DATA MINING APPLIED TO AGENT BASED SIMULATION

Marco Remondino
Department of Computer Science
University of Turin
C.so Svizzera 185
10149 Turin, Italy
E-mail: remond@di.unito.it

Gianluca Correndo
Department of Computer Science
University of Turin
C.so Svizzera 185
10149 Turin, Italy
E-mail: correndo@di.unito.it

KEYWORDS

Data Mining, Agent Based Simulation, validation, emergence, artificial intelligence

ABSTRACT

Agent Based Modeling is the most interesting and advanced approach for simulating a complex system: in a social context, the single parts and the whole are often very hard to describe in detail. Besides, there are agent based formalisms which allow to study the emergency of social behavior with the creation and study of models, known as artificial societies. Thanks to the ever increasing computational power, it's been possible to use such models to create software, based on intelligent agents, which aggregate behavior is complex and difficult to predict, and can be used in open and distributed systems.
Data mining is born in the last decades in order to help users in finding useful knowledge from the otherwise overwhelming amount of data available nowadays from the web and the data collected every day by companies.
Data Mining techniques can therefore be the keystone to reveal non-trivial knowledge expressed by the initial assumption used to build the micro-level of the model and the structure of the society of agents that emerged from the simulation.

INTRODUCTION

Nowadays the simulation is one of the best paradigms for modeling the behavior of complex systems even thought it has some leaks. Above all, the simulation model is only a rough approximation of the real system to study; each approximation produced will not cover the whole set of details we can actually study looking at the real system. The gap between model and reality is well known in fields like Computer Science and Mathematics but the situation is far from being desperate. The gap can be intentional since the realm of interest can be a small piece of the whole sensible world. Moreover the ability to create artificial worlds whose relations and states can be arbitrarily changed allows us to explore the full possibility of the real system. The fact to simulate a system under unnatural conditions can help us to study scenarios of unimaginable flexibility.

The other side of the coin is that the procedure of modeling introduces a bias that it is difficult to detect. How reliable is a model? How to proceed in the model proposal? These are difficult questions to answer when there are no strong and formal fundamentals in model proposal.
Statistical techniques usually try to overcome such bias using distribution hypothesis and strong mathematical foundations for the procedures used during data analysis.
In the present paper the authors try to propose a cross fertilizations between Agent Based Simulation and Statistical Learning techniques (or more specifically Data Mining techniques) in order to handle and possibly overcome the limitations of both.

AN INTRODUCTION TO DATA MINING

Data Mining is the key element of the Knowledge Discovery in Databases (KDD) task. KDD is defined as "*the process of identifying valid, novel, potentially useful and ultimately understandable patterns in data*". We could finally add that such task involves usually great amount of data, usually stored in analysis oriented data stores called Data Marts.
Data Mining is not a field in itself; it is more a collection of methods of data analysis coming from different fields of computer science, artificial intelligence and statistics. Just statistics supplies mathematical concreteness to many of the date mining methods.
Data mining was born in the latest decades in order to help users in finding useful knowledge from the otherwise overwhelming amount of data available nowadays, both coming from the web and the data collected every day by companies.
The kind of knowledge the users can extract from the raw data is heterogeneous and most depends on the nature of the data available. In fact the nature of the data and the kind of task guide the process of data analysis itself, that is more the production of an artigianal process guided by the experiences rather than the result of an automatic process.
The types of tasks Data Mining could accomplish can be roughly divided in two categories: predictive tasks and descriptive tasks. The first type of tasks try to discover a model that drives the behavior of some variables in a system in order to be able to predict such values in zones not covered by the examples. The

Proceedings 19th European Conference on Modelling and Simulation
Yuri Merkuryev, Richard Zobel, Eugène Kerckhoffs © ECMS, 2005
ISBN 1-84233-112-4 (Set) / ISBN 1-84233-113-2 (CD)

second type of task tries to find some categorizations of the data producing a shrinked descriptor for wider segments of data.

Predictive Data Mining

One of the predictive tasks of Data Mining is the task of finding some form of classifications of the items contained in the data mart from a set of raw data. When there is a finite set of classes that describe the domain of the data, the classification can be carried on by some *if-then* rules that help users to classify a new item in one of such predefined classes. Such classification process is based on the values of some characteristics of the item itself and can be deterministic (e.g. there is no doubt about the belonging of the item to the given class) or heuristic (e.g. the association of the item to one or more classes is given with a degree of certainty).
The association model so far extracted can have the form of a decision tree, instead of a set of if-then rules, but the purposes of the model retrieved remains the same. When the classification domain is not finite (e.g. when the variable interested by the prediction process is a real number) the operation is called *regression*. The regression task helps the user to model an analytic function that describes the set of data submitted to the task and that can predict new, not submitted, values.

Descriptive Data Mining

In descriptive Data Mining the task is to discover interesting regularities in the data, to uncover patterns and find interesting subgroups in the bulk of data.
Such kind of Data Mining produce a categorization of the initial amount of data uncovering patterns that were not evident before the execution of the task. Expert of the domain must then interpret the patterns so far uncovered in order to explain them.
A typical product of this kind of task is the discovery of association rules that find untilled relationships between features' values looking at the examples proposed as training.
Such association rules can be used as classifiers to find some subgroups dividing the population in relevant clusters. The division in clusters reflects some important division present in the data that could be crucial in order to reason using a small number of stereotypes instead of a huge number of single items.
Another important task associated to Data Mining is the use of advanced techniques of visualization. In fact, since data analysts and domain specialists do most of the work of discovery, it is very important to find good visual metaphors to give users right intuitions to guide the analysis.
Naturally such metaphors are only useful to guide the intuition, in order to provide mathematical soundness the Data Mining is supported by statistical methods such as probabilities laws for the items values' prediction, Bayesian theorems for defining some sort of causality and so on. The techniques of Data Mining, having their foundations in statistic, require a large number of items to build satisfying results. When only a small number of examples are available, techniques of Machine Learning, coming from AI and inductive logic fields, are suggested. Such techniques find their fundaments in symbolic reasoning and non-classical logics and do not require statistical tools for soundness checking.

AGENTS FOR SOCIAL SIMULATION

The concept of software agent originates in the early fifties with J. McCarthy, while the term has been coined by O.G. Selfridge some years later, when both of them were working at the Massachusetts Institute of Technology. Their original project was to build a system which, given a goal, could be able to accomplish it, looking for human help in case of lack of necessary information. In practice, an agent was considered a software robot that lives and acts in a virtual world. In (Wooldridge and Jennings 1995): "... a hardware or (more usually) software-based computer system that enjoys the following properties:

- *autonomy*: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;
- *social ability*: agents interact with other agents (and possibly humans) via some kind of agent-communication language;
- *reactivity*: agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;
- *pro-activeness*: agents do not simply act in response to their environment, they are able to exhibit goal-directed behaviour by taking the initiative." The Wooldridge and Jennings definition, in addition to spelling out autonomy, sensing and acting, allows for a broad, but finite, range of environments. They further add a communications requirement.

Franklin and Graesser (1997) also try to find the typical features of agency, deriving them from the word itself: an "agent" is 1) one who acts, or who can act, and 2) one who acts in place of another with his permission. Since "one who acts in place of " acts, the second usage requires the first. Humans act, as do most other animals. Also, some autonomous mobile robots act, for example Brooks' Herbert (Brooks 1990; Franklin 1995). All of these are real world agents. Software agents "live" in computer operating systems, databases, networks, MUDs, etc.
Finally, artificial life agents "live" in artificial environments on a computer screen or in its memory (Langton 1989, Franklin 1995).
Each is situated in, and is a part on some environment. Each senses its environment and act autonomously upon it. No other entity is required to feed it input, or to interpret and use its output. Each acts in pursuit of it's own agenda, whether satisfying evolved drives as in humans and animals, or pursuing goals designed in by

some other agent, as in software agents. (Artificial life agents may be of either variety.) Each acts so that its current actions may effect its later sensing, that is its actions effect its environment. Finally, each acts continually over some period of time. A software agent, once invoked, typically runs until it decides not to. An artificial life agent often runs until it's eaten or otherwise dies. Of course, some human can pull the plug, but not always. Mobile agents on the Internet may be beyond calling back by the user.
These requirements constitute for sure the essence of being an agent, hence the definition by Franklin and Graesser (1997):

An autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future.

And the very general, yet comprehensive one by Jennings (1996):

…the term is usually applied to describe self-contained programs which can control their own actions based on their perceptions of their operating environment.

Agents themselves have traditionally been categorized into one of the following types (Woolridge and Jennings, 1995):

- Reactive
- Collaborative/Deliberative
- Hybrid

When designing any agent-based system, it is important to determine how sophisticated the agents' reasoning will be. Reactive agents simply retrieve pre-set behaviors similar to reflexes without maintaining any internal state. On the other hand, deliberative agents behave more like they are thinking, by searching through a space of behaviors, maintaining internal state, and predicting the effects of actions. Although the line between reactive and deliberative agents can be somewhat blurry, an agent with no internal state is certainly reactive, and one that bases its actions on the predicted actions of other agents is deliberative.
In Mataric (1995) we read that reactive agents maintain no internal model of how to predict future states of the world. They choose actions by using the current world state as an index into a table of actions, where the indexing function's purpose is to map known situations to appropriate actions. These types of agents are sufficient for limited environments where every possible situation can be mapped to an action or set of actions.
The purely reactive agent's major drawback is its lack of adaptability. This type of agent cannot generate an appropriate plan if the current world state was not considered a priori. In domains that cannot be completely mapped, using reactive agents can be too restrictive.

Different from reactive agents are the deliberative ones. The key component of a deliberative agent is a central reasoning system (Ginsberg, 1989) that constitutes the intelligence of the agent. Deliberative agents generate plans to accomplish their goals. A world model may be used in a deliberative agent, increasing the agent's ability to generate a plan that is successful in achieving its goals even in unforeseen situations. This ability to adapt is desirable in a dynamic environment.
The main problem with a purely deliberative agent when dealing with real-time systems is reaction time. For simple, well known situations, reasoning may not be required at all. In some real-time domains, such as robotic soccer, minimizing the latency between changes in world state and reactions is important.
Hybrid agents, when designed correctly, use both approaches to get the best properties of each (Bensaid and Mathieu, 1997). Specifically, hybrid agents aim to have the quick response time of reactive agents for well known situations, yet also have the ability to generate new plans for unforeseen situations.

Multi Agent Systems (MAS)

A multi agent system can be thought of as a group of interacting agents working together to achieve a set of goals. To maximize the efficiency of the system, each agent must be able to reason about other agents' actions in addition to its own. A dynamic and unpredictable environment creates a need for an agent to employ flexible strategies. The more flexible the strategies however, the more difficult it becomes to predict what the other agents are going to do. For this reason, coordination mechanisms have been developed to help the agents interact when performing complex actions requiring teamwork. These mechanisms must ensure that the plans of individual agents do not conflict, while guiding the agents in pursuit of the goals of the system.

AGENT BASED SIMULATION

The most diffused simulation paradigms are: Discrete Event (DE) Simulation, System Dynamics (SD) and Agent Based (AB) Simulation.
The term DE simulation applies to the modeling approach based on the concepts of entities, resources and block charts describing entity flow and resource sharing. DE simulation is usually applied to process modeling, hence the definition of "process simulation", which is a sub-set of the DE one.
According to Jay W. Forrester in the 1950s, SD is "*the study of information-feedback characteristics of industrial activity to show how organizational structure, amplification (in policies), and time delays (in decisions and actions) interact to influence the success of the enterprise*". SD heavily relies upon systems of differential equations, which best represents the feedback loops typical of this approach.
In (Ostrom 1988), agent based simulation is described as a third way to represent social models, being a

powerful alternative to other two symbol systems: the verbal argumentation and the mathematical one. The former, which uses natural language, is a non computable way of modelling though a highly descriptive one; in the latter, while everything can be done with equations, the complexity of differential systems rises exponentially as the complexity of behaviour grows, so that describing complex individual behaviour with equations often becomes an intractable task. Simulation has some advantages over the other two: it can easily be run on a computer, through a program or a particular tool; besides it has a highly descriptive power, since it is usually built using a high level computer language, and, with few efforts, can even represent non-linear relationships, which are tough problems for the mathematical approach. According to (Gilbert, Terna 2000):

"*The logic of developing models using computer simulation is not very different from the logic used for the more familiar statistical models. In either case, there is some phenomenon that the researchers want to understand better, that is the target, and so a model is built, through a theoretically motivated process of abstraction. The model can be a set of mathematical equations, a statistical equation, such as a regression equation, or a computer program. The behaviour of the model is then observed, and compared with observations of the real world; this is used as evidence in favour of the validity of the model or its rejection*"

In Remondino (2003) we read that computer programs can be used to model either quantitative theories or qualitative ones; simulation has been successfully applied to many fields, and in particular to social sciences, where it allows to verify theories and create virtual societies. In order to simulate the described problem, multi-agent technique is used. Agent Based Modelling is the most interesting and advanced approach for simulating a complex system: in a social context, the single parts and the whole are often very hard to describe in detail. Besides, there are agent based formalisms which allow to study the emergency of social behaviour with the creation and study of models, known as artificial societies. Thanks to the ever increasing computational power, it's been possible to use such models to create software, based on intelligent agents, which aggregate behaviour is complex and difficult to predict, and can be used in open and distributed systems. The concept of Multi Agent System for social simulations is thus introduced: the single agents have a very simple structure. Only few details and actions are described for the entities: the behaviour of the whole system is a consequence of those of the single agents, but it's not necessarily the sum of them. This can bring to unpredictable results, when the simulated system is studied.
In an AB model, there is not a place where the global system behavior (dynamics) would be defined. Instead, the modeler defines behavior at individual level, and the global behavior emerges as a result of many (tens, hundreds, thousands, millions) individuals, each following its own behavior rules, living together in some environment and communicating with each other and with the environment. That is why AB modeling is also called bottom-up modeling.
The agent-based view takes a different approach to modeling. Instead of creating a simple mathematical model, the underlying model is based on a system comprised of various interacting agents. Therefore, its structure and behavior have potential to resemble the actual economic theory and reality better than simple mathematical models. Especially, when the underlying real relationships are complex.

In (Bonabeau, 2002), we read that AB paradigm can be used successfully to model different situations, like flows, markets, organizations, social diffusion of phenomena.

DATA MINING IN AGENT BASED SIMULATION TASKS

While in the process simulation the focus is on the function description of the single parts that are modeled in detail, in agent based simulation the most important facet is the interaction among entities. In fact it is such interaction that produce a variety of behavior that was not explicitly described in the model of the single parts. In agent based simulation there are therefore two main levels that use distinct languages with distinct purposes. A micro-level used to describe a simple local behavior and a macro-level whose effects derive in part from the micro-level and in part from the interaction of more elements. Such emergent behaviors could be revealed by non-explicit patterns in the simulation data and a following phase to the simulation can be needed in order to reveal the model that subtend the data production. Data Mining techniques can therefore be the keystone to reveal non-trivial knowledge expressed by the initial assumption used to build the micro-level of the model and the structure of the society of agents that emerged from the simulation.
Data Mining, and Machine Learning in general can be used in a number of ways in agent-based simulation, we can classify these contributions in two main tasks:

- *Endogenous modeling*. Where Machine Learning and Data Mining techniques can be used to provide the single agent a sort of intelligent behavior that analyze the data of past executions of the simulation learning from experience and tuning some initial parameters of the simulation in order to reach some local maximum (Remondino, 2003).
- *Exogenous modeling*. Where the final results of a simulation are analyzed using Data Mining techniques in order to reveal interesting patterns in data that could help to better model the behavior of the overall systems. Note that the system's behavior is usually more that the sum of the parts and it is not described in the first phase of the simulation task. Data Mining could be used to build a model supported by statistical evidence that

could validate or refuse some initial hypothesis on the system.

Endogenous Modeling

A lot of models used in agent-based simulation tries to capture the emergent unpredictable behavior of rational agents when they interact with a population of peers. The machine learning algorithms allows an agent to learn from its past history in a human similar way, that is to say, by induction. we can choose to create agents with the ability to compute rules and strategies, and evolve according to the environment in which they act; in order to model them, we can use some methods derived from the studies on artificial intelligence, such as artificial neural networks and evolutionary algorithms. While the former is a collection of mathematical functions, trying to emulate nervous systems in the human brain in order to create learning through experience, the latter derives from observations of biological evolution. Genetic Algorithms derive directly from Darwin's theory of evolution, often explained as "survival of the fittest": individuals are modelled as strings of binary digits and are the encode for the solution to some problem. The first generation of individuals is often created randomly, and then some fitness rules are given (i.e. better solutions for a particular problem), in order to select the fittest entities. The selected ones will survive, while the others will be killed; during the next step, a crossover between some of the fittest entities occurs, thus creating new individuals, directly derived from the best ones of the previous generation. Again, the fitness check is operated, thus selecting the ones that give better solutions to the given problem, and so on. In order to insert a random variable in the genetic paradigm, that's something crucial in the real world, a probability of mutation is given; this means that from one generation to the next one, one or more bits of some strings can change randomly. This creates totally new individuals, thus not leaving us only with the direct derivatives of the very first generation. Genetic Algorithms have proven to be effective problem solvers, especially for multi-parameter function optimization, when a near optimum result is enough and the real optimum is not needed. This suggests that this kind of methodology is particularly suitable for problems which are too complex, dynamic or noisy to be treated with the analytical approach; on the contrary, it's not advisable to use Genetic Algorithms when the result to be found is the exact optimum of a function. The risk would be a convergence to some results due to the similarity of most the individuals, that would produce new ones that are identical to the older ones; this can be avoided with a proper mutation, that introduces in the entities something new, not directly derived from the crossover and fitness process. In this way, the convergence should mean that in the part of the solution space we are exploring there are no better strategies than the found one. It's crucial to choose the basic parameters, such as crossover rate and mutation probability, in order to achieve and keep track of optimal results and, at the same time, explore a wide range of possible solutions.
Classifier Systems derive directly from Genetic Algorithms, in the sense that they use strings of characters to encode rules for conditions and consequent actions to be performed. The system has a collection of agents, called classifiers, that through training evolve to work together and solve difficult, open-ended problems. They were introduced in (Holland 1976) and successfully applied, with some variations from the initial specifics, to many different situations. The goal is to map if-then rules to binary strings, and then use techniques derived from the studies about Genetic Algorithms to evolve them. Depending on the results obtained by performing the action corresponding to a given rule, this receives a reward that can increase its fitness. In this way, the rules which are not applicable to the context or not useful (i.e. produce bad results) tend to loose fitness and are eventually discarded, while the good ones live and merge, producing new sets of rules. In (Kim, 1993) we find the concept of Organizational-learning oriented Classifier System, extended to multi-agent environments with introducing the concepts of organizational learning. According to (Takadama 1999), in such environments agents should cooperatively learn each other and solve a given problem. The system solves a given problem with multi-agents' organizational learning, where the problem cannot be solved simply by the sum of individual learning of each agent.

Exogenous Modeling

In particular, the exogenous modeling can be an important task in agent-based simulation since it provides safe techniques to analyze the results of this kinds of simulation paradigm. In fact, one of the most debated issues in agent based simulation community is the absence of a safe technique for validate the results of the simulations. This kind of statistical analysis of the results of the simulation could provide a real added value to this kind of representation of social models. In fact, in modeling social systems, the first step is to create a metaphor of the real system. Such models of the reality suffer, as we said in the introduction, of some initial hypothesis that must test when the first results came up. The usual validation is based upon the matching of the simulation values; if the model predicts, to some extent, the values observed in reality then this is taken as a proof of validity of the model itself (Gilbert, Terna 1999). The goodness criterions follow usually statistical theories and make reference to the knowledge of hypothesis testing, where a distribution of values is compared to a reference distribution in order to come up with a fitness number.
Using Data Mining we can use statistical foundations in order to deduce from the values of the simulation a model that well describe such values. Such models provided by statistical analysis are relative to the whole system; they try to describe, with simple and

deterministic models, how the single entities cooperate in order to produce the observed behavior.
There are many Data Mining tools that can be used in order to help the analysts to extract valuable knowledge about the reality whose drives the modeling phase or about the model itself. In the following we will provide a short overview of those whose are more interesting in our point of view, but the discussion is far from being closed. This is just a hint in order to stimulate the discussion.

Analysis of variance

The analysis of variance is one method used in statistical analysis to discover unsaid relationships between variables of a system. In few words, variables are related if the distribution of their values systematically corresponds. For example, in a population, the height is related to weight because typically tall individuals are heavier than short ones.
Analysis of variance can be a good starting point in model proposal. In fact, looking at the system to be modelled, the user can be prompted to recognize some relationships existents between internal variables trying to model such relationships accordingly.

Multiple regression

In multiple regression, as well as in the analysis of variance, the goal is to find relationships between variables of a system. The difference in multiple regression, and in regression in general, is that such method tries to estimate such relationship rebuilding an equation that describe the behaviour of one or more dependant variables in function of one or more independent variables. There is more than one method in order to operate such regression whose main distinction can be seen from linear methods (where the equation obtained is linear in the input parameters) and non-linear methods (where the equation can be a polynomial or other functions).
Pushing further the concept of preliminary analysis of the system to simulate, we can use multiple regression in order to:

- guiding the modeling phase proposing some algorithm that code the so far discovered behaviour
- make the tuning of some initial parameters of the simulation before the simulation starts
- use the multiple regression above the real system and the modelled one in order to provide a degree of adherence of the model to the real world

Cluster analysis

In cluster analysis the goal is to retrieve some collections of individuals whose description (or behaviour) is alike. In clustering analysis, the users can define a distance measure based on the properties of single agents. Moreover he can recognizes if, within the system, are present well-defined set of individuals that are similar, based of the given distance measure.
This is useful in order to decrease the number of element to describe within the system. In fact, instead of focusing over the single agent behaviour in an object-oriented way, the user could look at the system as a set of clusters whose elements are in some way equivalent. Recognizing the fact that the description of single elements can be summarized by the description of few clusters can help to decrease the heterogeneity of the system.

Association rules

In this method the aim is to find regular patterns that describe categorical data and express such patterns using "if then" rules that recall a causal semantics. The rationale used to extract these rules is quite simple, the hard part is to apply it to huge amount of data. The method records the frequencies of certain patterns within a load of observations. For example, if every time the variable "a" has value "1" then the variable "b" has the value "0" we can deduce that the rule "$a=1 \rightarrow b=0$" holds. We are not able to say why it is like that, but the available observations give us a certain degree of certainty.
The causal semantics associated to the results and its algorithmic nature provide us with a natural instrument to explore the hidden model followed by the system

Iterative process in modeling phase

By using the above described methods, and many others not mentioned here, we can define a modeling and model revision process. Such process starts from the first task of model building (Model Building task in Figure 1) where a first proposal of model is done and will be tested after various runs. As we introduced in the first part of the paper, such task suffers from a set of initial hypothesis and it produces a first proposal of model used in the simulations. In this very first phase Data Mining (DM Analysis in Figure 1) can be used in order to make safe hypothesis over the real behavior of the system (or at least for that portion of the behavior that is observable, simulation is a good way to discover new scenarios that are not observed before).
When the simulation has produced a good amount of observations to work with (Simulation task in Figure 1) a new phase of Data Mining analysis can be used to make hypothesis above the model produced (DM Analysis task in Figure 1). Such results could validate or refuse the initial hypothesis about the real world and could guide a revision process in order to refine our knowledge about the overall system (Model Revision task in Figure 1).

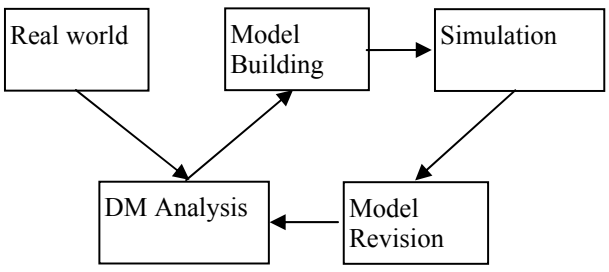

Figure 1: DM revision process applied to AB Simulation

Such iterative process could produce finer and finer model hypothesis until a desired convergence is found. Moreover, during the revision process the user could have a sound statistical theory as a guidance that provides him/her with a measure of the fitness of the model.

CONCLUSIONS AND FUTURE DIRECTIONS

In our work we explored the ways in which Data Mining techniques could be successfully applied to Agent Based Modeling and Simulation, in order to exploit hidden relations and emergent behavior. We found that Data Mining, and Machine Learning in general can be used in a number of ways in agent-based simulation, we can classify these contributions in two main tasks: Endogenous modeling, where Machine Learning can be used to provide the single agent a sort of intelligent behavior and Exogenous modeling, Where the final results of a simulation are analyzed using Data Mining techniques in order to reveal interesting patterns in data that could help to better model the behavior of the overall systems. We provide an overview of the tools that we think could be useful to accomplish this task: Analysis of variance, Multiple regression, Cluster analysis, Association rules. By using the above described methods, and many others not described here, we can define a modeling and model revision process.
In future works we plan to apply the techniques described here to simple agent based models and demonstrate they can be useful for model validation and hidden patterns analysis.

REFERENCES

Bensaid N., & Mathieu P. (1997), A hybrid architecture for hierarchical agents

Bonabeau, E. 2002. "Agent-based modeling: Methods and techniques for simulating human systems", PNAS 99 Suppl. 3: 7280-7287.

Brooks, R. A. (1990). Elephants Don't Play Chess, *Designing Autonomous Agents*, Pattie Maes, ed. Cambridge, MA: MIT Press

Franklin, S. (1995). Artificial Minds, Cambridge, MA: MIT Press

Franklin, S., & Graesser, A. (1997). Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents, *Proceedings of the Agent Theories, Architectures, and Languages Workshop, Berlin* (pp. 193-206). Springer Verlag.

Gilbert, N. and Terna, P. 2000. "How to build and use agent-based models in social science", Mind & Society 1, 57-72

Ginsberg M. (1989). Universal planning: An (almost) universally bad idea. *AI Magazine 10(4)* pp. 40-44

Langton, C. (1989). Artificial Life, Redwood City, CA: Addison-Wesley

Mataric, M. (1995). Issues and approaches in the design of collective autonomous agents. *Robotics and Autonomous Systems 16* (pp.321-331)

Ostrom T. (1988). Computer simulation: The third symbol system. *Journal of Experimental Social Psychology, 24* (pp. 381-392)

Remondino, M. (2003). Agent Based Process Simulation and Metaphors Based Approach for Enterprise and Social Modeling, *ABS 4 Proceedings* (pp.93-97). SCS Europ. Publish. House – ISBN 3-936-150-25-7

Woolridge, M., & Jennings, N. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review 10(2)* pp.115-152

AUTHORS BIOGRAPHY

MARCO REMONDINO was born in Asti, Italy, and studied Economics at the University of Turin, where he obtained his Master Degree in March, 2001 with 110/110 cum Laude et Menzione and a Thesis in Economical Dynamics. In the same year, he started attending a PhD at the Computer Science Department at the University of Turin, which he got in January 2005. His main research interests are Computer Simulation applied to Social Sciences, Enterprise Modeling, Agent Based Simulation, Multi Agent Systems and BDI agents. He has been part of the European team which defined a Unified Language for Enterprise Modeling (UEML). He is also participating to a University project for creating a cluster of computers, to be used for Social Simulation.

GIANLUCA CORRENDO was born the 12th of August 1974, he started his research in 1999 with a laurea thesis in model based diagnosis in a joint project sponsored by ASI (Italian Space Agency). Gianluca Correndo got his PhD in January 2005 with a thesis in ontologies in medical applications. His research activities are model based diagnosis, agent technologies, medical informatics and data mining. He has started a post doc in data mining field at the TOTAL CeRT (Centre of Research and Technique) where has the role of data mining engineer.