An Overview of the Interrelation Among Agent Systems, Learning Models and Formal Languages

Leonor Becerra-Bonache[1] and M. Dolores Jiménez-López[2](✉)

[1] Laboratoire Hubert Curien, Université Jean Monnet, Rue du Professeur Benoit Lauras 18, 42000 Saint Etienne, France
leonor.becerra@univ-st-etienne.fr

[2] Research Group on Mathematical Linguistics, Universitat Rovira i Virgili, Av. Catalunya 35, 43002 Tarragona, Spain
mariadolores.jimenez@urv.cat

Abstract. Considering the important role of interdiciplinarity in current research, this article provides an overview of the interchange of methods among three different areas: *agent technologies*, *learning models* and *formal languages*. The ability to learn is one of the most fundamental attributes of the intelligent behaviour. Therefore, any progress in the theory and computer modelling of learning processes is of great significance to fields concerning with understanding intelligence, and this includes, of course, artificial intelligence and intelligent agent technology. Agent technologies can offer good solutions and alternative frameworks to classic models in the area of computing languages and this can benefit formal models of learning. Formal language theory –considered as the stem of theoretical computer science– provides mathematical tools for the description of linguistic phenomena. This theory is central to grammatical inference, a subfield of machine learning. The interest of the interrelation among these disciplines is based on the idea that the collaboration among researchers in these areas can clearly improve their respective fields. Our goal here is to present the state-of-the art of the relationship among these three areas and to emphasize the importance of this interdisciplinary research.

1 Introduction

Nowadays, although disciplines are highly specialized, research areas are getting more and more interdisciplinary. Researchers realize that in order to solve problems in their respective areas, it is necessary to cross traditional academic boundaries. There is a great need to connect and integrate disciplines, methods and technologies in order to improve our knowledge. Subjects must be attacked from various angles and across disciplines. Interdisciplinarity should be an essential trait of research and, of course, the three areas considered in this paper are not an exception. This is why in this article we focus not on single fields but on the common space delimited by those three areas: agent technologies, learning

N.T. Nguyen (Ed.): TCCI XVII 2014, LNCS 8790, pp. 46–65, 2014.
DOI: 10.1007/978-3-662-44994-3_3

models, and formal language theory (see Fig. 1). The main goal here is to show how interdisciplinarity among people working in such disciplines can provide new models that may improve the current scientific results in artificial intelligence technologies.

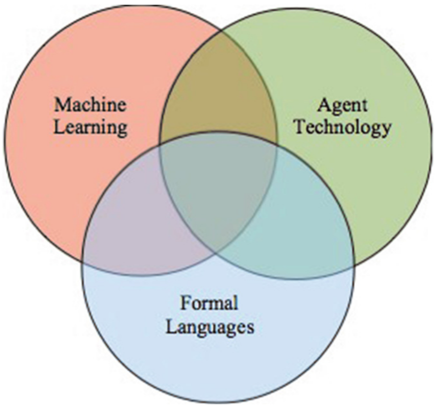

Fig. 1. Intersection among *machine learning*, *agent technology* and *formal language theory.*

Understanding human learning well enough to reproduce aspects of that learning capability in a computer system is a worthy scientific goal that have been considered by the research on *machine learning*, a field of artificial intelligence that aims to develop techniques that allow computers to learn. As Nilsson says, "a machine learns whenever it changes its structure, program or data (based on its inputs or in response to external information) in such a manner that its expected future performance improves" [35]. Machine learning techniques have been successfully applied to different domains, such as bio-informatics (e.g., gene finding), natural language processing (e.g., machine translation), speech and image recognition, robotics, etc.

Agent technology is one of the most important areas of research and development that have emerged in information technology in the 1990s. It can be defined as a *distributed artificial intelligence* approach to implement autonomous entities driven by beliefs, goals, capabilities, plans and agency properties. Roughly speaking, an agent is a computer system that is capable of flexible autonomous action in dynamic, unpredictable, multi-agent environments. The metaphor of autonomous problem solving entities cooperating and coordinating to achieve their objectives is a natural way of conceptualizing many problems. In fact, the multi-agent system literature spans a wide range of fields including robotics, mathematics, linguistics, psychology, and sociology, as well as computer science.

Formal languages originated from mathematics and linguistics as a theory that provides mathematical tools for the description of linguistic phenomena. The main goal of formal language theory is the syntactic finite specification of infinite languages. The theory was born in the middle of the 20th century as a tool for modelling and investigating the syntax of natural languages. However, very soon it developed as a new research field, separated from linguistics, with specific problems, techniques and results and, since then, it has had an important role in the field of computer science, in fact it is considered as the stem of theoretical computer science.

Taking into account the important advantages that collaboration among researchers of these fields can have for the area of artificial intelligence, we focus on the common space delimited by them and organize the paper as follows.

In Sect. 2 the relationship between *learning* and *formal languages* is taken into account. The theory of formal language is central to the field of machine learning, since there exists even an area called *grammatical inference* dealing with the process of learning formal grammars and languages from a set of data.

In Sect. 3, the relationship between *agents* and *formal languages* is considered. While in classic formal language theory, grammars and automata modelled computing devices where the computation was accomplished by one central agent, new models in formal languages take into account distributed computing. The idea of several devices collaborating for achieving a common goal was formalized in many subfields of formal language theory giving rise to the so-called *agent-based models* of formal languages.

In Sect. 4, we consider the relationship between *learning* and *agents*. The intersection of multi-agent systems and machine learning techniques have given rise to two different research areas [21]: (1) *learning in multi-agent systems* where machine learning solutions are applied to support agent technology and (2) *agent-based machine learning techniques* where agent technology is used in the field of machine learning with the interest on applying agent-based solutions to learning.

Finally, Sect. 5 concludes the paper by suggesting potential and promising directions of future research on the intersection among learning, agents and formal languages.

2 Learning and Formal Languages

The intersection between machine learning and formal languages constitutes a well-established research area known as *grammatical inference*. As A. Clark says, "grammatical inference is the study of machine learning of formal languages" [8]. This new area was born in the 1960s and since then has attracted the attention of researchers working on different fields, including machine learning, formal languages, computational linguistics, information theory, pattern recognition, and many others.

E.M. Gold [17] originated the study of grammatical inference and gave the initial theoretical foundations of this field. Motivated by the problem of children's language acquisition, E.M. Gold aimed "to construct a precise model for the intuitive notion able to speak a language in order to be able to investigate theoretically how it can be achieved artificially" [17]. After Gold's work, there has been developed a considerable amount of research to establish a grammatical inference theory, to find efficient methods for inferring formal grammars, and to apply these methods to practical domains, such as bioinformatics or natural language processing.

As H. Fernau and C. de la Higuera pointed out [16], there is a number of good reasons for formal language specialists to be interested in the field of grammatical inference, among others:

– Grammatical inference deals with formalisms describing formal languages, such us formal grammars, automata, etc.
– Grammatical inference uses formal language methodologies for constructing learning algorithms and for reasoning about them.
– Grammatical inference tries to give mathematical descriptions of the classes of languages that can be learned by a concrete learning algorithm.

Most of grammatical inference research has been focused on learning *regular* and *context-free* languages. Although these are the basic classes of the Chomsky hierarchy, it has been proved that even to learn these classes is already too hard under certain learning paradigms. Next, we review the main formal models proposed in this field and some of the main learnability results obtained.

2.1 Learning Paradigms

Broadly speaking, in a grammatical inference problem, we have a *teacher* that provides data to the learner (or learning algorithm), and a *learner* that must identify the underlying language from this data. Depending on the kind of data given to the learner, how this data is provided to it and the criteria used to say that a learner has successfully acquired the language, we can distinguish three main learning paradigms:

– Identification in the limit, proposed by Gold [17].
– Query learning, proposed by Angluin [1].
– Probably Approximately Correct learning (PAC), proposed by Valiant [48].

Imagine an adult and a child that is learning his native language. The adult uses his grammar, G, to construct sentences of his language, L. The child receives sentences and, after some time, he is able to use grammar G to construct sentences of L. From a mathematical point of view, the child is described by a learning algorithm, which takes a list of sentences as input and generates a language as output. Based on these ideas, Gold introduced a new formal model known as *identification in the limit* [17], with the ultimate goal of explaining the process of children's language acquisition. In this model, examples of the unknown language are presented to the learner, and the learner has to produce a hypothesis of this language. Its hypothesis is updated after receiving each example; if the new examples received are not consistent with the current hypothesis, it changes its hypothesis. However, at some point, always, the learner will found the correct hypothesis and will not change from it. Therefore, according to Gold's model, the learner identifies the target language in the *limit* if after a finite number of examples, the learner makes a correct hypothesis and does not change it from there on.

There are two traditional settings within Gold's model: (a) learning from text, where only examples of the target language are given to the learner (i.e., only *positive data*); (b) learning from informant, where examples that belong and do not belong to the target language are provided to the learner (i.e., *positive and negative* information).

It is desirable that learning can be achieved from only positive data, since in the most part of applications the available data is positive. However, one of Gold's main results is that *superfinite classes* of languages (i.e., classes of languages that contains all finite languages and at least one infinite language) *are not identifiable in the limit* from *positive data* [17]. This implies that even the class of regular languages is not identifiable in the limit from positive data. The intuitive idea is that, if the target language is a finite language contained in an infinite language, and the learner infers that the target language is the infinite language, it will not have any evidence to refute its hypothesis and it will never converge to the correct language. Due to these results, learning from only positive data is considered a hard task. However, learnability results have been obtained by studying subclasses of the languages to be learned, providing additional information to the learner, etc. For more details, see [14].

In Gold's model, the learner *passively* receives examples of the language. Angluin proposed a new learning model known as *query learning model* (or active learning), where the learner is allowed to *interact* with the teacher, by making questions about the strings of the language [1]. There are different kinds of queries, but the standard combination to be used are: (a) *membership queries*: the learner asks if a concrete string belongs to the target language and the teacher answers "yes" or "no"; (b) *equivalence queries*: the learner asks if its hypothesis is correct and the teacher answers "yes" if it is correct or otherwise gives a counterexample. According to Angluin's model, the learner has successfully learnt the target language if it returns the correct hypothesis after asking a finite number of queries.

The learnability of DFA (Deterministic Finite Automata) has been successfully studied in the context of query learning. One of the most important results in this framework was given by D. Angluin [1]. She proved that DFA can be identified in polynomial time using membership and equivalence queries. Later, there were developed more efficient versions of the same algorithm trying to increase the parallelism level, to reduce the number of EQs, etc. (see [3,18,41]). Moreover, some new type of queries have been proposed to learn DFA, such as corrections queries, that has led to some interesting results [5]. Angluin and Kharitonov [2] showed that the problem of identifying the class of context-free languages from membership and equivalence queries is computationally as hard as the cryptographic problems.

In order to obtain some positive learnability results for classes of languages more powerful than regular, researchers have used different techniques: to investigate subclasses of context-free languages, to give structural information to the learner, to reduce the problem to the learning of regular languages, etc. For more details, see [14].

In Gold's and Angluin's model, exact learning is required. However, this has always been considered a hard task to achieve. Based on these ideas, Valiant introduced the *PAC model*: a distribution-independent model of learning from random examples [48]. According to this model, there exist an unknown distribution over the examples, and the learner receives examples sampled under this distribution. The learner is required to learn under any distribution, but exact learning is not required (since one may be unlucky during the sampling process). A successful learning algorithm is one that with high probability finds a grammar whose error is small.

In the PAC learning model, the requirement that the learning algorithm must learn under any distribution is too hard and has led to very few positive results. Even for the case of DFA, most results are negative. For a review of some positive results in this model, see [14].

3 Agents and Formal Languages

Multi-agent systems offer strong models for representing complex and dynamic real-world environments. The formal apparatus of agent technology provides a powerful and useful set of structures and processes for designing and building complex applications. Multi-agent systems promote the interaction and cooperation of autonomous agents to deal with complex tasks. Taking into account that computing languages is a complex task, formal language theory [42] has taken advantage of the idea of formalizing architectures where a hard task is distributed among several task-specific agents that collaborate in the solution of the problem: in this case, the generation/recognition of language.

The first generation of formal grammars, based in rewriting, formalized classical computing models. The idea of several devices collaborating for achieving a common goal has given rise to a new generation of formal languages that form an agent-based subfield of the theory. *Colonies*, *grammar systems* and *eco-grammar systems* are examples of this new generation of formal languages. All these new types of formalisms have been proposed as grammatical models of agent systems. The main advantage of those agent-based models is that they increase the generative power of the system thanks to interaction, distribution and cooperation.

A third generation of formal language theory has started with the introduction of biological ideas in the field. In the last decades, natural computing has become the most extended framework where new models in formal language theory have been developed. DNA computing [39] is an example of those models. During the last years, systems biology and cellular biology have achieved an important development. These advances have provided new models for computer science. One of them is cellular computing, a model that emphasizes the concept of microbiological populations as well as the equilibrium of the devices and the relationships between the elements. P systems [38] can be considered an example of this emerging paradigm. On the other hand, natural computing has evolved from the first numeric models –like neural networks– to symbolic models –as cellular computing– which are closer to multi-agent systems. *Networks of evolutionary processors* (NEPs) [7] are inspired in both, bio cellular models and basic structures for parallel and distributed symbolic processing. The main reason for adopting such theoretical perspectives is the need to reach a more realistic human-designed computing, both understanding the processes the nature carries out and taking advantage of the natural mechanisms that science is discovering.

3.1 Colonies

Colonies as well-formalized language generating devices have been proposed in [24], and developed during the nineties in several directions in many papers [4,12,23,25,26,30,31,37,45,46].

Colonies can be thought of as grammatical models of multi-agent systems motivated by Brooks' subsumption architectures [6]. They describe language classes in terms of behaviour of collections of very simple, purely reactive, situated agents with emergent behaviour.

A colony consists of a finite number of simple agents which generate finite languages and operate on a shared string of symbols –the *environment*– without any explicitly predefined strategy of cooperation. Each component has its own reactive behaviour which consists in: (1) sensing some aspects of the *context* and (2) performing elementary tasks on it in order to achieve some local changes.

Formally, a colony is defined as follows.

Definition 1. *A colony C is a 3-tuple: $C = (R, V, T)$, where $R = \{R_i | 1 \leq i \leq n\}$ is a finite set of regular grammars $R_i = (N_i, T_i, P_i, S_i)$ producing finite languages $L(R_i) = F_i$ for each i. R_i will be referred to as a component of C; $V = \bigcup_{i=1}^{n}(T_i \cup N_i)$ is the alphabet of the colony; $T \subseteq V$ is the terminal alphabet of the colony.*

Components $R_i \in R$ $(1 \leq i \leq n)$ of a colony C are *regular grammars* generating finite languages and operating on a shared string of symbols –the *environment*– without any explicitly predefined strategy of cooperation of the components.

An *environment* of a colony is formed by strings of symbols from V. Strings are modified only by sequential activities of components of a colony. Because of the lack of any predefined strategy of cooperation between components, each component may participate in the rewriting of the current string whenever its start symbol is present in the current string. Conflicts are solved non-deterministically, as it is usual in classical theory of formal grammars. The activity of components in a colony is performed by string transformation on a common tape. Elementary changes of strings are determined by a basic derivation step:

Definition 2. *For $x, y \in V^*$ we define $x \Longrightarrow y$ iff $x = x_1 S_i x_2$, $y = x_1 z x_2$, where $z \in F_i$ for some $i, 1 \leq i \leq n$.*

The behaviour of a colony is defined as the set of all strings which can be generated by the colony from a given starting string. A terminal symbol of one component can occur as a non-terminal symbol of another one, so that the possibility of cooperation of components of the colony allows to generate substantially more than finite languages. The global behaviour of the whole colony emerges from the strictly individual behaviours of components.

Definition 3. *The language determined by a colony C starting with the word $w_0 \in V^*$ is given by:* $L(C, w_0) = \{v | w_0 \Longrightarrow^* v, v \in T^*\}$.

A colony is outlined in Fig. 2.

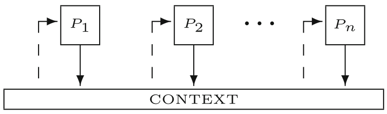

Fig. 2. A colony.

Colonies offer a formal framework for the emergence of complex behaviours by using purely reactive simple components. The main advantage of colonies is their generative power, the class of languages describable by colonies that make use of strictly regular components is beyond the set describable in terms of individual regular grammars.

3.2 Grammar Systems

Grammar system theory is a consolidated and active branch in the field of formal languages that provides syntactic models for describing multi-agent systems at the symbolic level, using tools from formal grammars and languages. The attempt of the 'parents' of the theory was "to demonstrate a particular possibility of studying complex systems in a purely syntactic level" [10] or, what is the same, to propose a grammatical framework for multi-agent systems.

A grammar system is a set of grammars working together, according to a specified protocol, to generate a language. Note that while in classical formal language theory *one* grammar (or automaton) works individually to generate (or recognize) *one* language, here we have *several* grammars working together in order to produce *one* language.

The theory was launched in 1988 [9], when Cooperating Distributed Grammar Systems (CDGS) were proposed as a syntactic model of the blackboard architecture of problem solving. A CDGS consists of a finite set of generative grammars with a common sentential form (axiom) that cooperate in the derivation of a common language. Component grammars generate the string in turns (thus, sequentially), under some cooperation protocol. At each moment in time, one grammar (and just one) is active, this is, rewrites the common string, while the rest of grammars of the CDGS are inactive. Conditions under which a component can start/stop its activity on common sentential form are specified in the cooperation protocol. Terminal strings generated in this way form the language of the system.

An analogy can be drawn between CDGS and the blackboard model of problem solving described in [34] as consisting of three major components: (1) *Knowledge sources.* The knowledge needed to solve the problem is partitioned into knowledge sources, which are kept separate and independent; (2) *Blackboard data structure.* Problem solving state data are kept in a global database, the *blackboard.* Knowledge sources produce changes in the blackboard that lead incrementally to a solution to the problem. Communication and interaction among knowledge sources take place solely through the blackboard; (3) *Control.* Knowledge sources respond opportunistically to changes in the blackboard. There is a set of control modules that monitor changes in the blackboard and decide what actions to take next. Criteria are provided to determine when to terminate the process. In CDGS, component grammars correspond to knowledge sources. The common sentential form in CDGS plays the same role as the blackboard data structure. And finally, the protocol of cooperation in CDGS encodes control on the work of knowledge sources. The rewriting of a non-terminal symbol can be interpreted as a developmental step on the information contained in the current state of the blackboard. And, finally, a solution to the problem corresponds to a terminal word.

One year later, in 1989, Parallel Communicating Grammar Systems (PCGS) were introduced as a grammatical model of parallelism [40]. A PCGS consists of several grammars with their respective sentential forms. In each time unit, each component uses a rule, which rewrites the associated sentential form. Cooperation among agents takes place thanks to the so-called *query symbols* that allow communication among components.

If CDGS were considered a grammatical model of the blackboard system in problem solving, PCGS can be thought of as a formal representation of the *classroom model.* Let us take the blackboard model and make the following modifications: (1) Allow each knowledge source to have its own 'notebook' containing the description of a particular subproblem of a given problem; (2) Allow each knowledge source to operate only on its own 'notebook' and let exist one distinguished agent which operates on the 'blackboard' and has the description of the problem; (3) and finally, allow agents to communicate by request the content of their own 'notebook'. These modifications on the blackboard model lead to the 'classroom model' of problem solving where the classroom leader (the master) works on the blackboard while pupils have particular problems to solve in their notebooks. Master and pupils can communicate and the global problem is solved through such cooperation on the blackboard. An easy analogy can be established between PCGS and the classroom model: pupils correspond to grammars which make up the system, and their notebooks correspond to the sentential forms. The set of rules of grammars encodes the knowledge of pupils. The distinguished agent corresponds to the 'master'. Rewriting a nonterminal symbol is interpreted as a developmental step of the information contained in the notebooks. A partial solution, obtained by a pupil corresponds to a terminal word generated in one grammar, while solution of the problem is associated to a word in the language generated by the 'master.'

The sequential CDGS and the parallel PCGS are the two main types of grammar systems. However, since 1988, the theory has developed into several directions, motivated by several scientific areas. Besides distributed and decentralized artificial intelligence, artificial life, molecular computing, robotics, natural language processing, ecology, sociology, etc. have suggested some modifications of the basic models, and have given rise to the appearance of different variants and subfields of the theory. For more information on those new types see [10,13].

3.3 Eco-Grammar Systems

Eco-grammar systems have been introduced in [11] and provide a syntactical framework for eco-systems, this is, for communities of evolving agents and their interrelated environment. An eco-grammar system is defined as a multi-agent system where different components, apart from interacting among themselves, interact with a special component called 'environment'. Within an eco-grammar system we can distinguish two types of components *environment* and *agents*. Both are represented at any moment by a string of symbols that identifies the current state of the component. These strings change according to sets of evolution rules. Interaction among agents and environment is carried out through agents' actions performed on the environmental state by the application of some productions from the set of action rules of agents.

An eco-grammar system can be thought of as a generalization of CDGS and PCGS. If we superpose a CDGS and a PCGS, we obtain a system consisting of grammars that contain individual strings (like in PCGS) and a common string (like in CDGS). If we call this common string *environment* and we mix the functioning of CDGS and PCGS, letting each component to work on its own string and on the environmental string, something similar to an ecosystem is obtained. If we add one more grammar, expressing evolution rules of the environment, and we make evolution of agents depend on the environmental state, the thing we obtain is an eco-grammar system.

The concept of eco-grammar system is based on six postulates formulated according to properties of artificial life [27]:

1. An ecosystem consists of an *environment* and a *set of agents*.
2. In an ecosystem there is a *universal clock* which marks time units, the same for all the agents and for the environment, according to which agents and environment evolution is considered.
3. Both *environment* and agents have characteristic *evolution rules* which are in fact L systems [22,28], hence are applied in a parallel manner to all the symbols describing agents and environment; such a (rewriting) step is done in each time unit.
4. *Evolution rules of environment* are *independent* of agents and on the state of the environment itself. Evolution rules of agents *depend on* the state of the environment.

5. Agents act on the environment according to *action rules*, which are pure rewriting rules used sequentially. In each time unit, each agent uses one action rule which is chosen from a set depending on the current state of the agent.
6. *Action has priority over evolution* of the environment. At a given time unit exactly the symbols which are not affected by action are rewritten by evolution rules.

Main features of eco-grammar systems are captured in Fig. 3.

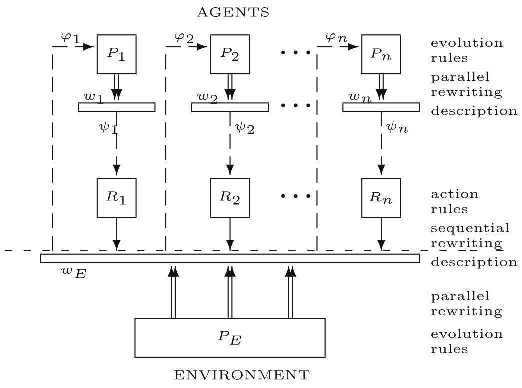

Fig. 3. An eco-grammar system.

Definition 4. *An Eco-Grammar System (EG-system) is an $n+1$-tuple: $\Sigma = (E, A_1, \ldots, A_n)$, where:*

- *$E = (V_E, P_E)$, V_E is a finite alphabet; P_E is a finite set of 0L rules over V_E.*
- *$A_i = (V_i, P_i, R_i, \varphi_i, \psi_i)$ for i, $1 \leq i \leq n$, where: V_i is a finite alphabet; P_i is a finite set of 0L rules over V_i; R_i is a finite set of rewriting rules of the form $x \rightarrow y$ with $x \in V_E^+$, $y \in V_E^*$;*
- *$\varphi_i : V_E^* \longrightarrow 2^{P_i}$;*
- *$\psi_i : V_i^+ \longrightarrow 2^{R_i}$.*

Definition 5. *A state of an Eco-Grammar System $\Sigma = (E, A_1, \ldots, A_n)$ is an $(n+1)$-tuple: $\sigma = (w_E, w_1, w_2, \ldots, w_n)$, where $w_E \in V_E^*$ and $w_i \in V_i^*$, $1 \leq i \leq n$; w_E is the state of the environment, and w_i is the state of i-th agent, $1 \leq i \leq n$.*

Definition 6. *Let $\sigma = (w_E, w_1, w_2, \ldots, w_n)$ be a state of EG-system $\Sigma = (E, A_1, \ldots, A_n)$. Agent A_i is said to be active in state σ if the set of its current action rules, this is $\psi_i(w_i)$, is nonempty. By an action of an active agent A_i in state σ we mean an application of an action rule r, $r \in \psi_i(w_i)$, to the environmental state w_E. A simultaneous action of agents $A_{i_1}, \ldots, A_{i_r}$, $\{i_1, \ldots, i_r\} \in \{1, \ldots, n\}$ being active in state σ, onto the environment is a parallel derivation step $w_E \Longrightarrow w'_E$ such that $w_E = x_1 u_1 x_2 u_2 \ldots u_r x_{r+1}$, and $w'_E = x_1 v_1 x_2 v_2 \ldots v_r x_{r+1}$, where $u_j \rightarrow v_j \in \psi_{i_j}(w_{i_j})$, $1 \leq j \leq r$, and $x_i \in V_E^*$, $1 \leq i \leq r+1$.*

Definition 7. *Let $\sigma = (w_E, w_1, w_2, \ldots, w_n)$ be a state of EG-system $\Sigma = (E, A_1, \ldots, A_n)$. We say that w_i' is a current evolution of agent A_i in state w_i, if w_i' can be derived from w_i by productions of $\varphi_i(w_E)$, in 0L-manner, $1 \leq i \leq n$. For two states w_E and w_E' of the environment we say that w_E' is an evolution of w_E if w_E' can be derived from w_E by productions of P_E in 0L-manner. A change of a state of an Eco-Grammar System means an evolution of the state of every agent and an evolution of the environment at each place except some distinguished ones where currently active agents perform simultaneously an action.*

Definition 8. *Let $\sigma = (w_E, w_1, w_2, \ldots, w_n)$ and $\sigma' = (w_E', w_1', w_2', \ldots, w_n')$ be two states of EG-system $\Sigma = (E, A_1, \ldots, A_n)$. We say that σ changes into σ', written as: $\sigma \Longrightarrow_\Sigma \sigma'$, iff the following conditions hold: (i) w_E' arises from w_E by an evolution affected by all the active agents in state σ: $w_E = z_1x_1z_2x_2 \ldots z_mx_mz_{m+1}$ and $w_E' = z_1'y_1z_2'y_2 \ldots z_m'y_mz_{m+1}'$ such that: $z_1x_1z_2x_2 \ldots x_mz_{m+1} \Longrightarrow z_1y_1z_2y_2 \ldots y_mz_{m+1}$ is a simultaneous action of all the agents $A_{i_1}, \ldots, A_{i_m}$, $\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$, that are active in state σ and, $z_1'z_2' \ldots z_{m+1}'$ is an evolution of $z_1z_2; \ldots z_{m+1}$; (ii) w_i' is an evolution of A_i in state w_i, $1 \leq i \leq n$.*

3.4 NEPs-Networks of Evolutionary Processors

Networks of Evolutionary Processors (NEPs) are new computing mechanisms directly inspired in the behaviour of cell populations. NEPs, introduced in [7,29], can be defined as systems consisting of several devices whose communication is regulated by an underlying graph. Such devices, which are an abstract formalization of cells, are described by a set of words (DNA) evolving by mutations, according to some predefined rules. Their outcome travels to the other nodes if they accept it after passing a filtering process. At the end of the process, only the cells with correct strings will survive.

The cellular basis of NEPs relate them with P systems, especially with tissue P systems [32,33]. In tissue P systems, cells form a multitude of different associations performing various functions. NEPs could be linked to systems biology as well, because the model aims to develop a holistic theory where the behaviour of each agent can influence the environment and the other agents. From the computational point of view, NEPs are related to the Connection Machine [19] and the Logic Flow paradigm [15]. Another important theoretical relationship of NEPs is the theory of grammar and eco-grammar systems [10,11] which share with NEPs the idea of several devices working together and exchanging results.

With all this background and theoretical connections, it is easy to understand how NEPs can be described as agential bio-inspired context-sensitive systems. Many disciplines are needed of these types of models that are able to support a biological framework in a collaborative environment. The conjunction of these features allows applying the system to a number of areas, beyond generation and recognition in formal language theory.

Definition 9. *A Network of Evolutionary Processors of size n is a construct: $\Gamma = (V, N_1, N_2, ..., N_n, G)$, where:*

- *V is an alphabet and for each $1 \leq i \leq n$,*
- *$N_i = (M_i, A_i, PI_i, PO_i)$ is the i-th evolutionary node processor of the network. The components of every processor are (we denote by e the empty word): M_i is a finite set of evolution rules of one of the following forms only (i) $a \rightarrow b, a, b \in V$ (substitution rules), (ii) $a \rightarrow e, a \in V$ (deletion rules), (iii) $e \rightarrow a, a \in V$ (insertion rules); A_i is a finite set of strings over V. The set A_i is the set of initial strings in the i-th node; PI_i and PO_i are subsets of V^* representing the input and the output filter, respectively. These filters are defined by the membership condition, namely a string $w \in V^*$ can pass the input filter (the output filter) if $w \in PI_i (w \in PO_i)$.*
- *$G = (\{N_1, N_2, \ldots, N_n\}, E)$ is an undirected graph called the underlying graph of the network. The edges of G, that is the elements of E, are given in the form of sets of two nodes. The complete graph with n vertices is denoted by K_n.*

Definition 10. *Configuration of a NEP is an n-tuple $C = (L_1, L_2, \ldots, L_n)$, with $L_i \subseteq V^*$ for all $1 \leq i \leq n$. It represents the sets of strings which are present in all the nodes at a given moment.*

A given configuration of a NEP can change either by an evolutionary step or by a communicating step. When changing by an evolutionary step, each component L_i of the configuration is changed in accordance with the evolutionary rules associated with the node i. The change in the configuration by an evolutionary step is written as $C_1 \Rightarrow C_2$. When changing by a communication step, each node processor N_i sends all copies of the strings it has which are able to pass its output filter to all the node processors connected to N_i and receives all copies of the strings sent by any node processor connected with N_i, if they can pass its input filter. The change in the configuration by a communication step is written as $C_1 \vdash C_2$.

A scheme of the basic architecture of NEPs is shown in Fig. 4.

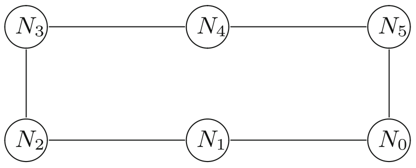

Fig. 4. Basic architecture of a NEP.

4 Learning and Agents

The intersection of agent technology and machine learning constitutes a research area whose importance is nowadays broadly acknowledged in artificial intelligence: *learning in multi-agent systems*. This new area has emerged as a topic of research in the late 1980s and since then has attracted increasing attention in both the multi-agent systems community and the machine learning area. However, until the late 80s, multi-agent learning had been widely ignored by both researchers in distributed artificial intelligence and in machine learning. This situation was due to two facts: (1) work in distributed artificial intelligence mainly concentrated on developing multi-agent systems whose organization and functioning were fixed and, (2) research in machine learning mainly concentrated on learning techniques and methods for single-agent settings [51].

Nowadays, it is commonly agreed by distributed artificial intelligence and machine learning communities that multi-agent learning –this is, learning that requires the interaction among several intelligent agents [20]-deserves particular attention. Two important reasons for the interest in studying learning in multi-agent systems have been stressed [49]:

1. The need for learning techniques and methods in the area of multi-agent systems in order to equip multi-agent systems with learning abilities to allow agents to automatically improve their behaviour.
2. The need in the area of machine learning of considering not only single-agent learning but also multi-agent learning in order to improve the understanding of the learning processes in natural multi-agent systems (like human groups or societies).

The area of multi-agent learning shows how developments in the fields of machine learning and agent technologies have become complementary. In this intersection, researchers from both fields have opportunities to profit from solutions proposed by each other. In fact we can distinguish two directions in this intersection [21]:

1. *Learning in Multi-Agent Systems* (MAS), this is, using machine learning techniques in agent technology.
2. *Agent-Based Machine Learning*, this is, using agent technology in the field of machine learning.

4.1 Learning in Multi-agent Systems

Learning is increasingly being seen as a key ability of agents and, therefore, several agent-based frameworks that utilize machine learning for intelligent decision support have been reported. Theoretical developments in the field of learning agents focus mostly on methodologies and requirements for constructing multi-agent systems with learning capabilities.

Many terms can be found in the literature that refer to learning in multi-agent systems [43]: *mutual learning, cooperative learning, collaborative learning, co-learning, team learning, social learning, shared learning, pluralistic learning*, and *organizational learning* are just some examples.

In the area of multi-agent learning –the application of machine learning to problems involving multiple agents [36]–, two principal forms of learning can be distinguished [43,49]:

1. *Centralized or isolated learning* where the learning process is executed by one single agent and does not require any interaction with other agents.
2. *Decentralized, distributed, collective or interactive learning* where several agents are engaged in the same learning process and the learning is done by the agents as a group.

There are three main methods/approaches to learning in multi-agent systems which are distinguished by taking into account the kind of feedback provided to the learner [36,49]:

1. *Supervised learning*, where the correct output is provided. This means that the environment or an agent providing feedback acts as a "teacher".
2. *Reinforcement learning*, where an assessment of the learner's output is provided. This means that the environment or an agent providing feedback acts as a "critic".
3. *Unsupervised learning*, where no explicit feedback is provided at all. This means that the environment or an agent providing feedback acts as an "observer".

A third classification can be obtained by taking into account the learning strategies used by agents [43]. The main difference between the types of learning included in this classification is the amount of learning effort required:

1. *Rote learning* consists of the direct implantation of knowledge and skills.
2. *Learning form instruction and by advice taking.* This type of learning transforms new information (instruction or advice) into an internal representation and integrates it with prior knowledge.
3. *Learning from examples and practice* consists of the extraction and refinement of knowledge from positive and negative examples or from practical experience.
4. *Learning by analogy* is as solution-preserving transformation of knowledge from a solved problem to a similar unsolved problem.
5. *Learning by discovery* consists of gathering new knowledge by observations, experiments, testing hypotheses or theories on the basis of the observational and experimental results.

Space limitation prevents us of going deeper in the above models. For more information the reader can see [36,43,44,47,49–51]. Our goal in this section has been just to stress the fact that several dimensions of multi-agent interaction can be subject to learning –when to interact, with whom to interact, how to interact, and what exactly the content of the interaction should be [20]–, and machine learning can be seen as a primer supplier of learning capabilities for agent and multi-agent systems.

4.2 Agent-Based Machine Learning

In the intersection between multi-agent systems and machine learning we find the so-called *agent-based machine learning techniques* where agent technology is applied to solve machine learning problems. According to Jedrzejowicz [21], there are several ways in which the research of machine learning can profit from the application of agent technology:

- First of all, there are machine learning techniques where parallelization can speed-up learning, therefore, in these cases using a set of agents may increase the efficiency of learning.
- Secondly, there are machine learning techniques that rely on the collective computational intelligence paradigm, where a synergetic effect is expected from combining efforts of various agents.
- Thirdly, in the so-called distributed machine learning problems, a set of agents working in distributed sites can be used to produce some local level solutions independently and in parallel.

Taking into account those advantages, several models have been proposed that apply agent-based solutions to machine learning problems:

- Models of collective or collaborative learning.
- Learning classifier systems that use agents representing set of rules as a solution to machine learning problem.
- Ensemble techniques.
- Distributed learning models.

According to [21], agent technology has brought to machine learning several capabilities including parallel computation, scalability and interoperability. In general, agent based solutions can be used to develop more flexible machine learning tools. For the state of the art of agent-based machine learning see [21].

5 Conclusions

According to [49], the interest in multi-agent systems is founded on the insight that many real-world problems are best modelled using a set of agents instead of a single agent. Multi-agent modelling makes possible to cope with natural constraints like the limitation of the processing power of a single agent and to profit from inherent properties of distributed systems like robustness, fault tolerance, parallelism and scalability. These properties have facilitated the application of multi-agent technology to many types of systems that help humans to perform several tasks.

Machine learning is one of the core fields of artificial intelligence, since artificial intelligence has been defined as "the science and engineering of making intelligent machines" and the ability to learn is one of the most fundamental attributes of intelligent behaviour. It is usually agreed that a system capable of learning deserves to be called intelligent; and conversely, a system being considered as intelligent is, among other things, usually expected to be able to learn.

Formalization has a long tradition in science, besides traditional fields such as physics or chemistry, other scientific areas such as medicine, cognitive and social sciences and linguistics have shown a tendency towards formalization. The use of formal methods has led to numerous results that would have been difficult to be obtained without such formalization. Formal language theory provides good tools to formalize different problems. This flexibility and abstraction has been proven by the application of formal languages to the fields of linguistics, economic modelling, developmental biology, cryptography, sociology, etc.

From what we have said, it follows that multi-agent systems, machine learning and formal language theory provide flexible and useful tools that can be used in different research areas due to their versatility. All three areas have revealed to be very useful for dealing with complex systems. MAS provide principles for the construction of complex systems and mechanisms for coordination. Formal language theory offers mathematical tools to formalize complex systems. And machine learning techniques help to deal with the complexity of complex systems by endowing agents with the ability of improving their behaviour. We have seen in this paper that some intersection between those areas has been already performed: *agents with learning, agents with formal languages* and *formal languages with learning.*

Future research should help to further integrate the three fields considered in this paper in order to obtain what in [20] is seen as a must: a *formal theory of multi-agent learning.*

Another important and challenging working direction is the application of this formal theory of multi-agent learning to a real world domain as is the area of processing natural language. The interaction between researchers in those three topics can provide good techniques and methods for improving our knowledge about how languages are processed. The advances in the area of natural language processing may have important consequences in the area of artificial intelligence since they can help the design of technologies in which computers will be integrated into the everyday environment, rendering accessible a multitude of services and applications through easy-to-use human interfaces.

Acknowledgements. The work of Leonor Becerra-Bonache has been supported by Pascal 2 Network of Excellence. The work of M. Dolores Jiménez-López has been supported by the Spanish Ministry of Science and Innovation under the Coordinated Research Project TIN2011-28260-C03-00 and the Research Project TIN2011-28260-C03-02.

References

1. Angluin, D.: Learning regular sets from queries and counterexamples. Inf. Comput. **75**, 87–106 (1987)
2. Angluin, D., Kharitonov, M.: When won't membership queries help? In: STOC'91: 24th Annual ACM Symposium on Theory of Computing, pp. 444–453. ACM Press, New York (1991)
3. Balcázar, J.L., Díaz, J., Gavaldà, R., Watanabe, O.: Algorithms for learning finite automata from queries: a unified view. In: Du, D.Z., Ko, K.I. (eds.) Advances in Algorithms, Languages, and Complexity, pp. 73–91. Kluwer Academic Publishers, Dordrech (1997)

4. Baník, I.: Colonies with position. Comput. Artif. Intell. **15**, 141–154 (1996)
5. Becerra-Bonache, L., Dediu, A.-H., Tîrnăucă, C.: Learning DFA from Correction and equivalence queries. In: Sakakibara, Y., Kobayashi, S., Sato, K., Nishino, T., Tomita, E. (eds.) ICGI 2006. LNCS (LNAI), vol. 4201, pp. 281–292. Springer, Heidelberg (2006)
6. Brooks, R.A.: Elephants don't play chess. Robot. Auton. Syst. **6**, 3–15 (1990)
7. Castellanos, J., Martín-Vide, C., Mitrana, V., Sempere, J.M.: Networks of evolutionary processors. Acta Informatica **39**, 517–529 (2003)
8. Clark, A.: Grammatical inference and first language acquisition. In: Workshop on Psychocomputational Models of Human Language Acquisition, Geneva, pp. 25–32 (2004)
9. Csuhaj-Varjú, E., Dassow, J.: On cooperating/distributed grammar systems. J. Inf. Process. Cybern. (EIK) **26**, 49–63 (1990)
10. Csuhaj-Varjú, E., Dassow, J., Kelemen, J., Păun, G.: Grammar Systems: A Grammatical Approach to Distribution and Cooperation. Gordon and Breach, London (1994)
11. Csuhaj-Varjú, E., Kelemen, J., Kelemenová, A., Păun, G.: Eco-grammar systems: a grammatical framework for life-like interactions. Artif. Life **3**(1), 1–28 (1996)
12. Dassow, J., Kelemen, J., Păun, G.: On parallelism in colonies. Cybern. Syst. **14**, 37–49 (1993)
13. Dassow, J., Păun, G., Rozenberg, G.: Grammar systems. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 2, pp. 155–213. Springer, Berlin (1997)
14. de la Higuera, C.: Grammatical Inference: Learning Automata and Grammars. Cambridge University Press, Cambridge (2010)
15. Errico, L., Jesshope, C.: Towards a new architecture for symbolic processing. In: Plander, I. (ed.) Artificial Intelligence and Information-Control Systems of Robots'94, pp. 31–40. World Science, Singapore (1994)
16. Fernau, H., de la Higuera, C.: Grammar induction: an invitation to formal language theorists. Grammars **7**, 45–55 (2004)
17. Gold, E.M.: Language identification in the limit. Inf. Control **10**, 447–474 (1967)
18. Hellerstein, L., Pillaipakkamnatt, K., Raghavan, V., Wilkins, D.: How many queries are needed to learn? In: 27th Annual ACM Symposium on the Theory of Computing, pp. 190–199. ACM Press (1995)
19. Hillis, W.D.: The Connection Machine. MIT Press, Cambridge (1985)
20. Huhns, M., Weiss, G.: Guest editorial. Mach. Learn. **33**, 123–128 (1998)
21. Jędrzejowicz, P.: Machine learning and agents. In: O'Shea, J., Nguyen, N.T., Crockett, K., Howlett, R.J., Jain, L.C. (eds.) KES-AMSTA 2011. LNCS, vol. 6682, pp. 2–15. Springer, Heidelberg (2011)
22. Kari, L., Rozenberg, G., Salomaa, A.: L Systems. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 1, pp. 253–328. Springer, Berlin (1997)
23. Kelemen, J.: Colonies - a theory of reactive agents. In: Kelemenová, A. (ed.) Proceedings on the MFCS'98 Satellite Workshop on Grammar Systems, pp. 7–38. Silesian University, Opava (1998)
24. Kelemen, J., Kelemenová, A.: A grammar-theoretic treatment of multiagent systems. Cybern. Syst. **23**, 621–633 (1992)
25. Kelemenová, A.: Timing in colonies. In: Păun, G., Salomaa, A. (eds.) Grammatical Models of Multi-agent Systems. Gordon and Breach, London (1999)
26. Kelemenová, A., Csuhaj-Varjú, E.: Languages of colonies. Theoret. Comput. Sci. **134**, 119–130 (1994)

27. Langton, C.: Artificial life. In: Life, A. (ed.) Artificial Life, pp. 1–47. Addison-Wesley, Redwood City (1989)
28. Lindenmayer, A.: Mathematical models for cellular interaction in development, I and II. J. Theor. Biol. **18**, 280–315 (1968)
29. Martín-Vide, C., Mitrana, V., Pérez-Jiménez, M., Sancho-Caparrini, F.: Hybrid networks of evolutionary processors. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 401–412. Springer, Heidelberg (2003)
30. Martín-Vide, C., Păun, G.: PM-colonies. Comput. Artif. Intell. **17**, 553–582 (1998)
31. Martín-Vide, C., Păun, G.: New topics in colonies theory. Grammars **1**, 209–223 (1999)
32. Martín-Vide, C., Păun, G., Pazos, J., Rodríguez-Patón, A.: Tissue P systems. Theoret. Comput. Sci. **296**(2), 295–326 (2002)
33. Martín-Vide, C., Pazos, J., Păun, G., Rodríguez-Patón, A.: A new class of symbolic abstract neural nets: tissue P systems. In: Ibarra, O.H., Zhang, L. (eds.) COCOON 2002. LNCS, vol. 2387, pp. 290–299. Springer, Heidelberg (2002)
34. Nii, H.P.: Blackboard systems. In: Barr, A., Cohen, P.R., Feigenbaum, E.A. (eds.) The Handbook of Artificial Intelligence, vol. IV, pp. 3–82. Addison-Wesley, Stanford (1989)
35. Nilsson, N.J.: Introduction to Machine Learning. An Early Draft of a Proposed Textbook (1998). http://robotics.stanford.edu/people/nilsson/mlbook.html
36. Panait, L., Luke, S.: Cooperative multi-agent learning: the state-of-the-art. Auton. Agent. Multi-Agent Syst. **11**(3), 387–434 (2005)
37. Păun, G.: On the generative power of colonies. Kybernetika **31**, 83–97 (1995)
38. Păun, G.: Computing with membranes. J. Comput. Syst. Sci. **61**(1), 108–143 (2000)
39. Păun, G., Rozenberg, G., Salomaa, A.: DNA Computing: New Computing Paradigms. Springer, Berlin (1998)
40. Păun, G., Sântean, L.: Parallel communicating grammar systems: the regular case. Ann. Univ. Bucharest **38**, 55–63 (1989)
41. Rivest, R.L., Schapire, R.E.: Inference of finite automata using homing sequences. Inf. Comput. **103**(2), 299–347 (1993)
42. Rozenberg, G., Salomaa, A. (eds.): Handbook of Formal Languages. Springer, Berlin (1997)
43. Sen, S., Weiss, G.: Learning in multiagent systems. In: Weiss, G. (ed.) Multiagent Systems, pp. 259–298. MIT Press, Cambridge (1999)
44. Shoham, Y., Powers, R., Grenage, T.: If multi-agent learning is the answer, what is the question? Artif. Intell. **171**(7), 365–377 (2007)
45. Sosík, P.: Parallel accepting colonies and neural networks. In: Păun, G., Salomaa, A. (eds.) Grammatical Models of Multi-agent Systems. Gordon and Breach, London (1999)
46. Sosík, P., Štýbnar, L.: Grammatical inference of colonies. In: Păun, G., Salomaa, A. (eds.) New Trends in Formal Languages. LNCS, vol. 1218. Springer, Heidelberg (1997)
47. Stone, P., Veloso, M.: Multiagent systems: a survey from a machine learning perspective. Auton. Robot. **8**, 345–383 (2000)
48. Valiant, L.G.: A theory of the learnable. Commun. ACM **27**, 1134–1142 (1984)
49. Weiss, G.: Learning to coordinate actions in multi-agent systems. In: Proceedings of the 13th International Conference on Artificial Intelligence, pp. 311–316 (1993)

50. Weiss, G.: A multiagent perspective of parallel and distributed machine learning. In: Proceedings of the 2nd International Conference on Autonomous Agents, pp. 226–230 (1998)
51. Weiss, G., Dillenbourg, P.: What is "Multi" in multi-agent learning? In: Dillenbourg, P. (ed.) Collaborative Learning: Cognitive and Computational Approaches (Chapter 4). Pergamon Press, Oxford (1998)