The synthesis of these two methods, the Genetic Action Trees, was tested on the simple application SMALLWORLD. As a result, we succeeded in endogenously evoking changes in the agents' behaviour which are intuitively comprehensible. For instance, one could observe in the simulations that the agents changed from a barter to a monetary economy or that they changed their behaviour towards actions of fraud. The method is very promising and can be expanded and applied in many different ways.

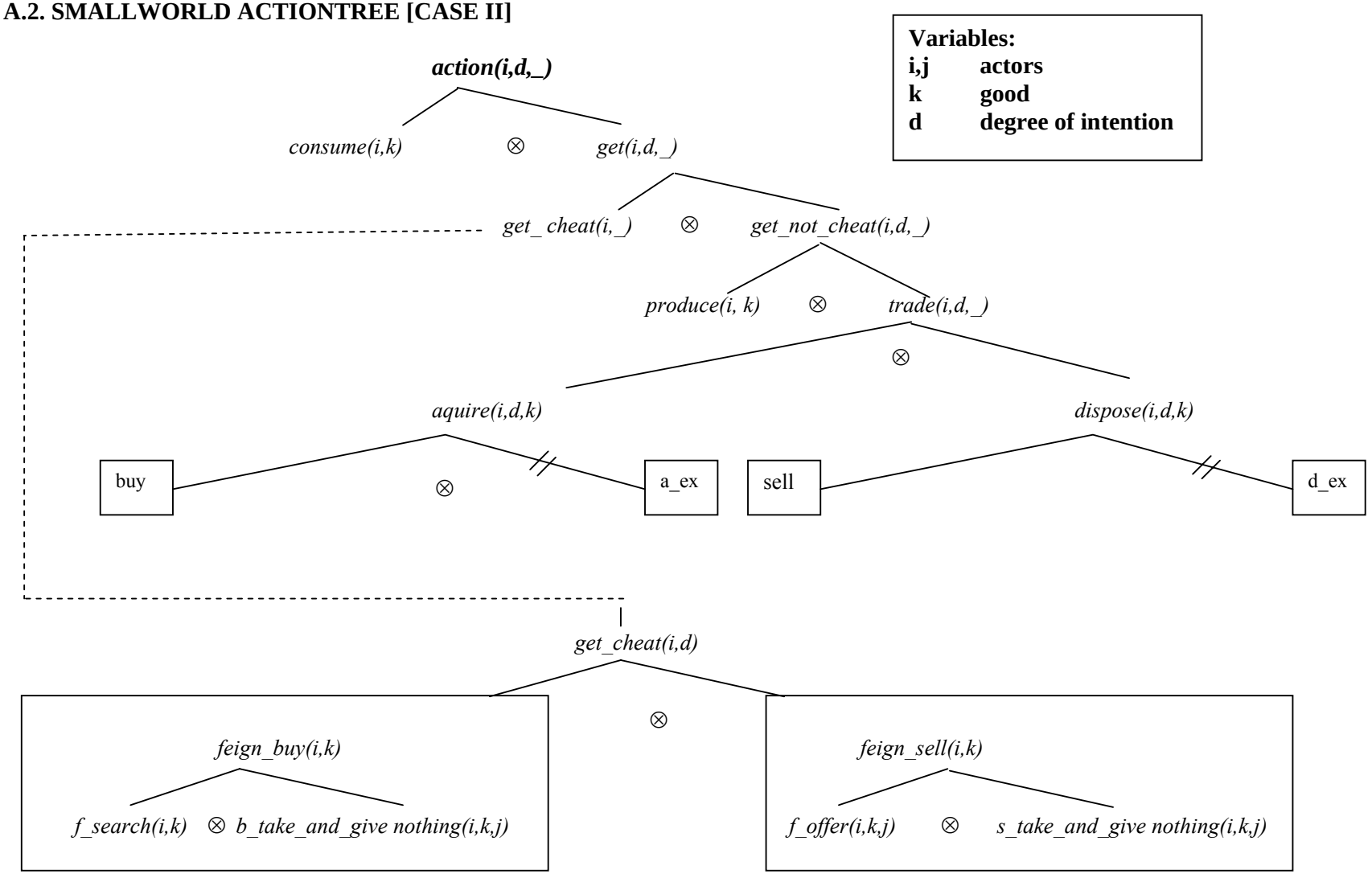
In this work, the turning on of the evaluation functions is done manually in the end through the manipulation of the source codes. One should mention in this context that the evaluations of actions can as well be understood as actions. Thus, an ordering of evaluative actions into an Action Tree is conceivable.

A. Appendix 1

A.1. SMALLWORLD ACTIONTREE [CASE I]



A.2. SMALLWORLD ACTIONTREE [CASE II]



Appendix 2 - Semantics of SMALLWORLD Action Types

We will first introduce some helpful technical terms.

- Let $j \in_r M$ mean that j is chosen randomly form a set M on the basis of uniform distribution.
- If x is the current value of a variable before the execution of any kind of action, $\circ x$ denotes the value of this variable after the execution of the action type.

Let m be a message as described under 4.1.1. We use the following abbreviations: $m \in CB$, ($m \notin CB$) is (not) an element in of the communication database CB. $m \rightarrow_w CB$ ($m \rightarrow_d CB$) m is inserted (deleted) in (from) CB.

The scheme for the specification of an action type H consists of the following aspects:

T: name of the action type, of the subsequent node and of all the free variables that have been allocated up to this node
C: necessary exogenous conditions of the subsequent nodes H_1 and H_2
R: the results of actions: changes in the attributes of the agents and their environment
A: modification of the fitness of the genes

action(i, $d_{consume}$,_)
T: $action([i, d_{consume}, \_]:[i, \frac{1}{8}, \_]) <_{by} consume(i,k) \otimes get(i,\_)$

consume(i,k)
T: $consume(i,k)$
C: $C = \{k \in G : V_{ik} > 0\} \neq \emptyset$
R: $k \in_r C, \circ V_{ik} = V_{ik} - 1$

get(i,p,_)
T: $get([i,d_{cheat}, \_]:[i,\_]) <_{by} get\_cheat(i,k) \otimes get\_not\_cheat(i, k)$

CASE I: $d_{cheat} = 0$

get_not_cheat(i, $d_{produce}$,_)
T: $get\_not\_cheat([i, d_{produce}, \_]:[i, \frac{1}{4}, \_]) <_{by} produce(i, k) \otimes trade(i,\_)$

produce(i,k)
T: $produce(i,k)$
R: $k \in_r G, \circ V_{ik} = V_{ik} + 1,$

trade(i, d_{aquire})
T: $trade([i, d_{aquire}, \_]:[i, \frac{1}{2}, \_]) <_{by} aquire(i,\_) \otimes dispose(i, \_)$
C: $S_i > 0 \ \vee \ \{k \in G.V_{ik} > 0\} \neq \emptyset$

aquire(i,d_{buy},_)

T: $aquire([i, d_{buy}, \_]) <_{by} buy(i,\_) \otimes aquire\_exchange(i, \_)$

CASE I: $d_{buy} = \frac{1}{2}$

CASE II: $d_{buy} = 2$

dispose(i, d_{sell},_)

T: $dispose([i, d_{sell}, \_]) <_{by} sell(i,\_) \otimes dispose\_exchange(i, \_)$

CASE I:: $d_{sell} = \frac{1}{2}$

CASE II: $d_{sell} = 2$

buy(i,_)

T: $buy([i,k]) <_{by} search(i,\_) \otimes pay(i, \_)$

C: $S_i > 0$

R: $k \in_r G$

a_exchange(i,_)

T: $a\_excange([i,k(i,in),k(i,out)]) <_{by} search(i,\_) \otimes deliver(i,\_)$

C: $E = \{k \in G : V_{ik} > 0\} \neq \varnothing$

R: $k(i,in) \in_r G,\ k(i,out) \in_r E$

sell(i,_)

T: $sell([i,k]) <_{by} offer(i,\_) \otimes deliver(i,\_)$

C: $S = \{k \in G : V_{ik} > 0\} \neq \varnothing$

R: $k \in_r S$

d_excange(i, k(i,out),k(i,in),_)

T: $d\_excange([i,k(i,out),k(i,in)]) <_{by} offer(i,\_) \otimes deliver(i,\_)$

C: $E = \{k \in G : V_{ik} > 0\} \neq \varnothing$

R: $k(i,out) \in_r E,\ k(i,in) \in_r G$

b_search(i,k)

T: $b\_search(i,k)$

C: $\neg\exists j \in I.mail(s\_offer(j,k)) \in CB$

R: $IF\ mail(b\_search(i,k)) \notin CB\ THEN\ mail(b\_search(i,k)) \rightarrow_w CB$

pay(i,k)

T: $pay(i,k)$

C: $\exists j \in I.mail(s\_offer(j,k))$

R: $\circ V_{ik} = V_{ik} + 1,\ \circ V_{jk} = V_{jk} - 1, \circ S_i = S_i - 1, \circ S_j = S_j + 1$

$mail(s\_offer([j,k])) \rightarrow_d CB$

A: $\circ \varphi(c(i,d_{buy})) = \varphi(c(i,d_{buy})) + 1$

a_search(i, k(i,out),k(i,in))

T: $a\_search(i,\ k(i,out),k(i,in),\_)$

C: $\neg\exists j \in I.mail(d\_offer(j,k(j,in),k(j,out))) \wedge$
$k(i,in) = k(j,out) \wedge k(j,in) = k(i,out)$

R: $IF\ mail(a\_search(i,k(i,out),k(i,in))) \notin CB$
$THEN\ mail(a\_search(i,k(i,out),k(i,in))) \rightarrow_w CB$

a_deliver(i, k(i,out),k(i,in))

T: $a\_deliver(i,\ k(i,out),k(i,in))$

C: $\exists j \in I.mail(d\_offer(j,k(j,in),k(j,out)))$
$\wedge\ k(i,in) = k(j,out) \wedge k(j,in) = k(i,out)$

R: $\circ V_{ik(i,out)} = V_{ik(i,out)} - 1, \circ V_{jk(j,out)} = V_{jk(j,out)} - 1$
$\circ V_{ik(i,in)} = V_{ik(i,in)} + 1, \circ V_{jk(i,in)} = V_{jk(i,in)} + 1$
$mail(d\_offer(j,k(j,in),k(j,out))) \rightarrow_d CB$

A: $\circ \varphi(c(i,d_{buy})) = \varphi(c(i,d_{buy})) + 1$

s_offer(i,k)

T: $s\_offer(i,k)$

C: $\neg\exists j \in I.mail(b\_search(j,k))$

R: $IF\ mail(s\_offer(i,k)) \notin CB\ THEN\ mail(s\_offer(i,k)) \rightarrow_w CB$

s_deliver(i,k)

T: $s\_deliver(i,k)$

C: $\exists j \in I.mail(b\_search(j,k))$

R: $\circ V_i = V_{ik} - 1, \circ V_j = V_{jk} + 1, \circ S_i = S_i + 1, \circ S_j = S_j - 1$
$mail(b\_search(j,k)) \rightarrow_d CB$

A: $\circ \varphi(c(i,d_{sell})) = \varphi(c(i,d_{sell})) + 1$

d_offer(i, k(i,out),k(i,in))

T: $d\_offer(i,k(out),k(in))$

C: $\neg\exists j \in I.mail(a\_offer(j,k(j,in),k(j,out))) \wedge$
$k(i,in) = k(j,out) \wedge k(j,in) = k(i,out)$

R: $IF\ mail(d\_offer(i,k(i,out),k(i,in))) \notin CB$
$THEN\ mail(d\_offer(i,k(i,out),k(i,in))) \rightarrow_w CB$

d_deliver(i, k(i,out),k(i,in))

T: $d\_deliver(i,\ k(i,out),k(i,in))$

C: $\exists j \in I.mail(a\_offer(j,k(j,in),k(j,out))) \wedge$
$k(i,in) = k(j,out) \wedge k(j,in) = k(i,out)$

R: $\circ V_{ik(i,out)} = V_{ik(i,out)} - 1, \circ V_{jk(j,out)} = V_{jk(j,out)} - 1$
$\circ V_{ik(i,in)} = V_{ik(i,in)} + 1, \circ V_{jk(i,in)} = V_{jk(i,in)} + 1$
$mail(a\_offer([j,k(j,in),k(j,out)])) \rightarrow_d CB$

A: $\circ \varphi(c(i,p_{sell})) = \varphi(c(i,p_{sell})) + 1.$

CASE II: $d_{cheat} = \frac{1}{2}$

get_cheat(i,p,_)
T: $get\_cheat([i,d_{feign\_buy}),\_]:[i,\frac{1}{2}]) <_{by} feign\_buy(i,k) \otimes feign\_sell(i,k)$

feign_buy(i,_)
T: $buy(i,\_) <_{by} f\_search(i,\_) \otimes b\_take\_and\_give\_nothing(i,\_)$

feign_sell(i,_)
T: $sell([i,c(i,d),k]) <_{by} f\_offer(i,\_) \otimes s\_take\_and\_give\_nothing(i,k)$

b_take_and_give_nothing(i,k)
T: $b\_take\_and\_give\_nothing(i,k)$
C: $\exists j \in I.mail(s\_offer([j,k]))$
R: $\circ V_{ik} = V_{ik} + 1, \circ V_{jk} = V_{jk} - 1, \circ S_{ik} = S_{ik}, \circ S_{jk} = S_{jk}$
A: $\circ \varphi(c(i,d_{cheat})) = \varphi(c(i,d_{cheat})) - 1$

s_take_and_give_nothing(i,k)
T: ***s_take_and_give_nothing(i,k)***
C: $\exists j \in I.mail(b\_search([j,k]))$
R: $\circ V_i = V_{ik}, \circ V_j = V_{jk}, \circ S_i = S_i + 1, \circ S_j = S_j$
A: $\circ \varphi(c(i,d_{cheat})) = \varphi(c(i,d_{cheat})) - 1$

Appendix 3 – Number of simulations

For the description of the development of the agents' action spaces the relative frequencies of actions executed per agent are determined from a set of simulations. We use the following considerations for the determination of the number N of simulations necessary for the assessment of the expectation value of relevant random variables: We assume a Gaussian distribution with expectation value μ and the standard deviation σ. In this case, we get the $1-\alpha$ confidence interval for μ with unknown σ:

$$\left[\bar{x}_n - t_{n-1;1-\frac{\alpha}{2}} \frac{s}{\sqrt{n}}; \bar{x}_n + t_{n-1;1-\frac{\alpha}{2}} \frac{s}{\sqrt{n}}\right].$$

Let $\bar{x}_n$ be the mean over n samples, s an unbiased estimator of σ and $t_{n-1;1-\frac{\alpha}{2}}$ the $1-\frac{\alpha}{2}$ quartile of the t-distribution with $n-1$ degrees of freedom.

Following the two-step method by [Stein 1945], one can determine the number N of the necessary simulations (samples) with the required exactness. Let d be the lengths of the confidence interval. First, N_0 simulations are conducted. The total number N can then be determined as follows:

$$N = min\left\{ k \in IN : k \geq \left(\frac{2st_{N_{N_0-1;1-\frac{\alpha}{2}}}}{d} \right)^2 \right\} .$$

Here,

$$s^2 = \frac{1}{N_0 - 1} \sum_{i=1}^{N_0} (x_i - \bar{x}_{N_0})^2$$

is the estimation of the variance. The following values were used in this paper: $N_0 = 50$, $\alpha = 0.1$, $d = 5$.

References

J. L. AUSTIN. H*ow to Do Things with Words, Harvard,* (1975)

R. AXELROD. *The Complexity of Cooperation*, Princeton University Press, Chichester, (1997)

M. E. BRATMAN. *Intentions, Plans and Reason*, Harward University Press, (1987)

R. CHISHOLM. *The Descriptive Element in the Concept of Action*, Journal of Philosophy, p. 613-624, (1964)

C. CAMERER, H. TECK-HUA. *Experience Weighted Attraction Learning in Normal-Form Games,* Econometrica, (1998).

D. DAVIDSON. *Essays on Action and Events*, Oxford University Press, (1980)

I. EREV, A. E. ROTH. *Predicting How People Play Games: Reinforcement Learning in Experimental Games with Unique*, Mixed Strategy Equilibria, American Economic Review, 88(4), p. 848-81, (1998).

M. GARDNER. *The fantastic combinations of John Conway's new solitaire game life*, Scientific American, 120-123, October, (1970)

M. GARDNER. *On cellular automata, self-reproduction, the Garden of Eden and the game life*, Scientific American, 224(2), 112-117, February, (1971)

D. E. GOLDBERG. Genetic *algorithms in search, optimization, and machine leaning*, Addison-Wesley, Reading, (1989).

A. GOLDMAN. *The individuation of action,* The journal of Philosophy, 68:761 – 774, (1971).

C. B. HARLEY. *Learning in Evolutionary Stable Strategy*, J. Theoret. Biol. 89, 611-633, (1981).

J. H. HOLLAND. *Adaptation in natural and artificial systems,* University of Michigan Press, Ann. Arbor, (1975), reprinted by Cambridge: MIT Press. (1992)

J. F. LASLIER, R. TOPOL, B. WALLISER. *A behavioral learning process in games*, Games and Economic Behavior 37, 340-366, (2001)

K. NAGEL, M. SCHRECKENBERG. *A cellular automaton model for freeway traffic*. J. Physique I 2, 2221–2229, (1992).

J. V. NEUMANN. *The Theory of self-reproducing automata*, in Burks, A. W., (Ed), University of Illinois, Urbana, (1966)

A. ROTH, I. EREV. *Learning in Extensive Games: Experimental Data and Simple Dynamic Models in the Intermediate Term*, Games and Economic Behavior, 8, 164-212, (1995).

J. R. SEARLE. *The philosophy of language*, London: Oxford University, (1971)

T. C. SCHELLING. *Models of segregation*, American Economic Review 59, 488 – 493, (1969)

T. C. SCHELLING. *Dynamic models of segregation*, Journal of Mathematical Sociology 1, 143-186, (1971)

R. SELTEN, T. PITZ, T. CHMURA, M. SCHRECKENBERG AND J. WAHLE. *Experiments on Route Choice Behaviour*, H. Emmerich, B. Nestler, (Ed.), Lecture Notes in Computers Science 32, Schreckenberg, Interface and Transport Dynamics, (Springer, Heidelberg), (2003)

C. STEIN. *A two sample test for a linear hypothesis whose power is independent of the variance,* Annals of Mathematical Statistics 43, 243-258, (1945)

G. WEISS. *Multi-Agent Systems*, MIT-Press, (1999)