A THEORY-BASED DYNAMICAL MODEL OF INNOVATION PROCESSES

David Lane, Roberto Serra*, Marco Villani, Luca Ansaloni

Department of Social, Cognitive and Quantitative Sciences
Modena and Reggio Emilia University
Via Giglioli Valle 9, I-42100 Reggio Emilia

MODEL OF INNOVATION PROCESSES

Key words: multi-agent based model, innovation processes, economy, design of complex artificial systems

* Corresponding author:
prof. Roberto Serra
Department of Social, Cognitive and Quantitative Sciences
Faculty of Communications and Economics
Modena and Reggio Emilia University
via Giglioli Valle 9; I- 42100 Reggio Emilia - Italy
tel. +39-0522-522433 (operator 522511), fax 522512
e-mail: serra.roberto@unimore.it

Abstract

We present an agent-based model of innovation processes, based upon a theory of innovation by Lane and Maxfield. The theory inspires and constrains the features of the model, thus reducing the *embarasse de richesse* that is one of the major methodological problems of agent-based modeling. Artifacts are produced by agents using recipes; the basic dynamics, absent innovation, is one of production and sales, where the external world supplies "raw materials" and external demand. Depending upon the initial conditions, self-sustaining cycles of production and exchange can emerge among the agents. Innovation – that is, the generation of new recipes, in particular desired directions, called "goals" – results in substantial modification of the system dynamics. Two innovation regimes are introduced: a "lonely" mode, in which each agent tries to introduce new products by itself, and a "relational" mode, in which two agents can improve their reciprocal knowledge and can decide to try to jointly develop a new artifact.

1 Introduction

Modelling social and economic phenomena is a major challenge. In recent years agent-based models (ABMs) have attracted considerable attention, as they allow one to escape from the constraints of the "representative agent" and to account for agent heterogeneity (see e.g. [1-6] and further references quoted therein) . ABMs are well suited to bridge the gap between hypotheses concerning the microscopic behaviour of individual agents and the emergence of collective phenomena in a population composed of many interacting agents. Moreover, by requiring that hypotheses be stated precisely, ABMs may improve the development of the theory itself

Attractive as agent-based models may be, they still face formidable problems. In particular they allow the modeller too much freedom with respect to the design of agents and their interactions. ABMs often have too many variables and parameters. Moreover, while in some cases experimental data for model testing are available, in many interesting cases data are scarce and do not sufficiently constrain the model. In these cases, what is needed to get meaningful results is a principle which limits the degrees of freedom of the modeller. While Ockham's razor is of the outmost importance, it does not seem to be sufficient by itself for the purpose of limiting the *embarasse de richesse* of ABMs.

In this work we consider another approach, namely that of relying upon an existing theory of the phenomenon in order to constrain our modelling options. The theory, which is qualitative in nature, provides the basic entities of the model and predicates about their relationships. The model represents a simplified universe inhabited by (some of) the theoretical entities. Simplification is necessary in order to deal with manageable systems: we do not look for an all-encompassing model, but we rather imagine a family of models which capture different features of the theory.

The phenomenon which is considered in this paper is that of innovation, and the theory we refer to has been initially developed by Lane and Maxfield [7,8], and is under further development within the UE-FET project Iscom (Information Society as a Complex System:), whose approach is based on the interaction between theory development, case studies and model (see the project website www.iscom.unimo.it). We will refer for brevity to this theory as the Lane-Maxfield (LM) theory, while the model will be called I_2M (Iscom Innovation Model).

A very brief outline of some of the major features of the LM theory will be presented in section 2. The theory constrains the model, and the main constraints will be summarized at the end of that section. Section 3 describes the main features of the model. Section 4 describes the characteristics of a software simulator that implements parts of the model, while section 5 presents the results of some simulations. Finally, section 6 contains comments on the results achieved so far and indications for further work.

2 Indications from the theory

In the work of Lane and Maxfield [7,8], artifacts are given *meanings* by the agents that interact with them, but these meanings cannot be understood without taking into account the *roles* which different agents can

play. Thus, artifacts may be given different meanings by different agents, or by the same agents at different times.

Therefore, LM theory can be seen as a theory of the interpretation of innovation. According to LM, a new interpretation of a potential artifact functionality can be put forth in the context of so-called generative relationships. By interacting, a few agents come to invent and share this interpretation, based on the discovery of different perspectives and use of existing artifacts. The generative potential of a relationship may be assessed in terms of the following criteria:

- heterogeneity: the agents are different from each other, they have different features and different goals; the heterogeneity is not so intense as to prevent communication and interaction
- aligned directedness: the agents are all interested in operating in the same region (or in neighbouring regions) of agent-artifact space
- mutual directedness: the agents should be interested in interacting with each other.

Moreover, Lane and Maxfield discuss two further features that deal with organizational issues, i.e. permissions and action opportunities. The former refers to the fact that the agents are authorized to engage in such a relationship, the latter to the possibility of moving from "talk" to action.

Lane and Maxfield further argue that, in a situation where innovations happen at a very fast pace, predicting the future is impossible; so a better strategy would be to identify those relationships that have the potential for generativeness, and to foster them in order to effectively explore the new opportunities they can give rise to. It is therefore very important to be able to estimate the "generative potential" of the existing and prospective relationships.

The LM theory of innovation is highly sophisticated in describing the interactions between different players in innovation processes, and it cannot be entirely mapped onto a specific computer-based model. Therefore, the modelling activity aims at developing models that are based on abstraction of some key aspects of this theory, which is of a qualitative nature.

The basic requirements for the model derived from the theory can be summarized as follows:

1. the meanings of artifacts must be generated within the model itself: since the LM theory claims that new meanings are generated through interactions among agents and artifacts, it would be inappropriate here to resort to an external oracle to decide a priori which meanings are better than others
2. the roles of agents must also be generated within the model: indeed the LM theory claims that also new roles are generated through interactions among agents and artifacts
3. agents must interact with artifacts and with other agents: interacting with artifacts only would prevent the possibility of describing agent-agent relationships
4. an agent should be able to choose the other agents with whom to start a relationship; in general, an agent will be able to handle a finite number of relationships at a time, and it will choose a subset of the other agents as its partners. Agents must be allowed to cut a disappointing or unsatisfactory relationship to look for a better one
5. an agent should be able to have different degrees of interaction with another agent: in this way it will be possible to tune the intensity of these relationships
6. some agent-agent relationships should be generative in character, i.e. they should be able to lead to new attributions (for agents and/or artifacts) and should respect the criteria for generative potential described above.

3 Model description

We introduce a model where innovation happens in an "artificial world" [9], which is inhabited by agents which "produce" artifacts, which in turn can be used by other agents to build their own artifacts, etc. An agent can produce several artifacts for different agents (and it can sell one type of artifact to several different customers).

The meaning of artifacts is just what agents do with them, while the role of agents is defined by which artifacts they produce, with whom, and for whom. The role of agents is also partly defined by the social network they are embedded into. In order to describe these features, and to have them generated in the model, without being enforced *a priori*, we envisage a network of agents that are producers of artifacts, which are in turn used by other agents.

So the "strong ties" between two agents are mediated by a chain of artifacts: for example, agent A1 produces artifact O1 which is used e.g. by agent A2 to produce artifacts O2 and O3, and by agent A3 to produce artifact O4. O2, O3 and O4 are used by other agents, etc. If we describe the agent network, then there is a strong (directed) link from A to B if A produces an artifact which is used by B: the network of agents may well have loops, and an agent may have multiple outgoing and multiple incoming links.

There are also weak ties between two agents ("acquaintances") which refer to the fact that agent A knows something about agent B (e.g. its products, or part of its providers): so the social neighbours of a given agent are a superset of its providers and customers.

A key point is the structure of artifact space. For modelling purposes, we have considered different alternatives: binary coding as it is done in classifier systems, or λ-calculus as in the Alchemy model [10, 11] or numbers (either natural or real). What is required is that the space has an algebraic structure, and that suitable constructors can be defined to build new artifacts by combining existing ones. We concentrated on the real number representation and the use of mathematical operators because it is more compact than the binary representation and simpler than the predicate calculus and the λ-calculus. A full comparative evaluation of the features of the different representations lies beyond the purpose of this paper.

Therefore the agents are "producers" of numbers by combination of other numbers. The mathematical operators which are available to an agent represent its "technology", so a major technology shift might be described by the appearance of new operators, like e.g. the discovery of division in a world were addition and subtraction were the only available technologies.

It is well known that technological constraints limit the set of possible artifacts, a feature which is sometimes described by defining a fitness landscape on the artifact space; however, in our model the value of an artifact must be related to what other agents do, so assigning *a priori* a given fitness would not be a suitable alternative.

A more precise description of the model is given below.

3.1 Entities description

Artifatcs

The artifact Ar_i ($i \in [1,N]$) is described by:

- the identifier $i \in \mathbf{N}$, which indicates in a unique manner the artifact itself
- the integer value $name \in \mathbf{N}$, which identifies the artifact type and is the main value manipulated by the agent recipes
- the identifier of the agent that produced it
- the level of its stock
- the identifiers of the agents that acquire it.

We foresee the possibility to develop in the future a richer representation of the agent type, by resorting to a vector of integers (or real numbers) instead of a single scalar quantity

Agents

The agent Ag_j ($j \in [1,M]$) is described by:

- the identifier $j \in \mathbf{N}$, which indicates in a unique manner the agent itself

- the number of recipes nrecipes$\in \mathbf{N}$ owned by the agent
- the number of goals ngoal$\in \mathbf{N}$ and the list listgoal$\in \mathbf{N}^{ngoal}$, "owned" by the agent
- the "catalogue", i.e. a list of existing recipes, available to the agent
- the list of the agent acquaintances
- the list of the known artifacts
- the real vector vote$\in \mathbf{R}^{M}$, which measures the quality that the agent attributes to each relation (with other agents) it maintains
- the real value activity$\in \mathbf{R}$, which measures the "wealth" of the agent
- a set of parameters that control the agent propensity to innovate, to enlarge its own acquaintances, to participate to joint projects, and so on (in the future, we will sometimes refer to this set of parameters as the agent "style").

Recipes

The third object we define is the production recipe. The k-th recipe of the agent Ag_j, Rc_{jk} ($k \in [1, Ag_j.nrecipes]$) is described by:

- the number ninput$\in \mathbf{N}$ of needed inputs
- the ordered list of identifiers of the input artifacts
- the ordered list of operators, whose number is equal to ninput-1
- the identifier of the product
- the integer value prod$\in \mathbf{N}$, which states the number of tokens created at each recipe production
- the real value cost$\in \mathbf{R}$, which measures the cost required to produce the product associated to that recipe
- the time tlast $\in \mathbf{N}$ elapsed from the last time the recipe has been active, i.e. actually used for production.

It is useful to define a further entity, the recipe support, which is simply a subset of recipe features:

- the number ninput$\in \mathbf{N}$ of needed inputs;
- the ordered list of operators, whose number is equal to ninput-1.

This concept allows the separation between the "fundamental" structure of one recipe (i.e. the recipe support, which stores the construction modalities that are the basis of the recipe itself) and one of its instantiations (which leads from the particular input artifacts to the output one).

External environment

The model presented here is not necessarily a closed one, even if, as we will see, it can behave in a self-sustaining mode, without need of external inputs. However, in order to allow a wider generality of the model, we will assume that there may be an "external world", which represents other agents which are not included in the model (external customers) or artifacts which are not produced within the model (e.g. raw materials). We can call all these entities "environment".

More specifically, we identify:

- a set of "raw materials" (artifacts whose quantities are neither limited nor associated to some particular agent; they can be used as input to production recipes)
- a set of "external reward regions", areas where artifacts are acquired by entities not belonging to the simulated market system.

3.2 Model dynamics

We first provide an overall description of the model behaviour. Agents transform artifacts; in agent-artifact space they are defined by their production recipes and by their relationships with other agents. Each agent can produce more than one product, and for each product there is at least one production recipe. For simplicity we state that at each simulation step only one recipe is active for a given product. Agents have also "goals", that is, particular names that the agent aims to produce. Furthermore, agents are able to participate to joint ventures, to enlarge their acquaintances, to give birth to new agents.

A schematic sketch of the possible agents' actions is given here below.

At each time step (updating is asynchronous):

1) a randomly chosen agent is selected for updating
   a) for each recipe, the agent looks for the required inputs
      I) if the inputs are found (i.e. they have nonzero stock level), the recipes outcome is produced and inserted immediately into the stock, while the corresponding input stocks are decreased
      II) if the inputs aren't found, the agent searches, among the artifacts known to it, another object with the same name, whose stock level is different from zero
         (i) if this artifact is found, it substitutes the old one; return to step (1.a.I)
         (ii) if this artifact isn't found, pass to another recipe; in this case, a counter associated to the unused recipe is incremented (this counter will be used to cancel recipes which have not been used for several times)
   b) the following actions need not be performed at each time step. Depending upon the value of suitable parameters, and upon the level of its strength, the agent may perform one or more of the following (the decision about which ones is taken in a stochastic way):
      I) try to innovate by itself its catalogue, by:
         (i) choosing a goal
         (ii) trying to realise it, i.e. to find a recipe for the goal (or for a product close enough to the goal, see below section 3.2.1)
         (iii) producing it, if a successful recipe is found
      II) try to innovate its catalogue together another agent, by:
         (i) choosing a partner
         (ii) with the partner, choosing a goal
         (iii) with the partner, trying to realise it
         (iv) sharing the obtained results, if the innovation process succeeds
      III) try to enlarge its acquaintances
      IV) cancel the less useful recipes

   c) The agents' activations are updated.

Moreover, new agents can be created, according one of the following rules:

   d) randomly, with a given probability
   e) in order to substitute the dead agents
   f) according to the internal dynamics.

Now we present in detail the actions just sketched above.

3.2.1 Goals

In order to survive (i.e. to keep its strength above a given threshold) it is useful for an agent to find artifacts that could have high sales. Therefore identifying a proper goal (the choice of the artifact that has to be built) is crucial for the agents. When its turn comes, an agent which is reach enough (that is, whose strength is high enough) can try to create a new recipe. Let us consider "lonely" agents, which try to do this basically by themselves; the agent:

- looks at the portion of artifact space which it knows
- updates the set of its goals
- chooses a specific goal
- tries to come close to that goal by using the available operators on the available inputs and recipes
- if, using the available inputs, it comes close to the goal (within a given range) it puts the corresponding recipe in the set of active recipes.

Heuristics for goal setting include:

- locate a region of artifact space which is populated by several artifacts (and that therefore is likely to be able to sustain a high number of them)

- explore an existing reward region looking for higher reward points
- explore new regions of artifact space, e.g. by mutating an existing artifact, by interpolating between some of them, etc
- respond to the request of another agent
- examine another agent's recipes and find some input product which can become a goal for the first agent.

3.2.2 Recipes

Agents transform artifacts by means of their production recipes. Each agent can produce more than one product, and for each product there is one production recipe. A recipe is a function Rc_{ij}:In→Out, where In=$\{x_1,x_2,\ldots,x_{ninput}\}$ is the vector of the names of the input artifacts and Out is the name of the product.

The total transformation is composed by a set of simpler transformations, called operators; the presence (or absence) of particular kinds of operator defines the "technological level" achieved by the recipe. The operators are not limited to the basic numerical operators (addition, subtraction, multiplication, division), but more generally are any sort of functions Op:E⊆$\mathbf{R}^m$→$\mathbf{R}$, like for example exponential, mean, maximum, or function restrictions (where the operand domain is restricted to certain intervals, e.g. addition with names∈[0,120], multiplication between artifacts with names ≤10, etc.). If we assume that Op=$\{op_1,op_2, \ldots , op_{ninput-1}\}$ is the list of the operators, the result of a recipe could be written as:

$$Out=((((((x_1.op_1.x_2).op_2.x_3).op_3.x_4).op_4.x_5)\ldots).op_{ninput-1}.x_{ninput})$$

Note that, in the present notation, the order of the application of operators is from op_1 to $op_{ninput-1}$. Changes of the order of the input artifacts, as well as changes of the operator's order, generally lead to changes in the recipe output. Of course, changes of the names of the input artifacts also lead to changes in the recipe output. All these alterations nevertheless could be done without modifications of the recipe support, which remains the fundamental structure of the recipe.

Agents are capable of changing the recipe repertoire by:
1) eliminating one recipe and the corresponding product (at no cost)
2) introducing a new recipe, by means of:
   a) the change of the providers (new recipe, same recipe support, same input names)
   b) the change of the order of the inputs (new recipe, same recipe support);
   c) the change of the order of the operators (new recipe, new recipe support)
   d) modification of the choice of operators and possibly of their number (different recipe, different recipe support).

It is possible to introduce new recipes with at least two different approaches:
- acting alone and using only one's own recipes
- initiating a collaboration with another agent and combining the knowledge of both partners.

A recipe may be active as long as it has input artifacts; of course, if one or more of its inputs are absent from the system, the recipe does not operate and its output is not produced. A recipe whose product doesn't have any client for a certain number of time steps is removed and the product disappears.

When a new recipe is first introduced, the corresponding agent inserts its product in its catalogue, making it known to its acquaintances. If at least one customer is found the recipe is activated, and remains active as long as it has at least one customer. The first time no customers are found the recipe does not act, although it remains in the accessible catalogue for some time t_{forg}. Recipes that have been constantly inactive for a certain number of steps are removed ("forgotten").

Recipes which are regarded as useless (e.g. the recipes that don't give a sensible contribution to the agent activity) or even damaging (those that give a negative contribution) may be removed by the owner, in a way

whose details (e.g. parameters) depend upon the agent "style". Moreover, an agent could choose to eliminate the less performing recipes to leave room for more performing ones.

Note also that, in the model described so far, as well as in the simulations shown in section 5, artifact space is "flat", i.e. uniform, although some regions may be unreachable at a certain time because of lack of proper inputs or operators. The model could however also deal in a straightforward way with intrinsic difficulties in reaching some regions of artifact space ("technical barriers"), by modulating the success probability of an innovation using some suitable external function F(x) (e.g. |sin x|): when an innovation is tried, the probability of success is proportional to F(x).

3.2.3 Joint developments of new products

In order to align and coordinate their actions, two or more agents, which have goals which are close in artifact space, can cooperate; if one agent alone is not able to reach the interesting region, the combined efforts of two agents could be successful.

A way to combine the efforts is that of sharing the agents' knowledge (input artifacts, recipes). The agents could create a new recipe (for example combining their recipes by means of genetic operators) which is able to reach the interesting region; after this action, they could divide the obtained recipe into two different recipes in such a way that the product of the first recipe is one of the input artifacts of the second recipe. Finally the first agent becomes the owner of the first recipe and the second agent becomes the owner of the second recipe (one recipe for each agent). In such a way they are able to share the success of the research results.

3.2.4 Artifacts stock level

Artifacts are characterised, among other variables, also by their stock levels, that is the "quantity" of the available tokens. When a recipe uses one artifact the corresponding level decreases, whereas when the appropriate recipe produces the artifact, the corresponding level increases.

Generally speaking, a recipe combines a certain amount of each input product to generate a certain amount of output. A precise quantitative description of these aspects would lead to further complications. Furthermore, we expect that this complication is related more to the specific aspects of the production chain than to the model aims, i.e. the study of innovation processes. On the other hand, we feel it necessary to maintain the fundamental distinction between types (i.e. names of the artifacts) and tokens (i.e. quantities). This is an example of how the theory constrains the model! In order to meet this need while keeping the complications at a minimum, we consider recipes where the quantity of each input is one, while the quantity of the output product is a property of the recipe itself (described by the variable "*prod*").

If the number of produced tokens is equal to the number of inputs, the total number of tokens is conserved; otherwise, this number is allowed to change. Therefore, if the value of this variable is greater than the number of inputs the total number of artifacts (tokens, not types) will tend to increase in time.

The agents control the stock levels: their aim is to avoid stock depletion and to avoid a useless surplus. There are several methods to obtain these aims:

- to allow production only when the stock level is equal to zero
- to allow production only when the stock level is lower than a given value (e.g. *prod*)
- to allow production depending on the artifact sales (e.g. by comparing sales trend with stock levels).

3.2.5 Activation dynamics and agents' wealth

A real variable, called activation, is associated to each agent, and measures its "wealth". At each time step, the agent activation is updated; it receives positive contributions from those products that are sold, while it has to "pay" for all its recipes. Whenever an agent performs a change (e.g. by introducing a new product or changing a recipe) it has to pay some activation in order to be allowed to try. A stochastic mechanism then

determines whether the attempted modification happens or not: this simulates the effort that can be put in R&D to discover new recipes.

The formula that describes activation dynamics is:

$$Ag_j.act(t+1) = Ag_j.act(t) + \sum_{k \in sold.products} n_{jk}(t) - Ag_j.act(t) \cdot \sum_{i \in active.recipes} Rc_{ji}.prod \cdot Rc_{ji}.\cos t - \lambda Ag_j.act(t)$$

[Eq. 1]

where $n_{jk}(t)$ indicates the number of sales of product k (belonging to agent j) at time t. The final term is necessary to make the activation of an agent which never acts vanish.

Of course, different formulae for the activation dynamics can be tested; simulation results will provide indications about the better ones.

Agents whose activation is equal to or smaller than zero are removed from the system. New agents could be generated, according to rules whose definition depends upon the aims of the simulations. For example, some suitable rules are:

- to substitute the removed agents (in order to maintain the agents' number)
- to introduce new agents at each step according to a given probability distribution (in order to simulate the appearance of new agents coming from outside the system)
- to introduce new agents according to internal dynamics, for example depending upon the total system wealth, or the wealth of a particular agent that can generate some spin-off.

3.2.6 Relationships among agents

Agents can use their knowledge about agent-artifact space, which is in general limited (an agent may not know all the existing artifacts, all the existing agents, all the goals and the recipes of the known agents).

An agent can also generate new goals or remove some old ones, with the aim to increase its activation. The definition of a new goal can be based upon the knowledge of the other agents' requirements and of the existing products, and upon the requirements of the external world.

An agent is also endowed with a "social network". There are two different kinds of interactions among agents: i) relationships with providers and customers clients, mediated by artifacts, and ii) relationships with the agent's acquaintances ("weak ties"), in which some exchanges of information can take place.

Each agent has its own measure of the quality of its relationship with other agents, which increases with the frequency of the positive events that take place during their interactions (purchases, sales, exchange of information, etc.), while it decreases with the frequency of negative events (wrong information, mistakes, poor activity income, failure in joint developments, etc.).

A reciprocal high vote could allow two agents to:

- exchange information (acquaintances, goals, operators, ...)
- share (part of) their recipes
- create alliances in order to reach interesting zones of agent-artifact space.

Agents can invest part of their activity, in order to:

- enlarge their acquaintances
- look for new providers
- improve their knowledge of some agent with whom they hold a high-ranked relationship
- increase the number of artifacts they know.

To improve their relationships, an agent can inform other agents about (some of) its desired inputs. If the interaction has been successful, the agent lets its providers "know more" about itself - e.g. other desired

inputs, or its production recipe(s). Moreover, reciprocity leads providers to inform their customers about themselves.

4 The prototype simulator

In order to test the behaviour of the model and the suitability of some mechanisms that have been proposed, we developed a preliminary, simplified version of the simulation environment, which has three main modules:

- a specialised initial condition builder, whose aim is to create the initial network of agents and artifacts
- the simulation engine
- a tool to visualise the main simulation results.

The interaction among the modules allows one to simulate a wide set of situations and conditions. The environment has been developed in C, while the visualization module has been developed by means of scripts that use the R environment. The simulations have been performed on personal computers and on clusters of PCs.

The model outlined above is rather complicated and the presently running software adopts some suitable simplifications, in order to test the appropriateness of some mechanisms that have been proposed. The main differences between the model described in section 3 and the current version of the simulation environment are the following:

- updating is synchronous
- the only arithmetic operators are addition and subtraction, without interval constraints
- no vote is given to the relationships with other agents
- production instantaneously adjusts to the total amount of requests (stocks are always adequate)
- at each time step all the agents have a certain probability to create a new artifact (no more than one per agent)
- new recipes can be generated by modifying the input of an existing recipe (without changing the recipe supports) or by applying an evolutionary mechanism to the existing recipes (a genetic algorithm, which allows changes in the recipe supports)
- no new agents are generated.

4.1 Generation of new goals and new recipes

In order to introduce new products, agents need to have new goals and to develop the recipes to produce them. Two different methods for devising new goals have been tested: “imitation” and “imitation & jump”. We will show in section 5 that only the latter is able to sustain continuous innovation.

In “imitation”, an agent is allowed to choose a goal identical to one of the existing products. Actually, the goal is chosen with uniform probability among the known artifacts. In such a way a simple sampling of artifact space is performed: the higher the number of artifacts which lie in a certain region of artifact space, the higher the probability of finding that region. The underlying heuristics is that clusters of artifacts are probably high reward regions. The agents try to build an artifact similar (within a given threshold) to the selected one (which represents the “goal” of the imitation): if they succeed, the new artifact is added to the set of those that are already present.

Imitation alone is unlikely to enhance the agents’ activities, since the new artifacts have no chance to be purchased. Indeed, the name of the artifact created by imitation can be either the same of the target one (and, in this case, the agents that already utilise this name don’t have any reason to change their previous suppliers) or different from the name of the target (and, in this case, no agent is able to utilise this new kind of artifact). The only possibility for a new artifact to survive is that another agent creates a new recipe, and chooses the artifact itself to build its own product. This event has a very low probability, and in effect with only imitation the initial agents’ activities drop to zero. A way to avoid this fate is that of allowing the agents to change their suppliers in a random way. If this may happen, many agents typically survive (as shown in section 5 below).

Although in this latter case some agents survive, it is observed that the production of new artifacts ends after a finite number of time steps. The problem is indeed related to the definition of new goals: as long as agents try to imitate already existing products, they can't "imagine" new things. It is therefore appropriate to allow agents to change the way in which the goal is chosen, by

- multiplying the name of the artifact, selected by imitation, times a randomly selected value (belonging to a finite interval [min,max])
- adding or subtracting a random value.

In such a way the jump allows the exploration of new regions of artifact space and, as it will be shown in the next section, the combination of imitation and jump allows a continuous generation of new artifacts.

Once a goal has been set, an agent needs to develop the corresponding recipe. Two methods have been devised: exhaustive search and genetics.

In exhaustive search, an agent looks at each of its own recipes and, for each input, it substitutes it with another artifact. The exploration concerns all the artifacts known to the agent, although in the present version not all the combinations of known artifacts are tried: indeed, substitutions in existing recipes are done one at a time. The search halts when the goal has been reached; if this never happens, the closest product is considered and, if its distance from the initial goal is smaller than a threshold, the corresponding recipe is added to the set.

While exhaustive search deals only with the input artifacts, in genetic search also the recipe supports are combined to give rise to new recipes. In particular, the method starts by generating an initial population from a set of recipes (which can either belong to a single agent in the "lonely" mode previously described, or to two collaborating agents). In order to have an initial population, the existing recipes are augmented with some modified versions. These modifications are obtained by changing the input (as in the previous method) with some probability.

The evolution of the population proceeds according to the well known genetic algorithm. In particular, the fitness function of each individual in the population is a decreasing function of its distance from the goal. The main genetic operators are uniform crossover (which has been modified in order to take into account the existence of recipes of different lengths) and mutation (note that this term is used here with a meaning which is very different from that used in previous section). Mutation involves here changing one input artifact with the closest one among those known to the agent.

4.2 Initial conditions

Setting up initial conditions "by hand" is time demanding, so we realised an automatic "initial condition builder". First of all, raw materials are introduced; afterwards, agents are added one at a time. The recipes of the newly added agent can use the already existing artifacts, producing in such a way new artifacts. Networks of this kind are such that older artifacts are likely to be connected to more agents than new one.

The networks generated according to the above mentioned procedure have the common feature that the unsold artifacts (belonging to the last added agents) can't provide any source of activation to their owners, and therefore the latter cant' survive, causing a "domino effect" that destroys the entire net. To avoid this collapse we can either provide an external reward region or introduce a modification of the dynamics (e.g. the possibility of changing a provider) which leads to self-sustaining rings, as shown in the next section.

5 Simulation results

Several simulations have been performed, in order to test the model and to understand some if its behaviours. We do not present statistical results, but rather typical model behaviours. The simulations that are shown below are organized in an order that reflects the introduction of an increasing number of features. In particular, the sequence is the following:

- agents operate without innovation

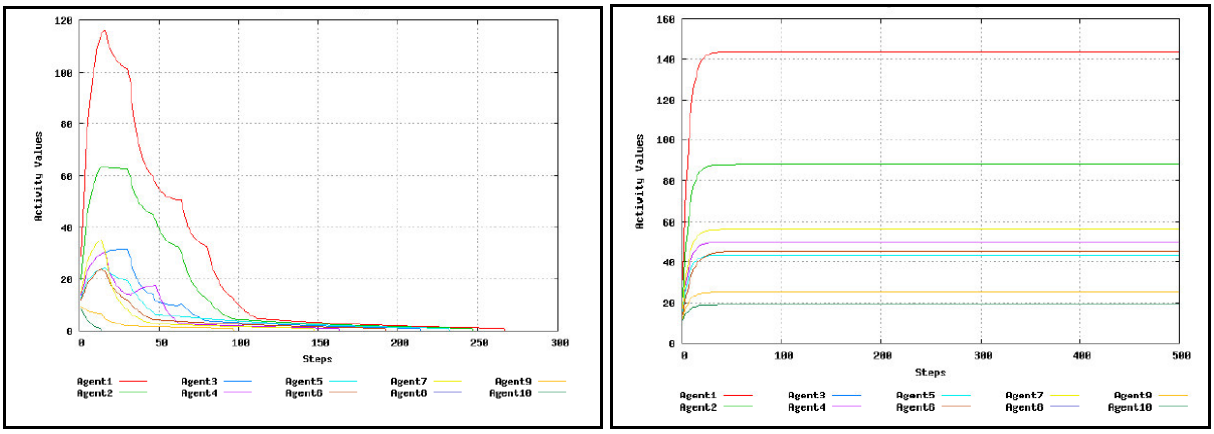
- agents perform innovations, either in suppliers or in goals and recipes, each agent knowing all the existing artifacts
- agents perform innovations, either in suppliers or in goals and recipes, each agent knowing only the artifacts produced by a limited number of other agents.

Different innovation methods have been tested. In particular, when innovation in recipes is performed by applying genetic operators to the set of existing recipes, we have considered both the case of "lonely" agents (which modify only their own recipes) and the case of "collaborative" agents, which share their recipes in genetic search of new ones.

5.1 Activation dynamics without changes

In this paragraph we show the behaviour of the model when agents do not perform any kind of innovation and never change their providers.

As already stressed, the actual initial condition builder generates networks that are never able to survive without external reward. Figure 1a shows the decrease of activities of a net with 10 agents and no self-sustaining loops or reward regions, whereas Figure 1b shows the stabilising action of a region of reward (the initial situation is the same in both simulations). During these runs the agents can't innovate and/or change providers.



(a) (b)

Figure 1: activities vs time in a net with 10 agents. (a) no self-sustaining loops or reward regions, (b) the stabilising action of reward regions.

5.2 Innovation with complete knowledge of existing artifacts

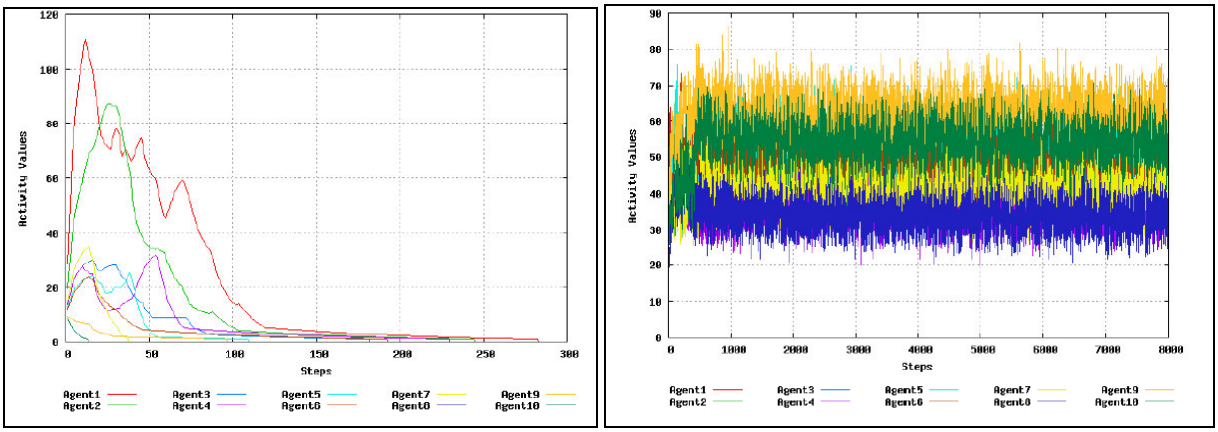
In this section we will consider agents which make use of different innovation methods (new goals, new recipes), and which can also modify their suppliers. The distinguishing feature of the different cases described here is that agents have a complete knowledge of all the artifacts produced by the other agents. In most cases we will show the behaviour of lonely agents which operate only on their own set of recipes, but we will also show the results of allowing "knowledge sharing" between pairs of agents.

5.2.1 Imitation and change of suppliers

In this section we will consider the effects of innovations (of different kinds) which agents are allowed to perform. Let us first consider imitation only. As it has been already anticipated, using imitation without change of suppliers eventually leads to the vanishing of agent activities, no matter which method is used to generate new recipes (see Figure 2a). A way to avoid this fate is that of allowing the agents to change their suppliers in a random way. If this may happen, many or all the agents typically survive (as shown in Figure 2b).

Figure 2b also shows that the system is self-maintaining. In absence of reward regions there are two possible explanations of this kind of behaviour:

1. there are self-sustaining loops;
2. the agents innovate at such a high frequency that they have always sell some artifact.



(a) (b)

Figure 2: agents' activities (no self-sustaining loops nor reward regions): (a) with imitation and without change of suppliers (the new artifacts are not purchased, and the agents die), (b) with change of suppliers the new artifacts have a chance to be purchased, and the network is able to self-maintain.

In effect, given the particular way to build the initial condition, small changes in the network structure (i.e. the change of providers) can create self-sustaining loops. It is possible to test this hypothesis, by deactivating the innovation mechanisms (imitation and change of suppliers) and therefore making the rate of innovations vanish. Figure 3 shows that after such deactivation a situation of stability is achieved, demonstrating in such a way the presence of self-sustaining loops. The removal of agent 1 at step 2000 changes the levels of stability but is not able to make the whole system collapse; further analysis could determine whether there is more than one loop or whether agent 1 is located downstream with respect to a loop.

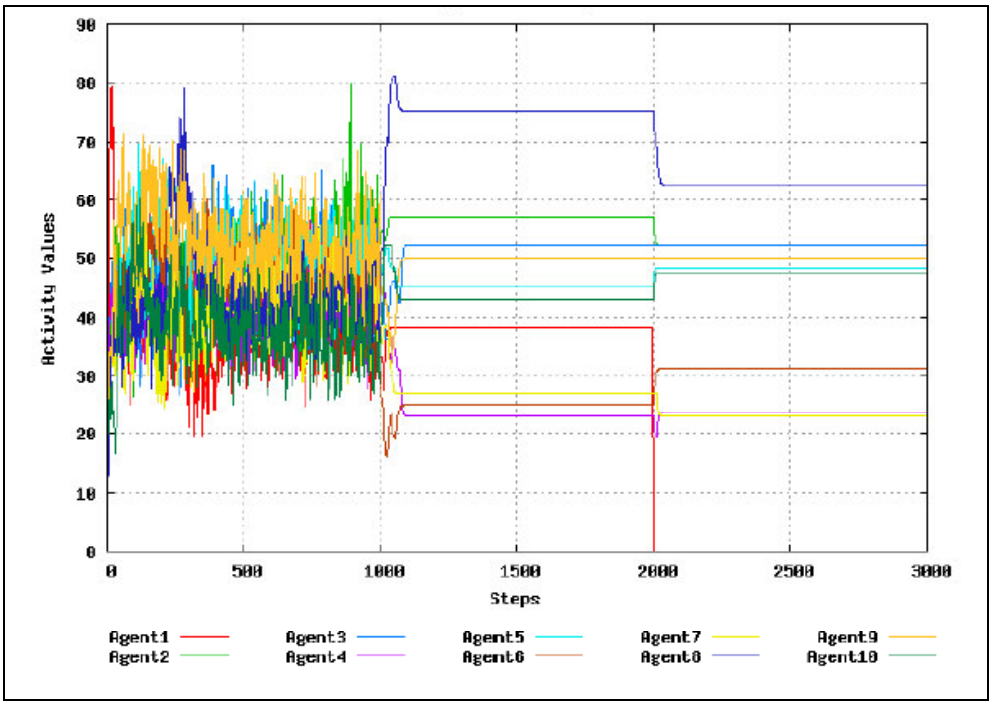
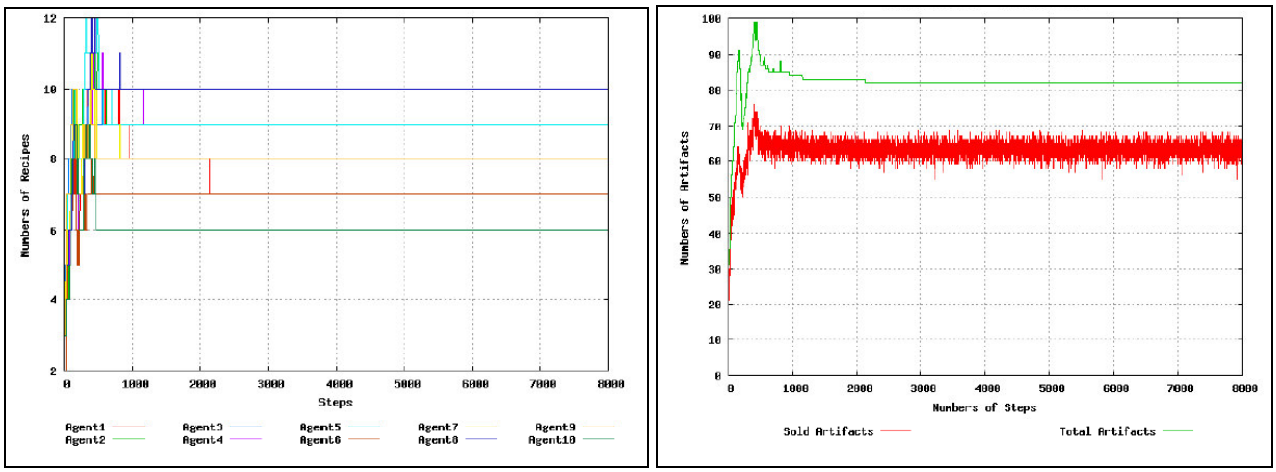


Figure 3: behaviour of the agents' activities under some external changes; step 0-999: imitation and change of suppliers are active; step 1000-3000: the rate of innovations is set equal to zero; step 2000: agent 1 is removed from the simulation.

Imitation and change of suppliers show a particular limitation, which is evident in Figure 4a: after a while, the number of recipes owned by each agent stops growing. This phenomenon indicates that imitation and

change of suppliers don't allow continuous innovation; therefore imitation is not a true "novelty generator", and in a few steps all the possible combinations are generated (Figure 4b).



(a) (b)

Figure 4: simulation with imitation and random change of suppliers: (a) number of recipes owned by each agent (b) total number of artifacts and number of sold artifacts; the distance between the two lines is due to the innovations which are not purchased by the agents.

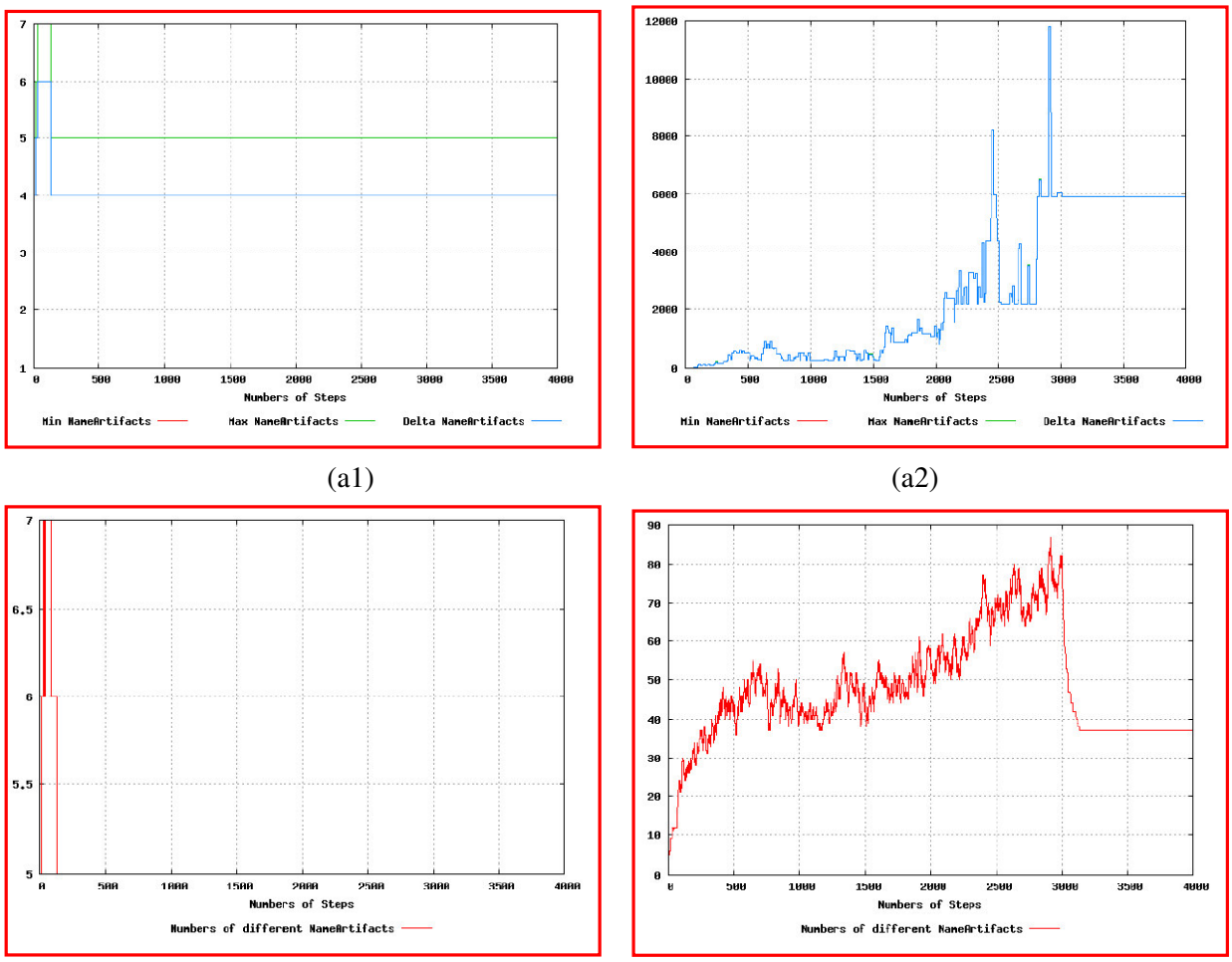
5.2.2 Jump as a continuous source of novelties

Note that the fact that innovations come to a halt, in the previous simulations, should not be attributed to a limitation in the methods used to generate new recipes but rather to the limitations in the available mechanism to devise new goals. Indeed, new recipes can be generated (see section 3) either by modifying the input of an existing recipe or by applying an evolutionary mechanism (e.g. a genetic algorithm) to the recipes owned by the agent. The simulations presented above made use of the former mechanism; however the results are similar in runs with the second mechanism turned on. The creation of new recipes is not upper limited, and all the numbers $n \in \mathbf{N}^+$ are potentially reachable, but after an initial success we saw that no new artifacts are produced.

Let us then allow agents to "modify" the chosen goal; so now an agent multiplies, with fixed probability, the name of the artifact, selected by imitation, times a randomly selected value (belonging to a finite interval [0.7, 6]). In order to appreciate the influence of this mechanism, it is convenient to compare the behaviour of two simulations, starting from the same initial conditions and differing only in the probability of performing the jump (0 in the reference case and 0.3, respectively). In both simulations the innovation process is stopped at step 3000, in order to check the existence at this time of self-sustaining loops (Figure 5).

The jump introduces very strong effects: the number of recipes owned by each agent during the second simulation is five times greater than that of the first simulation, and the number of artifacts present at step 3000 is larger by a factor of four. Even more interesting, the diameter of artifacts (the distance between the names of the most distant artifacts) and the number of different artifacts' names (the number of different kinds of artifacts present at each step of the simulation) are continuously growing, at least during the simulation time span. The network structure is continuously changing.

Note that the use of the word "jump" here refers to a change in the value of the variable "goal" which is not necessarily a small one. In effect, it has been observed (and could be theoretically justified) that the effects of jumps become sensible when their size exceeds the threshold which defines the maximum distance between a given goal and the closest product which can be actually built (i.e., when innovation is tried, a new product which can be produced using the existing recipes and inputs is actually produced only if its distance from the initial goal is smaller than the threshold). On the other hand, a too large jump coefficient leads to a basically random exploration of a too vast artifact space, which is ineffective.



(a1) (a2)

(b1) (b2)

Figure 5: comparison between the behaviours with and without the goal jump; the first column is relative to the simulation without jump, and the second column is relative to the simulation with jump. (row a): diameter of artifacts vs. time; (row b): number of different artifacts' names vs. time;

5.2.3 Response to perturbations

A powerful way to explore the global properties of a system is that of introducing a slight perturbation and following its subsequent dynamics, which can be compared to that of an unperturbed system. For example, we can consider questions like: What is the system response to perturbations? Is the system robust with respect to changes in the structure of the network? What is the typical system behaviour when one or more of its components (agents, artifacts) disappear? In our case we could:

- remove a recipe from an agent;
- remove an entire agent;
- remove a group of agents at the same moment;
- remove agents at randomly selected moments.

Different behaviours to such perturbations are possible: Figure 6 shows some examples. Agents 1, 3, 5, 7, 9 disappear respectively at step 50, 200, 500, 1000 and 1300. There are several consequences, depending upon the particular agent, the time of fault, the number of agents surviving at the moment of disappearance, the current network structure, and so on. Without describing the details, it is interesting to observe that some perturbations have important consequences (elimination of agent 5 and 9), whereas other perturbations have not. The perturbations affect mainly the number of recipes owned by each agent, the number of different kinds of artifacts and the number of artifacts present at each step of the simulation. Also the diameter of artifacts is deeply influenced. These changes modify in an irreversible way the system structure (agents,

artifacts). However, the global properties described in Figure 6 display a strong resilience, i.e. the system recovers after crashes.

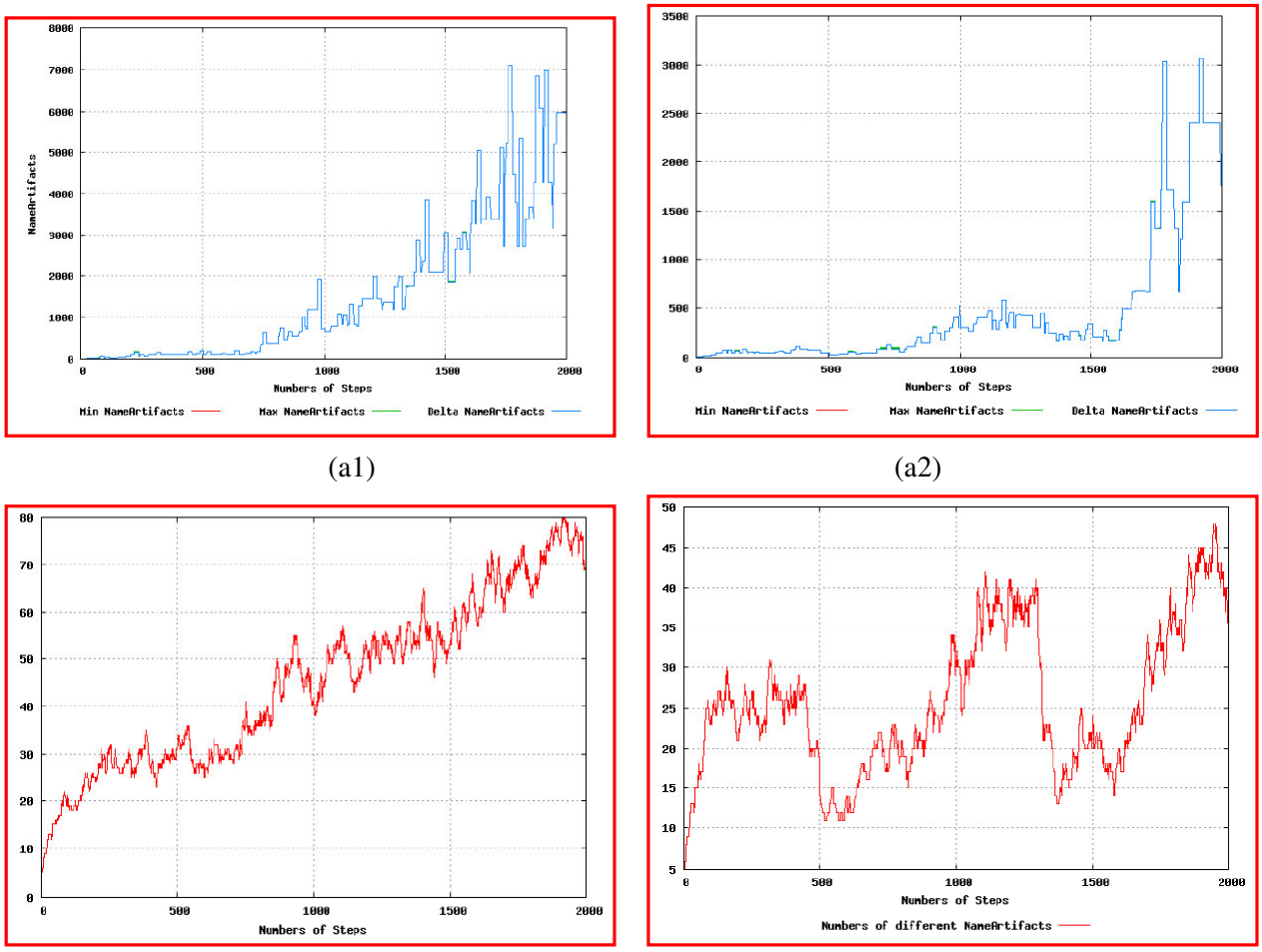


Figure 6: comparison between the behaviours with and without the externally enforced elimination of some agents (agents 1, 3, 5, 7, 9 disappear, respectively at step 50, 200, 500, 1000 and 1300). The first column is relative to the simulation without eliminations, and the second column is relative to the simulation with eliminations. (row a): diameter of artifacts vs. time; (row b): number of different artifacts' names vs. time.

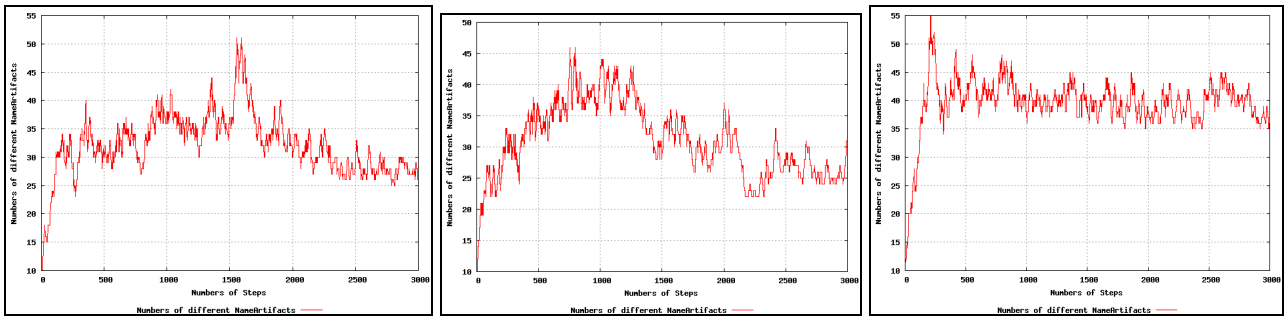
5.2.4 Using genetic search

Three different strategies have been used in these simulations

a) an agent performs genetic search using only its own set of recipes
b) an agent tries to reach its goals using only its own recipes; if it fails, it chooses another agent with which genetic is performed. If the search is successful, both partners own a copy of the successful recipe
c) agents always perform genetic search in pairs.

When joint search is performed, the chosen partner is the best supplier, i.e. the one which is selling most to the agent

Some results of the simulation are shown in Figure 7: Note that strategies a and b give very similar results, probably because, if each agent knows all the artifacts, the need for joint search is small and the two strategies behave in much the same way. An interesting observation, however, is that if joint search is prescribed, the number of different types of artifacts which are generated is higher than in the previous cases, as is also the number of different recipes per agent. Moreover, using the first two methods one observes a decline in the number of different artifacts, which instead seems to remain fairly constant in the third case.



(a) (b) (c)

Figure 7: number of different artifact types; (a), (b) (c) are described in the text

5.3 Innovation with limited knowledge of existing artifacts

While complete knowledge of the "artifacts" is probably useful for an agent, it represents an unrealistic assumption in many real world situations. It is therefore very interesting to explore the behaviour of a system whose agents have incomplete knowledge of the products of other agents. In particular, aspects related to the interaction between agents are expected to be better described in such a framework.

In this case, each agent knows the artifacts produced by some other agents (i.e. its own "network"). The properties of this network should play a major role in the system behaviour, as we will show in the following sections.

We will consider first the case where an agent knows only its suppliers, and we will then analyze the case where the network of acquaintances is enlarged.

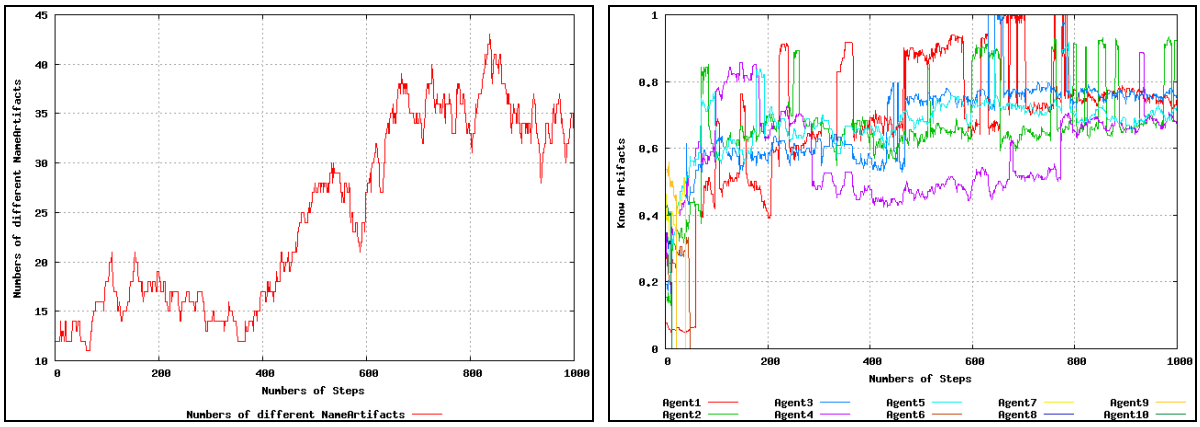
5.3.1 Knowing only one's suppliers

We start with the simplest case where an agent knows only the products of its current suppliers; the agent tries to innovate by

- defining its goal through imitation & jump (where imitation is performed among the known artifacts)
- finding new recipes by genetics, according to strategy b (i.e. it first tries alone, if it fails it cooperates with its best supplier).

It is interesting to note that, if we had used strategy a (i.e. lonely genetics), the system would have died out, while with strategy b it survives. So in this case the two strategies lead to markedly different results, which is likely to be due mainly to the enlargement of an agent's network which is possible only in the latter case.

Note also that the diversity (i.e. number of different artifact types) grows in time and that the network of each agent grows, without however reaching complete knowledge. This is shown in Figure 8b where the fraction of artifacts known to an agent (i.e. the ratio known artifacts / total artifacts) is plotted vs. time. It is particularly interesting to observe that, due to the continuous change which is taking place, even an agent which, at a certain time, knows all the artifacts, is bound to loose this property at subsequent time steps.



(a) (b)

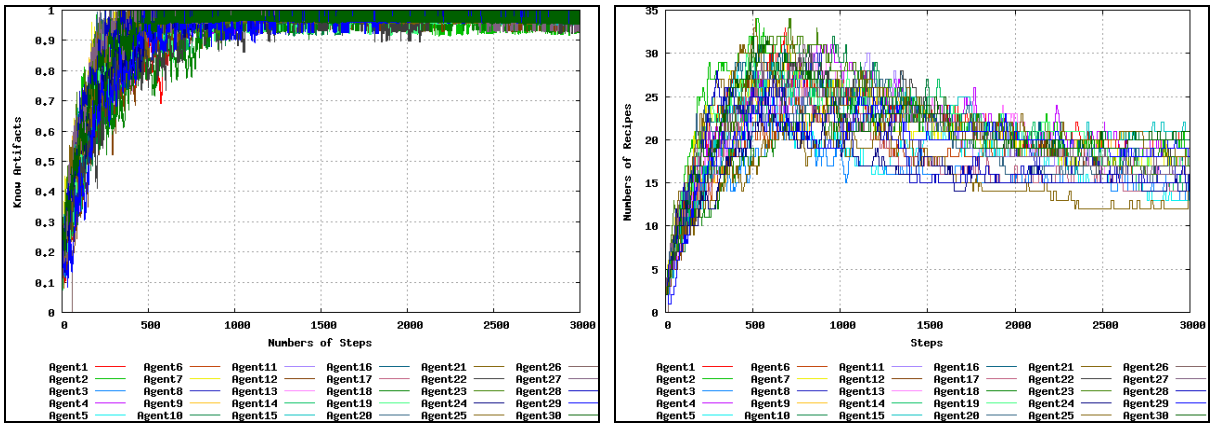
Figure 8: genetic strategy b: knowledge of only one's suppliers: (a) number of different artifacts' names vs. time; (b) fraction of artifacts known to each agent vs. time

5.3.2 Knowing not only an agent's suppliers

A particularly important aspect is the social network of each agent, and the way in which it changes in time. While in section 2 a sophisticated way to handle this aspect has been described, based upon an evaluation of each potential partner, here we limit to a simple way to "go beyond" the knowledge of an agent's suppliers, which has been described above. We will consider the case where an agent knows all the products of its current suppliers and all the products of one or more other agents chosen at random, at each time step, among the set of existing agents (which will be referred to below as "acquaintances").

In order to disentangle this effect from that of enlarging the set of suppliers, due to the combination of recipes described in the previous section, we limit here genetics to the recipes of a single agent. Therefore the agent here is lonely as far as the generation of new recipes is concerned, but it has a wider social network in that it knows the products of some agents which are not among its suppliers.

We have tested three cases which differ in the number of acquaintances per time step; the system is self sustaining and the number of different artifact types grows and then slowly declines in all cases. The most interesting observation is that each agent soon comes to know all the artifacts, and this situation persists in time, even if at each time step a single new acquaintance is considered (Figure 9). This particular behaviour, which may appear surprising, is due to the fact that, in all the simulations above, there is no hard limit on the number of recipes an agent can simultaneously possess. The only mechanisms for eliminating a recipe are indeed i) lack of inputs or ii) lack of sales. What is observed in the previous simulations is that each agent has a large number of recipes, and in such a way the network of suppliers covers the whole set of agents.



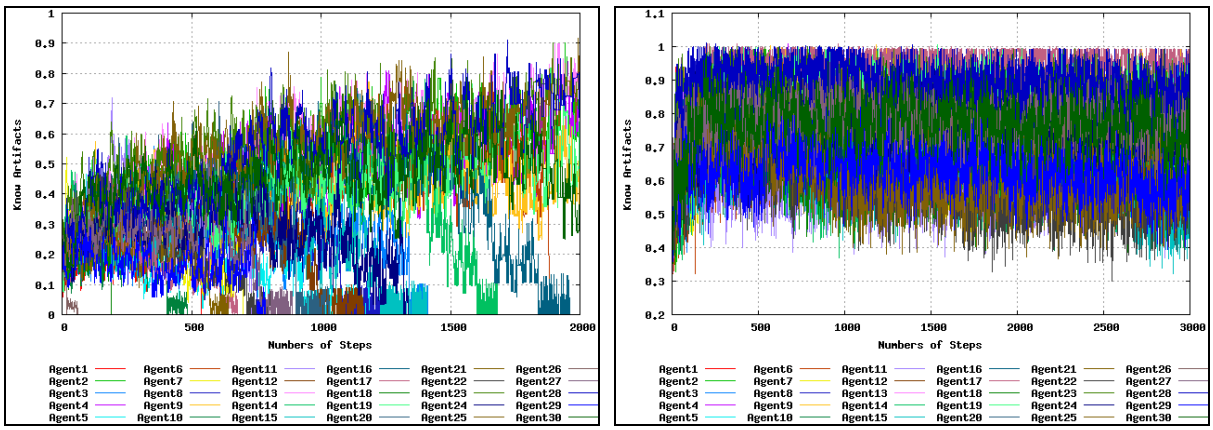
(a) (b)

Figure 9: genetic strategy a with acquaintances: (a) fraction of artifacts known to each agent vs. time; (b) number of recipes owned by each agent vs. time

In order to further investigate this aspect, we therefore introduced a novel mechanism for eliminating recipes: at each time step, there is a small but finite probability of eliminating the less performing recipe, defined as the one which has the lowest volume of sales in the last 15 steps.

Some very interesting phenomena have been observed in this case. The number of agents which are known to each agent increases with the number of acquaintances per time step N_a, as expected, but even in the case of a very high N_a (e.g. 10 new agents in a population of 30) a distribution of the number of known agents is observed: while there is an agent which knows all the other agents and therefore all the artifacts, there are some other surviving agents which know no more than 40% of the agents.

Similar results are obtained for the number of artifacts that are known to the different agents. In Figure 10 relative data are plotted: note that, although in the small N_a case the total number of artifatcts and agents declines, the relative number (known artifacts divided by total number of artifacts) actually increases, thus allowing the system long term survival. Indeed in this case the total number of agents declines, while with high N_a this number remains constant.

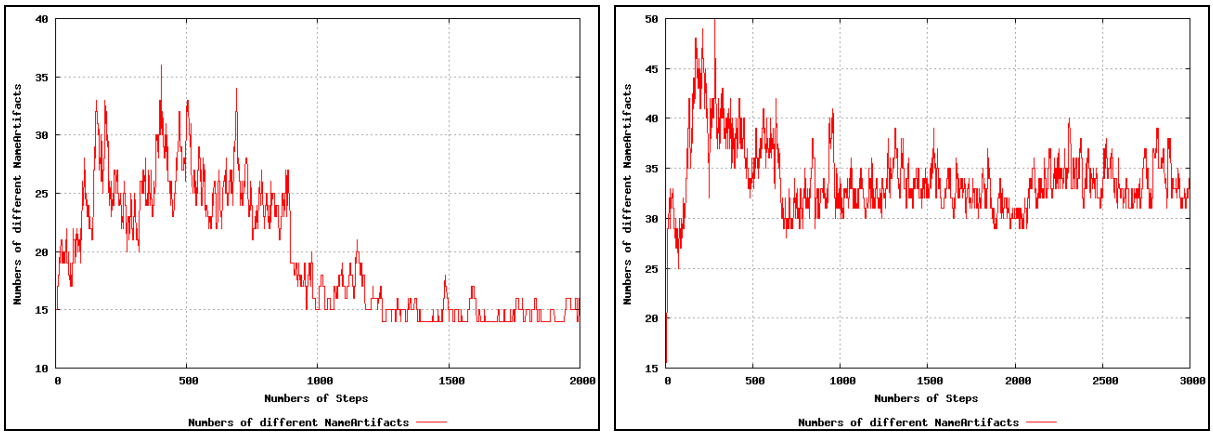


(a) (b)

Figure 10: genetic strategy a with acquaintances: f**raction of artifacts known to each agent**: **(a) one acquaintance at each step; (b) ten acquaintances at each step**

It is interesting to consider the number of recipes per agent that, as it may be expected, is a growing function of the number of acquaintances. Even in this case a wide distribution of the number of recipes per agent is observed: in the case N_a=1, the final (i.e. at t=2000) number ranges between 4 and 8 recipes, while if N_a=15 this number is comprised between 3 and 15. The analysis of the distribution of recipes (as well as the distribution of known agents and artifacts) is one of the most interesting future directions of research that can be pursued with this model.

Finally, let us consider the number of different artifact types, which is a growing function of N_a (Figure 11). Note that in the graph a maximum number of different artifacts is reached, which is followed by relaxation to a lower number which persists for a long time. This is an unexpected feature, which is found in many different simulations and which needs further investigation.



(a) (b)

Figure 11: genetic strategy a with acquaintances: number of different artifact types: (a) one acquaintance at each step; (b) ten acquaintances at each step

6 Conclusions

The model under consideration belongs to the class of agent-based models, which are well suited to bridge the gap between a careful description of the behaviour of single agents (including their interactions with another or a few other agents) and the system-level features (which come out of the interaction of many such agents).

The relationship with the innovation theory of Lane, Maxfield and co-workers requires that intentionality plays a key role. And indeed in I_2M the agents' intentions are explicitly represented as goals, i.e. products which the agents want to produce and, for this purpose, they try to develop new recipes. This is a major distinguishing feature with respect to many agent-based models which rely upon chance as a generator of novelty.

Another distinguishing feature is the role of the agents' social interactions (both of the customer-supplier and of the "acquaintance" type). Although only preliminary simulations have been performed concerning this aspect, it is apparent from the results of section 5 that the outcomes are deeply affected by the parameters which describe how this social network evolves.

In the present version of the prototype a very simplified way of modifying the social network has been tested, based upon chance, while one of the major future directions of research would involve intentionality also in the change of the social interactions.

It is also interesting to observe that the tests performed so far have shown that the network of customers and suppliers is actually able to maintain itself, often forming closed loops, and that it is able to evolve giving rise to "perpetual" novelties. Therefore the modelling framework which has been envisaged seems well suited for the exploration of the main goals of innovation modelling in Iscom.

A major line for further development concerns a treatment of agents social interactions more closely related to the LM theory. In particular, in a system like ours it will be possible to measure the generative potential of a binary relationship, and to compare the behaviour of agents with different strategies. Some agents will follow the heuristics of favouring relationships with high generative potential (as suggested by Lane and Maxfield), while others will use different heuristics like e.g. "imitate the most successful agents you know" or "cooperate with your best customer". In this way it will be possible to explore under which conditions one heuristics outperforms the others.

The measure of the generative potential may be the product of three normalized quantities, each of which can be determined in our model

- mutual directedness, which can be measured by the vote given to a relationship

- heterogeneity of the agents, which can be related to the differences between their sets of recipes
- aligned directedness, which is measured by the distance between the agents' goals

Moreover, in the future it will be useful to develop suitable tools in order to analyse directly the structure (i.e. characteristic path length, clustering coefficient, topology, etc) of the network which develops, connecting different agents and artifacts.
It will also be interesting to consider possible ways to simplify the model, at least in particular conditions. Indeed, in spite of our efforts to keep things as simple as possible, the intimate relationship with the LM theory led us to a rather complicated model. Although we do not subscribe the claim that a model with several parameters can describe almost everything [12, 13], we feel that considering slightly simplified versions of the model might provide useful insights about its behaviour.

Finally, we stress that further improvements are under way on the code in order to make it faster, thus allowing simulations which will involve very many agents, which will allow a better exploration of the emergent properties.

Acknowledgements

We wish to express our gratitude to Andrea Ginzburg, who discussed at length with us the features of this model, since the time of its first conception. We are also indebted to Margherita Russo and to many members of the group working on the Iscom project for many helpful discussions and suggestions. This work has been supported by UE-FET initiative under grant IST-2001-35505 Iscom: The Information Society as a Complex System.

References

[1] Epstein JM, Axtell R. : Growing Artificial Societies: Social Science from the Bottom Up. Cambridge (MA), MIT Press, 1996
[2] Gilbert N, Terna P: How to build and use agent-based models in social science, Mind & Society 2000; 1:57-72
[3] Cioffi-Revilla C**.:** Invariance and universality in social agent-based simulations, PNAS 2002; 99: 7314-7316
[4] Ormerod P, Rosewall B.: On the methodology of assessing agent-based evolutionary models in the social sciences, Second Brisbane workshop on complex adaptive systems 2002; www.paulormerod.com/brisbane3PO
[5] Gilbert, N : Agent-based social simulation: Dealing with complexity; 2005: http://www.complexityscience.org/NoE/ABSS-dealing%20with%20complexity-1-1.pdf
[6] Axelrod R, Tesfatsion, L.: A guide for newcomers to agent-based modeling in the social sciences. In Kenneth L. Judd, KL, Tesfatsion, L (editors): Handbook of Computational Economics, Vol.2: Agent-Based Computational Economics. Amsterdam, North-Holland, 2005 (forthcoming)
[7] Maxfield R, Lane D : Foresight, complexity and strategy. In Arthur WB, Durlauf S, Lane D (eds): Economy as an evolving complex system II (chap. 4). Redwood city (CA), Addison Wesley, 1997
[8] Lane D, Maxfield R. : Ontological uncertainty and innovation, Journal of Evolutionary Economics 2005, 15: 3-50
[9] Lane D: Articial worlds and economics (I and II), Journal of Evolutionary Economics 1993, 3: 88-107, 177-197
[10] Fontana W: Algorithmic chemistry, in Langton CG et al (eds): Artificial Life II. Redwood City (CA), Addison Wesley, 1992
[11] FontanaW, Buss LW: The arrival of the fittest: towards a theory of biological organization, Bull. Mathem. Biol. 1994, 56: 1-64
[12] Di Gregorio S, Serra R, Villani M.: Applying cellular automata to complex environmental problems: the simulation of the bioremediation of contaminated soils. Theoretical computer science 1999, 217**:**131-156
[13] Serra R, Villani, M, Colacci, A. (2001): Differential equations and cellular automata models of the growth of cell cultures and transformation foci. Complex Systems 2001, 13 : 347-380