The Semantic Web Paradigm for a Real-Time Agent Control (Part II)

Mazilescu, Vasile

Dunarea de Jos University Galati, Romania

17 February 2010

The Semantic Web Paradigm for a Real-Time Agent Control (Part II)

Vasile MAZILESCU

Department of Accounting and Economic Informatics
University Dunărea de Jos of Galati
vasile.mazilescu@ugal.ro

Abstract. This paper is the second part of The Semantic Web Paradigm for a Real-time Agent Control, and the goal is to present the predictability of a multiagent system used in a learning process for a control problem (**MASLCP**).

Keywords: learning process, fuzzy control, agent predictability
Jel code: C63, C 88

1. Introduction

Learning can be informally defined as follows: the acquisition and the incorporation of new knowledge and cognitive skills in the future system dynamical properties, provided that this acquisition and incorporation is conducted by the system itself and leads to an improvement in its performance. Machine learning (**ML**) dealt with the computational aspects of learning, as a centralized and isolated process that occurs in intelligent stand-alone systems. The main concern in most cases of learning is the incremental acquirement, the modification, consolidation and adjustment of the knowledge models for a specific domain [1,2]. In a large sense, the learning process could be divided in to two categories: i) *supervised learning*, for every training instance is assigned a valid classification; ii) *unsupervised learning*, there are no training instances, and then the learning algorithms must find significant classifications [6].

In section 2 the control characteristics of **MASLCP** are presented. In section 3 the learning problem in **MASLCP** is pointed out. In section 4 is reflected the predictability in the proposed controlled process agent. Section 5 presents the predictability of the learning process in **MASLCP**. Section 6 presents the paper's conclusions and future directions to develop new learning properties of **MASLCP**.

2. The control characteristics of MASLCP

In our **MASLCP**, the learning process is supervised by a control expert system agent and the goal of this problem is that human agent can assimilate in a gradual way the planning knowledge so that he becomes, as far as possible autonomous in a restricted learning time $\mathbf{l}_T$. The **MASLCP** consists of: ***the controlled process agent*** (**CPA**) defined by a certain class of discrete event system, with a precisely goal; this represents the problem domain; ***the control expert system agent*** (**CESA**) of the plant and learning process, which includes more fuzzy knowledge models; ***the diagnosis agent*** (**DA**) used in the generation of plausible explanations. The **DA** activates a certain intern knowledge model of the **CPA** that will be used by the **CESA**. Based on the generated explanations, the observer learns the used knowledge model of the **CPA**. If these explanations are valid, then they represent the sum of knowledge that permit the observer advance in the learning process; t***he observer*** or the human agent (**HA**) [4].

It is assumed that the **CPA** can be represented with the following model: **CPA**=(X, E, f_e, δ_e, g, E_v), that can represent certain class of discrete event systems, where X is the set of **CPA** states denoted by x, E is the set of all events, f_e are the state transition map, f_e: $X \rightarrow X$, $e_k \in P(E)$, $k \in T$, δ_e are the output maps, g is the enable function, $g: X \rightarrow P(E)$, and E_v is the set of all valid event trajectories (that are physically possible). Note that E is the union of the command-input events (E_u), the disturbance input events (E_d) and the output events (E_o) of the plant. When discussing the states and events at time k, $k \in T$ or k is a fuzzy instant or a fuzzy time interval, $x_k \in X$ is the **CPA** state, $e_{uk} \in E_u$ is a command input event of the plant, $e_{dk} \in E_d$ is a disturbance input event of the plant, $e_{ok} \in E_o$ is an output event of the plant, that is equal to input event $e_{pk} \in E_p$ for **CESA**. Each $e_k \subset g(x_k)$ is an event that is enabled at time k, and it represents a set of command and disturbance input events of the plant. If an event $e_k \in E$ occurs at time k and the current state of **CPA** is x_k, then the next state is $x_{k+1} = f_{ek}(x_k)$ and the output is $e_{ok} = e_{pk} = \delta_{ek}(x_k)$. Any sequence $\{x_k\}$ such that for all k, $x_{k+1} = f_{ek}(x_k)$, where $e_k \subset g(x_k)$ is called a *state trajectory*. The **CESA** has two inputs: the reference input events $e_{rk} \in E^{CES,r}$ (user inputs) and the output events of the **CPA** $e_{ok} = e_{pk}$, $e_{rk} \in$

$E^{CES,p}$. Based on its fuzzy state and these inputs, the **CESA** generates enable command input events to the **CPA** $e_{0_k}^{CES} \in E^{CES,0}$. Hence the **CESA** models how the observer in the loop coordinate the use of feedback information from the **CPA**, reference and user inputs (modeling the current control fuzzy objectives), and information in its memory (the fuzzy **CESA** state). This inference loop constitutes the core of the **CESA** where the knowledge is interpreted by the inference engine, actions are taken, the fuzzy factbase is updated and the process repeats. Usually, the fuzzifier may transform the measured value (e_{pk}) of the sensor measurement into a corresponding universe of discourse for each input variable, as an input fuzzy fact. Fuzzy rules $R_i \in R$, are used to express knowledge. Three kinds of variables are used: input, output and intermediate variables. The defuzzification process decides for each output variable a single value. The **CESA** is modeled by: **CESA** = (X^{CES},E^{CES},$f^{CES,e}$,δ^{CES}, g^{CES}, $E^{CES,v}$), where $X^{CES} = X^b x X^{int}$ is a set of fuzzy **CESA** states $x^{CES,k}$ (X^b is the set of fuzzy factbase states and X^{int} is the set of possibilistic inference engine fuzzy states), E^{CES} is the set of events of the **CESA** (reference inputs $E^{PES,r}$ user inputs, output **CESA** events E_0^{CES}, the set of fuzzy rules R and the **CPA** output events E_p^{CES}), so that: $E^{CES} \subset P(E_p^{CES} \cup E_r^{CES} \cup UI \cup R \cup E_0^{CES} \cup HC)$, g^{CES} is the enable function, f_e^{CES}, $e_k \in P(E^{CES})-\{\varnothing\}$ are the state transition maps, δ^{CES} is the output map of **CESA** and E_v^{CES} is the set of all valid inference loop trajectories that are possible. It is assumed that an occurrence of a command input event to the **CESA**, $e^{CES} \in E^{CES}$ is always accompanied by a firing of enabled instantions rules $R_i \in R$, i=1,...,n, so that the fuzzy inference loop can be updated accordingly. Similarly, the firing process of fùzzy rules cannot be active alone, because the inference loop is updated only if there is a change in the **CPA** reflected via its outputs, or a change in the reference input event $e_r \in E_r^{CES}$, or user inputs. It can control the hypothesis/conclusions for the user decision or the enabling of the command input events of the **CPA** ($e_{ok}=e_{uk}$). The input events inclusion in the fuzzy knowledge model (**FKB**) allows the **CESA** designer to incorporate the **CPA** feedback and the reference input variables directly as parts of the **FKB**. This is analogous to the use of variables in conventional rule-based expert systems [3].

It is important to note here that the consequent formulas of the rules represent how the fuzzy state x^b in the fuzzy factbase changes, based on the occurrence of input events, and they can be defined in a recursive manner. We can define the conclusions on $X^b x E^{CES}$ or in $X^b x E^{CES} x X^{int}$ so that the fuzzy rules could characterize changes made to the inference strategy. An event $e_k=\{R_i \in E^{CES}\} \subset g^{CES}(x_k^{CES})$ can possibly occur if the full premise of R_i evaluates satisfactory at time k, for the given state $x_k^b \in X^b$ and the command input event e_{uk}. Then, after the event's occurrence, the next state $x_{k+1}^{CES} = f_{e_k}^{CES}(x_k^{CES})$ is given by the application of the conclusion (taking into account their time of truth) to the fuzzy state $x_k^b \in X^b$ to produce x_{k+1}^b and by updating the inference engine state x^{int}. In this way it is evident that fuzzy decision-making capabilities of the **CESA** are more sophisticated than those of the standard fuzzy control systems. The **CESA** has to be designed so that it can eliminate the undesirable closed-loop system behaviors. There is a need to specify the initial state of the closed-loop system to reduce the insignificant state combinations that may unnecessarily complicate the model.The operation of the **CESA**, at the inference level, proceeds by the following steps [10]: acquiring the CPA outputs and reference input events at time k; forming the conflict set in the fuzzy match phase from the compiled set of rules in the fuzzy knowledge-model M_{KF} based on e_{uk}, the current status of the truth of various fuzzy facts, and the current values of variables in the knowledge-base; using conflict resolution strategies (refraction, recency, distinctiveness, priority, and arbitrary) in the select phase, find one rule r' to fire; executing the actions characterized by the consequent of rule r' in the act phase.

Although every occurrence of an input event of the **CPA** always affects the **CESA**, the occurrence of an input event of the **CESA** does not necessarily immediately affect the **CPA** state. In qualitative analysis of our **CESA**, we focus especially on testing if the closed-loop **CESA** satisfy certain properties, as follows: reachability, cyclic properties and stability [9].

3. The learning process in MASLCP

There are several major paradigms, or approaches, to **ML**. These include supervised, unsupervised, and reinforcement learning. In addition, many researchers and application developers combine two or more of these learning approaches into one system. How the training data is processed is a major aspect of these learning applications [1]. The most important learning paradigms are [4]:

1. *Supervised learning* is sometimes called programming by example. The learning agent makes a prediction based on the inputs and if the

output differs from the desired output, then the agent is adjusted to produce the correct output. This process is repeated over and over until the agent learns to make accurate classification or predictions.

2. *Unsupervised learning* is used when the learning agent needs to recognize similarities between inputs or to identify features in the input data. The clustering or segmenting process continues until the agent places the same data into the same group on successive passes over the data. An unsupervised learning algorithm performs a type of feature detection where important common attributes in the data are extracted.

3. *Reinforcement learning* is a type of supervised learning used when explicit input/output pairs of training data are not available. This process of identifying the relationship between a series of input values and a later output value is called temporal credit assignment. Another important distinction in learning agents is whether the learning is done on-line or off-line.

4. *On-line learning* means that the agent is sent out to perform its tasks and that it can learn or adapt after each transaction is processed. On-line learning is like on-the-job training and places severe requirements on the learning algorithms. It must be very fast and very stable.

5. *Off-line learning* is more like a business seminar. After a suitable training period, it is possible to apply newfound knowledge and skills. In an intelligent agent context, this means that we would gather data from situations that the agents have experienced.

The intersection of **DAI** and **ML** constitutes an important area of research and application. The **DAI** and the **ML** communities largely ignored this area for a long time. There are two major reasons for this attention, both showing the importance of bringing **DAI** and **ML** together [4]: **i**) There is a strong need to equip multiagent systems with learning abilities, because these systems act in complex – large, dynamic, and unpredictable – environments. For such situations it is very difficult to correctly specify these systems a priori, that is, at the time of their design and prior to their use. **ii**) An extended view of multiagent learning reflects the insight that learning in multiagent systems is not just a magnification of learning in stand-alone systems, and not just the sum of isolated learning activities of several agents. It is useful to distinguish two principal categories of learning in multiagent systems: *centralized learning* (isolated learning); *decentralized learning* (interactive learning).

The two learning forms described above are of a rather nature, and they cover a broad variety of categories of learning that can occur in multiagent systems. Centralized and decentralized learning are best interpreted as two appearances of learning in multiagent systems that span a broad range of possible forms of learning. In the following we describe several differencing features of learning forms in multiagent systems, for structuring this variety.

1. *The degree of decentralization* (concerns distributedness and parallelism);
2. *Interaction-specific features*, required for realizing a descentralized learning process (e. g. planning, inference or decision steps, that are executed to achieve a particular learning goal);
3. *Involvement-specific features* (the relevance of involvement and the role played during involvement);
4. *Goal-specific features*;
5. *The learning method*;
6. *The learning feedback.*

These features characterize learning in multiagent systems from different points of view and at different levels. Agents having a limited access to relevant information run the risk of failing in solving a given learning task. This risk may be reduced by enabling the agents to explicitly exchange information, to communicate with each other. Generally, the following two forms of improving learning by communication may be distinguished:

1. *learning based on low-level communication*, that is, relatively simple query-and-answer interactions for the purpose of exchanging missing pieces of information (knowledge and belief);
2. *learning based on high-level communication*, that is, more complex communicative interactions like negotiations and mutual explanation for the purpose of combining and synthesizing pieces of information.

In our **MASLCP**, the learning process is supervised and the goal of this problem is that human agent **HA** can assimilate in a gradual way the fuzzy planning knowledge so that he becomes, as far as possible autonomous in a restricted time l_T (the learning time).

4. The predictability in CPA

An example for **CPA** is a fuzzy load balancing problem (**FLBP**) and it is described by a directed graph (C, A) where C={1,2,..., N} represents a set of machines that are numbered with $i \in C$, and $A \subset C \times C$ is the set of connections between them ({(1,2), (2,1), (1,3), (3,4), (4,3), (4,2), (3,5), (5,6), (6,5), (6,4)}). We require that if $i \in C$ then there exists (i,j)$\in$A or (j,i) $\in$A fore some j$\in$C (i.e., every machine is connected). Also, if (i,j)$\in$A and if (i,j) $\in$A i$\neq$j. Each machine has a buffer which hold load, given by x_i, $x_i \geq 0$. Each connection (i,j)$\in$A allows for machine i to pass a portion of its load to machine j. It also allows machine i to sense the size of the load of machine j (for any two machines i and j such that (i,j)$\notin$A, i may not pass load directly to j or sense the size of j's load). This problem appears also in the papers [9,11]. Below we consider first the discrete case when the load is in the form of fixed uniform-sized blocks that cannot be subdivided. In this case, the crisp control knowledge base contains twelve rules R_i, i=1,...,12, of the following type:

R_i: If (the charge of m1 >= the charge of m2) and (the charge of m1 >= the charge of m3) and (the charge of m1 >= the charge of m4) and (the charge of m1 >= the charge of m5) and the charge of m1 >= the charge of m6) and (xb[1] $\neq$0) and (xb[3] $\neq$ 0) and (the charge of m1 $\neq$ the charge of m2)

Then *in order conclude* that xb[0] = 1 and *inform* the operator and i*nfer* that "[the name of this rule], xb1 [xb[1]], xb2 [xb[2]], xb3 [xb[3]], xb4 [xb[4]], xb5 [xb[5]], xb6 [xb[6]], xb7 [xb[7]], xb8 [xb[8]], xb9 [xb[9]], xb10 [xb[10]]" and *start modify_charge* (xb[0]) and *conclude that* xb[2] = 1 and *conclude that* xb[4] = 1 and *conclude that* xb[6] = 1 and ?*conclude that* xb[8]=1 and *conclude that* xb[9] = 1 and *conclude that* xb[10] = 1 and *conclude that* xb[1] = 0 (this is in the G2 formalism).

In spite of its greater expressiveness, the present crisp model (knowledge base and the simulation results) for the discrete load balancing problem has several limitations: the load cannot be infinitely subdivided, so that not for any initial loads the problem has a good balancing, or acceptable. The **CESA** does not have as many ways to perform redistribution, so that only imperfect or inexact load balancing can be achieved. In conclusion, the embedding a metaknowledge was used, like fuzzy knowledge, represented in our formalism, so that the balancing problem will have good solutions in any initial load cases. This is similar with the continuous load-balancing problem, for which the qualitative analysis can be performed.

It is obvious that the open-loop **CPA** has cyclic properties that may prevent the open-loop from achieving the desired control objective. When closed-loop fuzzy expert control is used, as in our example, the invariant set exists, by simple analysis of the **CESA** dynamics. Using a search algorithm, we show that there exists at least one path from any given initial part distribution in the **FBLP**. The reachability result (the **FBLP** described above is reachable for all initial states, because there exists a sequence of events to occur that produces a state trajectory, so that the end state of the **CPA** is in the invariant set). In our fuzzy **CESA**, any rule whose "partially matches" the current data can "fire" (i.e., contribute to specifying the control input). In the fuzzy compiled knowledge model we consider here, there may be more than one rule whose antecedent "exactly matches" the current data, but our inference engine allows only one rule to fire at a time. The fuzzy pattern-matching aims to determine the instantiations set of the causes. It is stronger than classic one because of its capacity of processing the fuzzy knowledge. It is a matter of evaluating the degree of this pattern-matching between a fuzzy cause and a fuzzy fact.

At the end of the fuzzy condition/fact pattern-matching stage for the cause C and the fact F, if the degrees of the pattern-matching satisfy the chosen thresholds and if there is a consistent substitution σ, then pattern-matching is successful. The substitution σ is a particular case when the variables in the causes can be associated to some fuzzy constants present in the facts. The instance $\sigma \cdot C$ obtained through the application of the fuzzy substitution σ to the condition C is not totally equal with F, i.e. the expresion $F=\sigma \cdot C$ is not always true then σ is fuzzy. Knowing the significance of the four parameters Π, N, θ, K, we can take into account the problem of finding the proper thresholds of the measures Π and N in order to determine the facts that do not filter the causes at all. The choice is not made at random, as between the two parameters of GMP it must be a tight link. Because of all these remarks and in order to correctly solve the problem, there are the links between Π,N,θ,K.

The fuzzy condition/fact pattern-matching constitutes the first stage in the running of the inference engine which takes into account the imprecision. After this stage, it results a lot of instantiations of the causes. Each instantiation of reason will be associated to a fuzzy substitution and to the four parameters Π,N,θ,K. The fuzzy unification aims at verifying the consistence of the fuzzy substitutions where the variables can be associated to fuzzy sets.

It is interesting to note that the fuzzy binary relation R, can be interpreted in various ways. For example, the equality relation my be regarded as a particular case of relation R. A last important problem is the parameters propagation [5]. Figure 1 shows a test simulated example for the evolution of the distribution in the fuzzy LBP, based on

CESA functioning (NI=Inference Number, CS=Conflict Set, ER=Executed Fuzzy Rule).

IN	μ_{x1}	μ_{x2}	μ_{x3}	μ_{x4}	μ_{x5}	μ_{x46}	x_{b1}	x^{b1}	CS_k	ER	Event	ge
0	77.000	88.000	205.00	382.00	166.00	0.00	0	00000000000	6,7	7	e_{43}	229.00
1	77.000	88.000	293.50	293.50	166.00	0.00	7	00000010000	4,6	4	e_{35}	153.00
2	77.000	88.000	229.75	293.50	229.75	0.00	4	00010010000	6	6	e_{42}	153.00
3	77.000	190.75	229.75	190.75	229.75	0.00	6	00000100000	5,8	5	e_{34}	153.00
4	77.000	190.75	210.25	210.25	229.75	0.00	5	00001000000	8	8	e_{56}	153.00
5	77.000	190.75	210.25	210.25	114.88	114.88	8	00001001000	4,6,7	7	e_{43}	76.000
6	77.000	190.75	210.25	210.25	114.88	114.88	7	00001010000	4,6	4	e_{35}	76.000
7	77.000	190.75	162.56	210.25	162.56	114.88	4	00010010000	6	6	e_{42}	76.000
8	77.000	200.50	162.56	200.50	162.56	114.88	6	00000100000	3,7	7	e_{43}	76.000
9	77.000	200.50	181.53	181.53	162.56	114.88	7	00000010000	3	3	e_{21}	76.000
10	138.75	138.75	181.53	181.53	162.56	114.88	3	00100010000	4,6	4	e_{35}	38.125
11	138.75	138.75	172.05	181.53	172.05	114.88	4	00010010000	6	6	e_{42}	38.125
12	138.75	160.14	172.05	160.14	172.05	114.88	6	00000100000	5,8	5	e_{34}	38.125
13	138.75	190.75	166.09	166.09	172.05	114.88	5	00001000000	8	8	e_{56}	38.125
14	138.75	190.75	166.09	166.09	143.46	143.46	8	00001001000	4,6,7	7	e_{43}	14.250
15	138.75	190.75	166.09	166.09	143.46	143.46	7	00001010000	4,6	4	e_{35}	14.250
16	138.75	190.75	154.78	166.09	154.78	143.46	4	00010010000	6	6	e_{42}	14.250
17	138.75	163.12	154.78	163.12	154.78	143.46	6	00000100000	3,7	7	e_{43}	14.250
18	138.75	163.12	158.95	158.95	154.78	143.46	7	00000010000	3	3	e_{21}	14.250
19	150.93	150.93	158.95	158.95	154.78	143.46	3	00100010000	4,6	4	e_{35}	9.5391
20	150.93	150.93	156.86	158.95	156.86	143.46	4	00010010000	6	6	e_{42}	9.5391
21	150.93	154.94	156.86	154.94	156.86	143.46	6	00000100000	5,8	5	e_{34}	9.5391
22	150.93	150.93	155.90	155.90	156.86	143.46	5	00001000000	7,8	7	e_{43}	9.5391
23	150.93	150.93	155.90	155.90	156.88	143.46	7	00001010000	8	8	e_{56}	9.5391
24	150.93	150.93	155.90	155.90	150.16	150.16	8	00001011000	3,4	3	e_{21}	2.9014
25	152.94	152.94	155.90	155.90	150.16	150.16	3	00101010000	4,6	4	e_{35}	2.9014
26	152.94	152.94	153.03	155.90	153.03	150.16	4	00010010000	6	6	e_{42}	2.9014
27	152.94	154.42	153.23	154.42	153.03	150.16	6	00000100000	3,7	7	e_{43}	2.8384
28	152.94	154.42	153.73	153.73	153.03	150.16	7	00000010000	3	3	e_{21}	2.8384
29	153.68	153.68	153.23	153.73	153.03	150.16	3	00100010000	8	8	e_{56}	2.8384
30	153.68	153.68	153.23	153.73	151.60	151.60	8	00000011000		0	$e \neq e_{00}$	1.4034

5. Predictability of the MASLCP learning process

In the following, we will designate the learning process the synthesizing of structured knowledge models $M_0,\ldots,M_k$, specific to a fuzzy control strategy based on the observation of certain cases for **CPA** and **CESA**, that means *supervised learning*. The existence of a number of fuzzy knowledge models: $M_0 \subset M_1 \subset \ldots \subset M_k$, means a gradual and incremental learning process. The important elements of parsimonious covering theory who represents the conceptual base of the **DA**, are according [5]. The **DA** acts as a supervisor in a learning **HA** process.

Definition 1. *A diagnostic problem* P is a 4-tuple $<D,M,C,M^+>$ where:

1. $D=\{d_1,d_2,\ldots,d_n\}$ is a finite, non-empty set of objects, called *disorders*;
2. $M=\{m_1,m_2,\ldots,m_k\}$ is a finite, non-empty set of objects, called *manifestations*;
3. $C \subseteq DXM$ is a *relation* with domain(C)=D and range(C)=M, called causation;
4. $M^+ \subseteq M$ is a distinguished subset of M which is said to be present.

Definition 2. The set $D_I \subseteq D$ is said to be a *cover* of $M_J \subseteq M$ if $M_J \subseteq$ effects(D_I).

Definition 3. A set $E \subseteq D$ is said to be an *explanation* of M^+ for a problem $P=<D,C,M,M^+>$ if and only if E covers M^+ and E satisfies the given parsimonious criterion. An explanation consists of three conditions:

1. the covering requirement (every manifestation in M^+ must be associated with some of E's members);
2. the covering must be parsimonious;
3. the explanation must consist of disorders only.

Definition 4. Let $g_1, g_2, \ldots, g_n$ be non-empty pair wise-disjoint subsets of D. Then $G=\{g_1,g_2,\ldots,g_n\}$ is a *generator*. The class generated by GI, designated as [GI],is defined to be $[GI]=\{\{d_1,d_2,\ldots,d_n\}|d_i \in g_i,\ 1 \leq i \leq n\}$. The generator-set operations are: *division* , *residual* of a division , *augmented residual* of a division.

Suppose that at some point during problem-solving, a set of manifestations M_1 are known to be present and generator-set G^1 represents a tentative solution (all explanation of M_1). If an additional manifestation $m_j \notin M_1$ is discovered, then manifestation $M_2=M_1 \cup \{m_j\}$ are known to be present. Then the division ***div**(G^1,causes(m_j))* results in a generator-set representing all explanations in $[G^1]$ that also cover m_j and hence M_2. Also, ***res**(G^1,causes(m_j))* is a generator set representing all explanations in $[G^1]$ that do not cover m_j. By adding appropriate elements of *causes(m_j)* to each set in *res(G^1,causes(m_j))* to form an augmented residual, ***augres(G^1, causes(m_j))*** we can convert each explanation for M_1 in $[G^1]$ that does not cover M_2 into a cover of M_2. The set of explanations represented by *div(G^1, causes(m_j))* plus the covers represented by *augres(G^1,causes(m_j))* come close to representing a revised solution or hypothesis for M_2. These operations define how to revise incrementally the existing generator-set when a new manifestation is discovered during sequential problem-solving. The **BIPARTIT** algorithm works in a sequential and constructive manner. It takes one present manifestation m_j at a time, either from M^+ or through an interactive question-answering process, and than incorporates *causes(m_j)* into the existing hypotheses. This continues until all present manifestations are processed. Algorithm **BIPARTIT** represents tentative hypotheses (explanations) and the final solution in the generator-set form, and is based on the operations of generator division, residual, and augmented residual. A function called "revise" is defined to construct new hypotheses from the existing hypotheses using disorders evoked by a newly arrived manifestation: revise(G,H_1)= $F \cup$res(Q,F), where: G is the previous generator-set; H_1= causes(m_j); F=div(G,H_1); Q=augres(G,H_1); res(Q,F) is used to remove all duplicate and redundant covers from [Q].

To capture by the **HA** the control knowledge for solving the **FBLP**, the **CESA** uses at any time, an internal knowledge model of the **CPA** based on the level of the last knowledge model. The output of the **CPA** is compared with the reference (goal) and, if this output doesn't satisfy the required criteria, it will represent a fuzzy qualitative error (i.e. a set of manifestations). This k+1 qualitative error represents the unique activated inputs in the diagnosis system **DA**, having the characteristics of a dynamic system **CPA**.

We have developed **MASLCP** structure, and the actual significance is according [4]:

1. ε_i, $i=0,\ldots,k$ - the calculated errors series in relation with the objective control function. The calculation is realized by identification all present manifestations in the **CPA** behavior, using the current knowledge model of **CPA**. The **HA** learns each of these knowledge control models, embedded in the **CES**. In short, ε_i represents the non-empty set of all manifestations which allow the **DA** to be activated for choosing the new control knowledge model.
2. The **DA** has as inputs a set of manifestations and the outputs represent explanations of the existing manifestations (through of the generating reasoning hypotheses). The **HA** takes over the results of the **DA** (that represents the level of perception for the control problem solving), abstracts these results (classifies previous knowledge obtained by knowledge acquisition) and can select a certain fuzzy knowledge model.
3. If there is i_0, $i_0 \in \{0, \ldots, k\}$ so that $\varepsilon_{i0} = 0$, then the learning process ends and the **HA** is considered trained at the level M_{i0}. The learning process is predictable. For each $s \geq k$, $M_k=M_s$ (that means that the maximum level of knowledge is achieved at l_T time). The last two remarks imply the finitude property of the learning process, using a given period of time l_T.

If for any $0 \leq i < k$ the **HA** (the learner) have used and tested the right model M_i, then the choice of the new knowledge model M_{i+1} can be made by the necessary condition $M_{i+1} \supset M_i$, certainly. This means that any following fuzzy knowledge model must represent a reached knowledge control model. The learning process used in our **MASLCP** shows that the relationships and the analogy between expert and control system architectures are important problems for intelligent control. This is possible because both are problem solving systems with different problem domain (environment) the **MASLCP** reasons about and takes actions on.

6. Conclusions

The predictability of our **MASLCP**, from the practical point of view, simulates only the diagnosis component but include knowledge models of the considered planning problem in different stages of its development. The diagnosis model involve diagnostic entities (disorders, manifestations), causal associations relating these entities (the causal network), the notion of diagnostic explanation and the process of hypothesize reasoning. The algorithm works in a sequential and constructive manner. It takes one present manifestation for each time and than incorporates its causes into the existing hypotheses. The process continues until all present manifestations are processed and the learning time is less or equal with $\mathbf{l_T}$. The **DA** accept as inputs a set of manifestations and supply outputs that represents explanation in the presence of the manifestation.

Considering the given **MASLCP** structure, the next conclusions can be pointed out: the **DA** works in a diagnostician manner (sequential and constructive); his learning process is supervised and the goal of this problem is in short the human agent **HA** can assimilate the planning knowledge so that he became, as far as possible autonomous [1]; a limitation is the dynamically loop closing, according to the conceptual structure of the **MASLCP**, that is carried out by the **HA** who learn.

There are important another future directions for this work, investigating the dynamics of AI reasoning systems that utilize learning and planning capacities in various complex applications, studying the semantic knowledge representations on Web.

References

1. **Antsaklis P.J., Lemmon M., Stiver J.A.** (1996), *Learning to Be Autonomous – Intelligent Supervisory Control*, in "Intelligent Control Systems – Theory and Applications" by Gupta M.M., Sinha N.K., IEEE Press;
2. **Farrell J., Baker W.** (1996), *Learning Control Systems – Motivation and Implementation*, in ***Intelligent Control Systems. Theory and Applications*** by Gupta M.M., Sinha N.K., IEEE Press;
3. **Mazilescu V.** (2001), *The Management of Fuzzy Knowledge in Planning Systems*, The Fifth International Symposium of Economic Informatics, Bucharest 10-13 May, p. 967-977;
4. **Mazilescu V., Căpriță D**., 2003 – ***Sistem de învățare bazat pe un model hibrid de cunoştințe***, Revista de Informatică Economică, Volumul VII, Nr3 , p. 23 - 29
5. **Peng Y., Reggia A.** (1990), *Abductive Inference Models for Diagnostic Problem Solving*, Springer–Verlag;
6. **Sandip S., Weiss G.** (2001), *Learning in Multiagent Systems*, Multiagent Systems, ISBN 0-262-23203-0
7. **Zadeh, L.A.** (1983), *The role of fuzzy logic in the management of uncertainty in expert systems*. Fuzzy Sets and Systems, n°. 11, p. 199-227