Reinforcement learning model for the emergence of common property and transhumance in Sub-Saharan Africa

Balázs Pintér
Eötvös Loránd University
Pázmány Péter s. 1/C
Budapest, Hungary
bli@elte.hu

Ákos Bontovics
Eötvös Loránd University
Pázmány Péter s. 1/C
Budapest, Hungary
bontovic@elte.hu

András Lőrincz
Eötvös Loránd University
Pázmány Péter s. 1/C
Budapest, Hungary
andras.lorincz@elte.hu

ABSTRACT

We consider social phenomena as challenges and measures for learning in multi-agent scenarios for the following reasons: (i) social phenomena emerge through complex learning processes of groups of people, (ii) a model of a phenomenon sheds light onto the strengths and weaknesses of the learning algorithm *in the context* of the model environment. In this paper we use tabular reinforcement learning to model the emergence of common property and transhumance in Sub-Saharan Africa. We find that the Markovian assumption is *sufficient* for the emergence of *property sharing*, when (a) the availability of resources fluctuates (b) the agents try to maximize their resource intake independently and (c) all agents learn simultaneously.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents, multi-agent systems*

Keywords

reinforcement learning, adaptive agents, NewTies platform

1. INTRODUCTION

The NewTies EU FP6 project[1][2] started from two constraints: NewTies defined a series of challenges from social phenomena and wanted to model those phenomena through emergences, that is, collective behavior that can not be trivially explained on the level of individual agents. NewTies ended with the following conclusion: modeling through emergences provides information about the efficiency of individual and social learning *together* with the constraints about the environment. The reason is that in every model one tries to satisfy Occam's razor principle: "entities should not be multiplied unnecessarily" and tries to build a minimal model, but simplicity of the model may constrain potential emergences. One particular aspect of NewTies was that individual learning was taken seriously: for each agent, sequential decision making was treated within the framework of reinforcement learning (RL) and the Markov decision process (MDP) model of RL [11] motivated by psychology and neuroscience [8]. Special constraints were (re-)discovered during this endeavor, e.g., (i) agents should build a model about the mind of the other agent [6] and (ii) factored reinforcement learning is necessary to counteract combinatorial explosion in complex scenarios [12, 3]. The so called 'herders challenge' is relevant, because it shows an example where neither of the above conditions is necessary to emerge joint social learning. The relevant aspect of this learning scenario is that the fluctuation of rainfall, and, because of that, the spatial and temporal fluctuation of resources is large.

We begin our paper with a brief introduction to the herders challenge, multi-agent systems, reinforcement learning and the modeling framework we used, NewTies (Section 2). Then, in Section 3 we describe our agent architecture: how our agents store information in their memory (in maps), how they perceive the world (through features), and how they act in this world (with macros). We provide the details about the model of the environment, the agents, and the interactions between the agents in Section 4. We continue with our results and their discussion (Section 5). Conclusions are drawn in Section 6.

2. PRELIMINARIES

2.1 The herders challenge

Hardin, in his noted paper about the tragedy of the commons [4] described how, if left unchecked, herders would keep increasing the size of their stocks grazing on a common pasture until the pasture is overgrazed to the point that it can no longer sustain them. If a herder decides to add one more animal to his herd, he gains all the benefit, but the community as a whole bears the cost. So the positive utility of adding the animal is nearly +1, but the negative utility is only a fraction of -1. A rational herdsman will add more and more animals to his stock, because that way he gets all the benefits and bears only a fraction of the cost.

In the last century the preeminent problem concerning African pastoralists was thought to be the degradation of rangelands because of excessive livestock numbers, based on the same argument [1]. The scientific basis for this view has been the concept of rangeland carrying capacity. This

[1]New and Emergent World models Through Individual, Evolutionary, and Social Learning, http://www.new-ties.eu

Cite as: Multi-agent model of the emergence of common property and transhumance in semi-arid areas like Sub-Saharan Africa, B. Pintér, Á. Bontovics and A. Lőrincz, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. XXX-XXX.

notion is also the basis of Hardin's paper: the increase in animal numbers decreases the availability of forage, a finite resource. In the end, there will be too many animals for the land to carry, and the land will no longer be able to sustain them. Hardin concludes that freedom in commons brings ruin to all.

One of the argument's premises is that the herders operate in a closed system. The scenario does not take into account any outside influences such as weather. But weather is the most powerful force in Africa, for example 83 percent of variation in the areal extent of the Sahara between 1980 and 1989 was explained by variations in annual rainfall [5]. And it changes everything: the equilibrium of animals and forage on which Hardin's argument rests ceases to exist.

Fluctuation is a significant risk that pastoralists have to cope with. According to [7], the most prominent livelihood strategy of pastoralists is the movement of their herds in reaction to anticipated seasonal and annual changes in pasture availability. Transhumance and common property are their means of averaging out the fluctuations. They can not change the environment, so they always move to the territories with more favorable weather.

Based on the arguments above, we decided on the following model. We start with a population of ranchers. Each of them owns a territory independently of the others. Rainfall is distinct and fluctuates independently in each territory. Ranchers can initiate the combining of territories; to do so, they only have to tell another to 'share': to give one another usufruct (i.e. the right to use it) to each other's territory. If many of these reciprocal agreements are established, groups of solitary ranchers will evolve into communities of herders, where everyone is free to graze on another's land.

In the next sections we review the two pillars our paper is based on: multi-agent systems (MAS) and reinforcement learning (RL). We also introduce NewTies, the framework our multi-agent model is realized in.

2.2 Multi-agent systems

The basic unit in multi-agent modeling is the agent. The agents are situated in an environment, and can interact with it and with each other. They are autonomous, have a local view of the system and operate in a decentralized way.

Multi-agent systems have several advantages compared to other, more traditional modeling methods such as dynamical systems. For one, the agent population can be heterogenous. We can create any number of different agents, and put them into the same model. Another huge advantage is that time is present and flowing in the simulation. The environment can change as the simulation advances, and so can the agents, they can adapt to the environment and to each other with the help of learning algorithms.

Another advantage of multi-agent systems is that they can connect the micro and the macro level. Other methods typically only model either the macro level with aggregate data, or the micro level. In multi-agent models, the agents act on their individual, micro level, but their behavior can produce phenomena on the macro level. Collective behavior can emerge that can not be trivially explained on the level of individual agents.

2.3 Reinforcement learning

Reinforcement learning (RL) [11] is a framework for training an agent for a given task based on positive or negative feedback called *immediate rewards* that the agent receives in response to its actions. Mathematically, the behavior of the agent is characterized by a *Markov decision process* (MDP), which involves the *states* the agent can be in, *actions* the agent can execute depending on the state, a *state transition model*, and the *rewards* the agent receives.

One popular method for solving MDPs is based on *value functions* that represent long term utilities that can be collected starting from a given state. The agent's behavior, also called *policy*, is then defined by acting greedily according to this value function, i.e. selecting actions that result in next states with highest possible value.

2.3.1 The Markov Decision Process and the Bellman equations

Consider an MDP characterized by the tuple (X, A, P, R, γ), where X is the (finite) set of states the agent can be in, A is the (finite) set of actions the agent can execute, $P : X \times A \times X \to [0, 1]$ is the transition probability model, i.e. $P(x, a, x')$ is the probability that the agent arrives at state x' when executing action a in state x, $R : X \times A \to \mathbb{R}$ is the reward function, and γ is the discount rate on future rewards.

A (Markov) policy of the agent is a mapping $\pi : X \times A \to [0, 1]$ so that $\pi(x, a)$ tells the probability that the agent chooses action a in state x. For any $x_0 \in X$, the policy of the agent determines a stochastic process experienced through the instantiation

$$x_0, a_0, r_0, x_1, a_1, r_1, \ldots, x_t, a_t, r_t, \ldots$$

where r_i is the reward received after executing action a in state x_i. In value-function based reinforcement learning the agent maintains a value function $V : X \to \mathbb{R}$, which reflects the expected value of the discounted total rewards collected by starting from state x and following policy π:

$$V^\pi(x) := E_\pi\Big(\sum_{t=0}^{\infty} \gamma^t r_t \;\Big|\; x = x_0\Big)$$

where $E_\pi(.)$ denotes the expectation value of the argument for policy π. Let the optimal value function be

$$V^*(x) := \max_\pi V^\pi(x)$$

for each $x \in X$. If V^* is known, it is easy to find an optimal policy π^*, for which $V^{\pi^*} = V^*$.

Optimal value function satisfies the famous Bellman equations:

$$V^*(x) = \max_a \sum_{x'} P(x, a, x')\Big(R(x, a) + \gamma V^*(x')\Big) \; . \qquad (1)$$

and the optimal policy acts greedily according to V^*:

$$a^*(x) \in \arg\max_a \sum_{x'} P(x, a, x')\Big(R(x, a) + \gamma V^*(x')\Big) \; .$$

One may also define a function of state-action values, or *Q-values*, expressing the expected value of the discounted total rewards collected by starting from state x and executing action a and following policy π onwards. Optimal action selection then becomes $a^*(x) = \arg\max_a Q(s, a)$. It is also true that the optimal policy satisfies the following equation:

$$Q^*(x, a) = \sum_{x'} P(x, a, x')\Big(R(x, a) + \gamma \max_{a'} Q^*(x', a')\Big) \; . \quad (2)$$

There are many alternatives to this setting. For example, one may use a reward function $R(x, x')$ depending on the current state x and the next state x', but not on the action executed.

2.3.2 SARSA learning

We will use the state-action-reward-state-action (SARSA) form (see [9] and references therein), a sampled iterative assignment of the Bellman equation:

$$Q(x_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(x_{t+1}, a_{t+1}) - Q(x_t, a_t)] \quad (3)$$

where α is an update rate, r_{t+1} is the immediate reward received upon arriving to state x_{t+1}, and $\delta_t = r_{t+1} + \gamma Q(x_{t+1}, a_{t+1}) - Q(x_t, a_t)$ is the difference between the currently approximated value of the state x and its approximation based on the next state and the immediate reward received. This is one version of the so called temporal difference (TD) learning methods [11]. SARSA has the advantage that it implicitly learns model parameters P and R; these functions are naturally sampled as the agent interacts with the world.

We take the state space as the Cartesian product of m variables or features: $X = X_1 \times X_2 \times \ldots \times X_m$. We discretize the full feature space and use tabulated temporal difference learning.

2.4 The NewTies framework

NewTies designed an architecture to run multi-agent simulations. An earlier version of the NewTies architecture is described in [2]. The architecture has changed during the project, a concise summary of the current architecture follows. We touch upon only on the parts used in the experiment.

There is a virtual clock set to 1 at the beginning of the simulation. Time passes in discrete amounts, and is measured by the clock in 'time steps'. When one unit of time elapses the world transfers into a new state, the agents act, and the current time step is incremented by one.

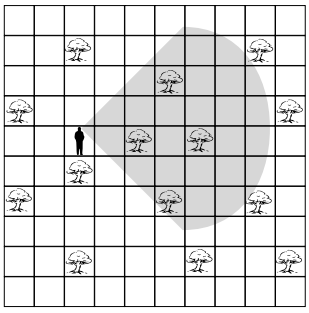

Figure 1: The NewTies environment. The surface is divided into a grid that contains various objects. This example contains some plants and an agent. The agent is facing east, its field of view is 90 degrees in that direction. The agent could turn to face any of the eight neighboring positions and then proceed forward.

The agents are located on a flat, 2 dimensional surface divided into same-sized square regions in a grid pattern called locations. Each agent fits into exactly one location. Locations are referenced with discrete, integer coordinates. The position of each agent is defined by its coordinates and its facing. It can face in any of the eight directions. The agents live in a finite enclosed part of this surface, that can have any shape. Agents can move between adjacent locations.

There can be a number of other objects besides agents on this surface, we used plants and places.

Plants are food sources the agents can eat. They also fit into exactly one location. The most important property of a plant is its energy: the amount of energy it can grant to the agent that eats it. Plants can reproduce in a number of ways depending on the requirements of the concrete simulation, we detail our model in Section 4.1.

Places are large interconnected areas grouping individual locations. They can be of any shape and they are invisible to the agents. We modeled the rainfall patterns with same-sized square-shaped places: the amount of rainfall may differ on each place.

3. THE AGENT ARCHITECTURE

Agents have several attributes, the most important is energy. It is a real number that represents the current well-being of the agent. When it reaches zero the agent dies.

In every time step an agent first perceives its environment, processes the perceptions, then acts upon them. It receives a number of distinct perceptions, among them the most important: it sees every object in a 90 degree arc in front of it, up to a set distance. It also knows its own energy level, and the messages sent to it by other agents in the previous time step.

The agent can perform various actions in each time step. We used the following: (1) `turn left/turn right`: change the direction the agent is facing, (2) `move`: move forward one location, (3) `eat`: if there is a plant in the location of the agent, eat it, and (4) `talk`: send a message to another agent.

The agent architecture has four components. Maps, macros and features help the fourth component, the controller, in its task. Additionally, they are an integral part of the model: they determine what the agent can remember (maps), see (features) or do (macros). In the controller we use reinforcement learning.

Maps collect and store the observations of the agent over time, thus serve as a kind of memory. For example, if a plant gets into the field of view of an agent, and then the agent turns away, it can no longer see the plant. But it takes note on the map and remembers where the plant was, so the agent can decide to collect the plant later. Thus, maps help the agent cope with the problem of partial observability, which severely affects our multi-agent simulation. Agents can retain observations made before. But much of partial observability that springs from the nature of multi-agent systems still remains: when many agents interact, they can not predict the actions of the others. For example when our agent returns to the area it remembers to be full of plants, it might find it completely barren because the other agents have eaten all the plants in the meantime.

Feature generators distill the complex information in the maps and reduce the complexity of the learning problem. A feature is one set in the Cartesian product that is the state space. One such set consists of a few (in our case at most ten) consecutive non-negative integers. The value of a feature is an element from that set. Feature generators generate the value of each feature, the current state is the

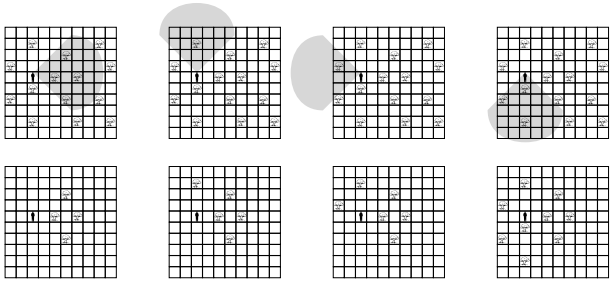

Figure 2: Maps serve as a kind of memory. In the top row the agent is seen turning continually left (in the pictures we included only four of the eight directions). The plants it remembers meanwhile is seen in the second row. It remembers more and more plants in its map as it is turning, in the end of its turn it will know of all the plants surrounding it.

combination of these values. Features are an integral part of the model, as they determine what the controller, and so the agent, can perceive. Features available to our agents are detailed in Section 4.2.

Macros are complex actions consisting of series of the simple actions described in Section 2.4. This way complex functions of the agent are automated. For example, the controller can choose between *find a plant and eat it* and *explore*, and not between simple actions like *go forward* or *turn left*. Macros not only reduce computational complexity, but are an integral part of the model: they determine what the agents can and can not do.

These series of actions are generated by algorithms. For example there is a *go to a food and eat it* macro generator that goes through these steps: (1) look for a high energy food in one of the agent's map, (2) plan a route to that food that goes through shared territories, (3) generate the necessary `turn` and `move` actions to reach the food, and (4) generate an `eat` action to eat the food.

The agents use RL: maps are included into state descriptions through features, macros make the action set. The RL parameters are $\gamma = 0.95$, $\alpha = 0.05$, and $\epsilon = 0.1$.

4. THE MODEL

In this section we detail our theoretical model and its implementation in the NewTies framework. We distinguish three main parts, each in its own section: the environment, the agents and the interactions between the agents. For the environment and the agents we provide the theoretical model first, and then the realization in NewTies follows. For the interactions, the theoretical model and the implementation are the same.

4.1 The environment

4.1.1 Theoretical model

The agents live in a finite, square-shaped enclosed area. They can choose to graze their herds on any location, in that case their (herd's) energy is increased, but the energy stored in the vegetation on that location is decreased. The energy that can be gained from the vegetation is enough that agents can never reach zero energy, that is, they can never die. However, we do model the need for a continuous supply of food, for details see Section 4.2.

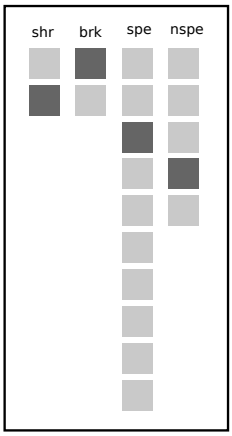

Figure 3: Feature generators distill the complex information in the maps and reduce the complexity of the learning problem. Features make up the state space the controller 'sees', or works with. One column represents one feature. The number of boxes in a column is the number of possible values of that feature, and the darker box is the currently active feature. There are $2*2*10*5 = 200$ possible states of the agent. The states are indexed from 0, the currently active state is $(1, 0, 2, 3)$. Abbreviations: *share*, *break*, *shared plant energy*, *not shared plant energy* – see Section 4.2.

The area is divided into a grid of same-sized square-shaped regions called territories. Rainfall periodically changes in each territory independently. The amount of rainfall is a uniform random number chosen from the interval $[0.5-x, 0.5+x]$, where x is the fluctuation. The regeneration rate of the vegetation on a territory is proportional to the amount of rainfall on that territory. Note that we do not model soil degradation: the vegetation regenerates at a constant rate that depends only on the current amount of rainfall, and does not depend on the amount of grazing that occurred.

Every agent has a home territory it can share and some usufructuary territories, territories it can use. An agent can only move into its home and usufructuary territories, the other territories are closed to it. The details can be found in Section 4.3.

From now on, when we say usufructuary territories of an agent we also mean its home territory, as naturally the agent has usufruct over it.

4.1.2 Realization in NewTies

The agents live in a square-shaped area completely filled with plants: there is a plant on every location. They are the sole source of food for the agents. The plants do not disappear when an agent eats them, their energy decreases by a fixed amount instead. In every time step all of the plants replenish their energy by a little amount. The rate of one plant's replenishing is a linear function of the amount of rainfall on the territory the plant is on.

We used the already mentioned Places to model the weather. Territories were realized as square Places, 5 lo-

cations high and 5 locations wide (Fig. 4). The amount of rainfall was a uniform random number chosen from the interval $[0.5 - x, 0.5 + x]$, where x is the fluctuation. This number is generated separately for every territory in every 10 000th time step.

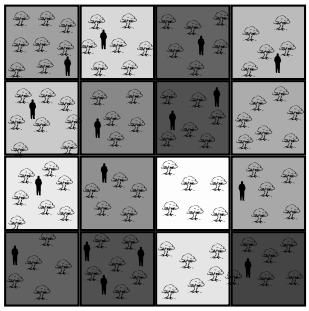

Figure 4: The territories. A scenario with 16 territories. Every territory has a different amount of rainfall represented with different shades.

4.2 The agents

4.2.1 Theoretical model

We think of our agent as a herder who controls a group of animals. The most important statistics of an agent is its energy that measures how well the animals are. So the rational aim of every agent is to maximize its energy. In order to accomplish this an agent is given a set of high level actions, or *macros*. The actions last for variable duration, for example if the agent goes to a location the duration of the action depends on the distance of the location (Fig. 5). Every time an action is finished the agent has to choose another, but it can choose to do nothing (wait). The energy of the agents decreases even if they do nothing, and it decreases faster if they move. Note that in our simulations the herder and his herd make one agent. So if the agent 'eats' then the herder grazes the animals.

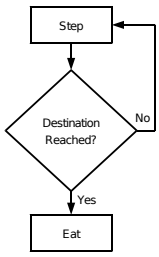

Figure 5: Flowchart of macro '*go to a location with high energy vegetation and graze on it*'.

The agents are informed about the outcome of each of their actions: they know how much energy they gained or lost using an action. They are given no information about the scenario in advance, they have to start exploring the possible effects of their actions by trial and error and form strategies based on past experiences.

The only help our agent gets is that it automatically rejects all share proposals if the energy of the vegetation on its usufructuary territories is less than 60% of the maximum. We used this simplification because learning of whether to accept proposals was not our aim, and would have increased the complexity of the learning problem tremendously.

As mentioned in Section 4.1, our agents can not die, nor do they get hungry, as they always gain much more energy from grazing than they need for survival. However, we do model the need for a continuous supply of food, with the help of reinforcement learning.

Reinforcement learning can look ahead to find sub-optimal actions that eventually lead to high rewards (in fact it implicitly *remembers* the previously experienced and so far optimal action sequence), but the fluctuation changes on a completely different timescale. So our agent can not keep track of the weather, or foresee that it will change, it is 'short-sighted'. It tries to optimize a policy that is short-term compared to the timescale of weather change. In other words it tries to consume as much energy as it can in the short term, so it needs a constant supply of food.

The agent was given the following actions:

- go to a location with high energy vegetation and graze on it (Fig. 5)
- propose a sharing agreement to another agent
- break a sharing agrement
- explore surroundings
- wait (do nothing)

Because of the limitations of reinforcement learning we were constrained in the amount of information we could give to our agent. We tried to give it the minimum information we think a human would require to decide:

- whether there is anyone to share with
- whether there is anyone to break the share agreement with
- the average energy of the vegetation in all the usufructuary territories (territories the agent can go into and graze in)
- the average energy of the vegetation in all the territories whose owner the agent could establish a share agreement with

4.2.2 Realization in NewTies

The agent tries to accumulate the maximum amount of energy possible. This is realized by the reward: the reward after each action is the difference in the energy of the agent before and after that action, that is, the energy gained or lost. Reinforcement learning is capable of finding action sequences where suboptimal actions at a given time instant (e.g., share) may lead to high rewards later (eat), so we suspected that even though share is an action that is not beneficial in itself, if the territory opened when it contains high energy vegetation, the agent will learn that it is a beneficial action.

In Section 3 we described the blueprint of our agents. Now we fill in the details, enumerate the concrete features and macros used. These define what the agent can perceive and how it can act. Maps are not enumerated, as they are not perceived directly by the agent (controller).

We used the following features:

- '`share feature`': 0 or 1. It is 1 if and only if the execution of a share macro would most likely be successful in this time step. That is, if the agent can see one of its neighbors with whom it has not already shared its territory. A share action will only be successful if the partner agent has more than 60% of the maximum resource possible on its shared territories. This information is not encompassed into this feature, the agent does not know it.

- '`break feature`': 0 or 1. It is 1 if and only if the execution of a break macro would be successful in this time step. That is, if the agent can see another agent with whom it has an agreement and who is not on the agent's home territory.

- '`average shared plants energy`': 0, ..., 9, the discretized average of the plants' energy on the usufructuary territories of the agent.

- '`average neighbors' not shared plants energy`': 0, ..., 4, the discretized average of the plants' energy on the neighboring territories the agent can not currently enter.

We used the following macros:

- '`go to the best food and eat it`': the agent goes to one of the foods with high energy on its usufructuary territories through its usufructuary territories and eats it.

- '`share home territory`': if the agent can see a neighbor with whom it does not have a sharing agreement, then it initiates one to share their respective home territories with each other. They also tell each other which territory they own. After their first interaction they will know this for the length of the simulation.

- '`break a share agreement`': if the agent can see another agent with whom it has a sharing agreement, then they break their agreement.

- '`explore the surroundings`': the agent turns a few times in a random direction then moves forward through a few time steps

- '`wait a time step`': the agent waits (does nothing) for a time step

4.3 The agreements between agents

We mentioned in the introduction that agents start as ranchers. Every rancher starts on a distinct territory, and they are confined to this territory, their home. They own it and can never lose it. In addition they can grant usufruct rights to other agents, if requested. They also gain usufruct rights to the home territory of the other agent in turn. This process is called sharing, because agents share their home territories. Agents can walk and graze their herds on all their usufructuary territories, so if there are enough of them then there is the possibility of transhumance.

Agents only initiate sharing with their neighbors: that is, *Agent 1* only initiates sharing with *Agent 2* if *Agent 2*'s home territory and one of *Agent 1*'s *usufructuary* territories have a common border. Otherwise *Agent 1* would not gain anything because it would not have a route to *Agent 2*'s territory.

The other agent only accepts a share proposal if it has already enough food for itself: we chose 60% of the maximum amount of food possible, because 50% (the expected value) is just enough for the agent, so it sets a safety margin.

The procedure of sharing is the following:

1. *Agent 1* requests sharing
2. If *Agent 2* has more than 60% of the maximum possible plant energy on his shared territories, it answers with yes. If it has less, the answer is no, because it would endanger its own survival.
3. From now on they can both move to and eat from the home territory of the other.

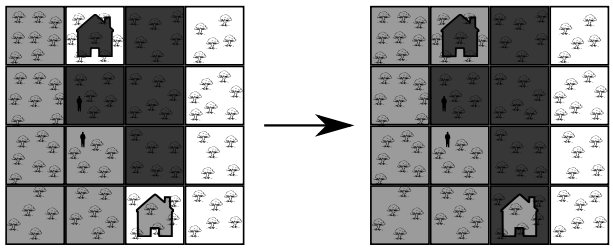

Figure 6: The process of sharing territories. There are two agents on the figure. Their respective home territories are represented by the two houses in different shades of gray. An agent can enter only its home territory and the territories with whose owners it has a sharing agreement. This is represented with the two shades of gray: the agent whose home territory is colored dark gray can only enter the dark home and dark colored areas (usufructuary territories) and vice versa. The home territory of the other agent can be entered only upon sharing.

Sharing agreements do not time out, they last forever. But they can be broken. In fact it is easier to break an agreement than to establish one, because the breaking of one does not require the more than 60% plant energy requirement the acceptance of one requires. An agent can break an agreement any time, the only constraints are that it must see the other agent it wants to talk to, and the other agent can not be on the home territory of the agent that breaks sharing at that time instant.

The procedure of breaking an agreement is the following:

1. *Agent 1* initiates the breaking of an agreement
2. *Agent 2* accepts it if it is not on *Agent 1*'s home territory.
3. From now on they can not step on the home territory of the other.

5. RESULTS AND DISCUSSION

We examined two scenarios. The first scenario consisted of 16 agents, each starting on its own home territory. Their home territories were placed onto a 4×4 chessboard (Fig. 4).

In the second scenario there were 25 agents and 25 territories on a 5×5 chessboard. The length of the scenario was 50 000 time steps. This was long enough for the learning algorithm to be able to learn (to visit the states enough times), and short enough to make the time required to run one simulation feasible.

We have run the simulation 10 times for each value of the rainfall fluctuation from 0.00 to 0.50 with increments of 0.01, then computed the average and standard deviation for each value.[2]

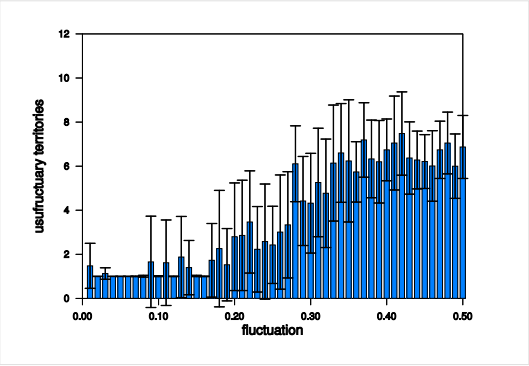

Figure 7: Number of usufructuary territories per agent as a function of the fluctuation. It can be seen that the agents share with more partners and create larger common territories as the fluctuation increases (16 agent scenario).

We got the same results on both of the scenarios. Figure 7 shows the average number of usufructuary territories per agent as a function of the fluctuation. If the fluctuation is low the agents do not share their territories. But as the fluctuation rises, the agents start to share. The number of usufructuary territories per agent is increasing, but so is the standard deviation: the system is unstable. As the fluctuation is above approximately 0.4, the number of usufructuary territories is constantly high, and the standard deviation is considerably smaller: the system becomes more stable. The agents gain usufructuary rights to 6-7 territories at most, out of the whole 16. For the scenario with 25 agents the usufructuary territories per agent rose a little bit, but not considerably: it is between 7 and 8. Although now there are 150% more territories, the number of usufructuary territories does not rise significantly. This may be because there is an optimal number of usufructuary territories per agent regardless of the size of the scenario.

It is also interesting to see the average plant energy: how much energy does the vegetation store at the end of the simulation, in other words how much energy do the agents conserve? In both scenarios (Fig. 8) the average plant energy rises as the fluctuation rises: the agents conserve more and more energy.

This is not necessarily a positive thing in our model, as the energy collected by our agents drops as the fluctuation rises. We think that there are three causes for this: first, scenarios with high fluctuations are much more difficult because of local variations, second, we did not model soil degradation,

[2]Other parameters can be found in the supplementary material `http://people.inf.elte.hu/lorincz/hparams.pdf`

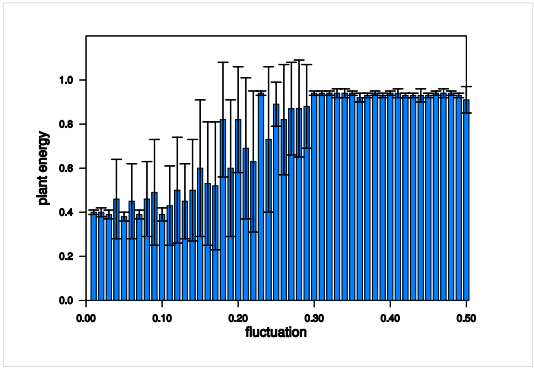

Figure 8: Average plant energy as the function of the fluctuation. Agents conserve more and more as the fluctuation rises (25 agent scenario).

and third, the way the agents choose a location to graze on. Figure 9 shows the energy collected by the agents as a function of fluctuation.

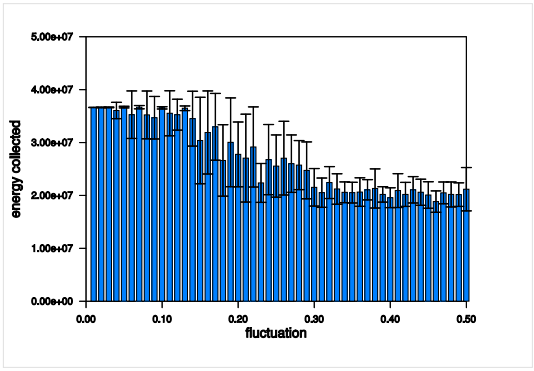

Figure 9: Energy collected by the agents as the function of the fluctuation (25 agent scenario).

If the average fluctuation is 0.0, then agents basically only have to eat, and do nothing else. If it is 0.5, then there are local difficulties that the agents have to face. Even though the average plant energy is the same as in the former case, it happens a lot that the bulk of that energy is not accessible to one agent. For example, if the agent faces a severe, long lasting drought (e.g. the rainfall coefficient is 0.1 in the agent's home territory) then it could perish despite the fact that on average there is a lot of food around. Because the agent optimizes short-term behavior in order to 'survive' (see Section 4.2), it has to wander more than in the case with small fluctuation. At the same time, if an agent eats only on its home territory, then it has more time to eat than an agent that has to move to the territory with more food.

The second cause is that we did not model soil degradation. If perpetual grazing would degrade the soil as in real life it does, then agents doing nothing but eating would eventually gain much less energy than the agents that wander from territory to territory, and eat only high-energy plants.

The third cause is that when agents graze they always choose one location they remember to have high energy, and

they move to that location. Clearly the average distance they travel grows as the number of their usufructuary territories grows. So even if we would create a controller that always chooses the *go to a high energy food and eat it* macro, it might occur that the agent who can not leave its home territory consumes more energy than the agent with several territories, because the former would travel less between *eat* actions.

There is a trend in the standard deviation that can be observed in all our measurements: it is small at values of low fluctuation, larger at middle values and small again at high values. At low values of fluctuation, the agents do not share their territories. At high values the agents always share with many partners. At middle values the system is unstable: the agents either share, as at high values of fluctuation, or they do not, as at low values. The system ends up in one of these possibilities.

There is a clear connection between the four main variables: the *rainfall fluctuation (F)*, the *usufructuary territories per agent (U)*, the *average plant energy (P)* and the *average energy collected per agent (E)*. There is a strong interdependence between U, P, and E, as the correlations show: 0.98 between U and P, -0.99 between U and E, and -0.98 between P and E. In the 25-agent scenario the coefficients are 0.98, -1, -0.98. We can say that the fluctuation determines the other three.

When U is large we talk about common ownership, because if one agent can use 7-8 territories on average then obviously one territory is used by 7-8 agents on average because there are the same number of agents as there are territories.

The agents established common territories, and with the help of these they managed to overcome the fluctuation. The expected value of rainfall is the same on all of the territories regardless of F. The fluctuation only creates local variations. So the more territories an agent has access to the closer the average amount of rainfall on all these territories is to this expected value, so the less the fluctuation affects the agent.

There are two main ideas present in this phenomenon. One is risk management or insurance. Basically when agents establish sharing agreements they insure themselves: they agree that they share their territories so if rainfall is low on either of them both can survive. The more sharing agreements an agent has, the more insured it is. They cope with local fluctuation by trying to be more 'global': to have enough territories so that fluctuation does not affect them.

The other idea is Adam Smith's invisible hand [10]: each agent intends only its own gain and promotes an end that was not part of its intention, but is good for the community as a whole. Every agent tries to maximize its own energy, but in doing so they insure themselves and the other agents, so the whole agent 'community' is insured against the fluctuation of the rainfall: if rainfall is low on a territory, they simply move somewhere else. The point is that this insurance is not deliberate, the agents are selfish and despite that achieve an outcome that is good for all of them.

6. CONCLUSIONS

We have demonstrated conditions where adaptive agents established common property even though they maximized their own gains, and did not consider the effect of their actions on the other agents.

We described why Hardin's *tragedy of the commons* is not applicable to the conditions in Sub-Saharan Africa. Modeling real-world conditions we constructed a learning scenario where the 'tragedy' did not occur, although all our agents were autonomous rational agents (they considered only their own benefit), just like Hardin's herdsmen. The most important characteristic of this scenario was the fluctuation of the regeneration rate of resources. The fluctuation was present in both space and time, and as an agent needed the resource in the short term, they could only choose to cope with the fluctuation in space. They established areas where they could freely move and graze, so they could always wander to a territory with high resources.

7. ACKNOWLEDGMENTS

Thanks are due to Nigel Gilbert for helpful discussions. This research has been supported by the EC FET 'New Ties' Grant FP6-502386. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of other members of the consortium or the European Commission.

8. REFERENCES

[1] R. Behnke and I. Scoones. Rethinking range ecology: Implications for rangeland management in Africa. *Int. Inst. for Envir. and Development Paper No. 33.*, 1992.

[2] N. Gilbert, M. den Besten, A. Bontovics, B. G. W. Craenen, F. Divina, A. E. Eiben, R. Griffioen, G. Hévízi, , A. Lőrincz, B. Paechter, S. Schuster, M. Schut, C. Tzolov, P. Vogt, and L. Yang. Emerging artificial societies through learning. *J. of Artificial Societies and Social Simulation*, 9(2):9, 2006.

[3] V. Gyenes, Á. Bontovics, and A. Lőrincz. Factored temporal difference learning in the New Ties environment. *Acta Cybernetica*, 18:651–668, 2008.

[4] G. Hardin. The tragedy of the commons. *Science*, 162(3859):1243–1248, December 1968.

[5] M. Hulme and M. Kelly. Exploring the links between desertification and climate change. *Environment*, 76:4–45, 1993.

[6] A. Lőrincz, V. Gyenes, M. Kiszlinger, and I. Szita. Mind model seems necessary for the emergence of communication. *Neural Inf. Proc. Lett. Rev.*, 11:109–121, 2007.

[7] N. Rass. Policies and strategies to adress the vulnerability of pastoralists in Sub-Saharan Africa. *PPLPI Working Paper No. 37., FAO*, 2006.

[8] W. Schultz. Getting formal with dopamine and reward. *Neuron*, 36:241–263, 2002.

[9] S. Singh, T. Jaakkola, M. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38:287–303, 2000.

[10] A. Smith. *The Wealth of Nations*. Bantam Classics, March 2003.

[11] R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[12] I. Szita and A. Lőrincz. Factored value iteration converges. *Acta Cybernetica*, 18:615–635, 2008.