Hybrid Behaviour Orchestration in a Multilayered Cognitive Architecture using an Evolutionary Approach

Óscar Javier Romero López, Angélica de Antonio

Abstract— **Managing and arbitrating behaviours, processes and components in multilayered cognitive architectures when a huge amount of environmental variables are changing continuously with increasing complexity, ensue in a very comprehensive task. The presented framework proposes an hybrid cognitive architecture that relies on subsumption theory and includes some important extensions. These extensions can be condensed in inclusion of learning capabilities through bio-inspired reinforcement machine learning systems, an evolutionary mechanism based on gene expression programming to self-configure the behaviour arbitration between layers, a co-evolutionary mechanism to evolve behaviour repertories in a parallel fashion and finally, an aggregation mechanism to combine the learning algorithms outputs to improve the learning quality and increase the robustness and fault tolerance ability of the cognitive agent. The proposed architecture was proved in an animat environment using a multi-agent platform where several learning capabilities and emergent properties for self-configuring internal agent's architecture arise.**

I. Introduction

Recently, cognitive architectures have been an area of study that collects disciplines as artificial intelligence, cognitive science, psychology and more, to determine necessary, sufficient and optimal distribution of resources for the development of agents exhibiting emergent intelligence. One of the most referenced is the Subsumption Architecture proposed by Brooks [1].

According to Brooks [1], the Subsumption Architecture is built in layers. Each layer gives the system a set of pre-wired behaviours, where the higher levels build upon the lower levels to create more complex behaviours: The behaviour of the system as a whole is the result of many interacting simple behaviours. Another characteristic is its lack of a world model, which means that its responses are always and only reflexive as proposed by Brooks.

However, Subsumption Architecture results in a tight coupling of perception and action, producing high reactivity. Poor adaptability to new environments, no learning capabilities, no internal representation and the need of all patterns of behaviours must be pre-wired, are some weaknesses of the Subsumption theory.

Several extensions have attempted to add representation and behaviour arbitration to Subsumption like Behavior-Based Control Architecture [2] and Hormonal Activation Systems [3], but pre-wired behaviours and non-learning characteristics still remain becoming the architecture applicable and restricted only for a specific pre-configured environments.

The present research focuses on developing an hybrid multilayered architecture for cognitive agents based on subsumption theory. Additionally this work proposes an Evolutionary Model which allows the Agent to self-configure and evolve its arbitration of processing layers through the definition of processes (like *inhibition*, *suppression* and *aggregation*), behaviours and number of layers. That means each agent instead of having a pre-configured structure of layers and processes it will have an artificial evolutionary process which is responsible for defining the multilayered structure. On the other hand, instead of using an Augmented Finite Machine System (AFSM) as subsumption theory states in [3] where no internal representation is done, in this paper we propose that each behaviour layer is driven by a different Reinforcement Machine Learning System (RMLS), which is chosen from a repertoire where behaviour co-evolution occurs. Each RMLS learns from the environment and generates an internal world-model by means of an unsupervised and reinforced learning. The RMLSs used in the approach are: Extended Classifier System XCS [5], Learning Classifier System LCS [6], Artificial Immune System AIS [7], [8], and Connectionist Q-Learning System CQL [9], [10].

The remainder of the paper is organized as follows. The description of the proposed approach is detailed in Section 2. Section 3 outlines and discusses the experimental results and emergent properties obtained. Finally concluding remarks are shown in Section 4.

II. Proposed Hybrid, Self-configurable and Evolutionary Model

In order to design an hybrid, self-configurable, scalable, and evolutionary architecture for cognitive systems which exhibits emergent behaviours and learning capabilities, the proposed work is exposed as follows.

Consider a virtual environment where there are several cognitive agents interacting with each others using a typical Subsumption Architecture. Some mayor constraints arise:

- Environmental conditions changing continuously.
- The number of behaviour layers inside each agent is variable.

Manuscript received November 30, 2007. This work was supported by the Programme Alban, the European Union Programme of High Level Scholarships for Latin America and Politécnica de Madrid University.

Óscar J. Romero is with the Software Engineering Department, Universidad Politécnica de Madrid, Spain (e-mail: ojrlopez@hotmail.com).

Angélica de Antonio is with the Software Engineering Department, Universidad Politécnica de Madrid, Spain (e-mail: angelica@fi.upm.es).

978-1-4244-1823-7/08/$25.00©2008 IEEE

- Arbitration of behaviours is pre-wired and not depends on agent's motivational states.
- The cognitive agent can inhibit or suppress behaviours "only" if an applicability predicate is pre-established and new environment changes are not considered.
- Agent's behaviours do not generate a model of the world, do not couple with the environment via the agent's sensors and actuators, do not learn about its own interaction with the environment and do not evolve the internal state of the behaviour.

These constraints address the following proposed approach of an hybrid, self-configurable and bio-inspired architecture for cognitive agents, depicted in Fig. 1:

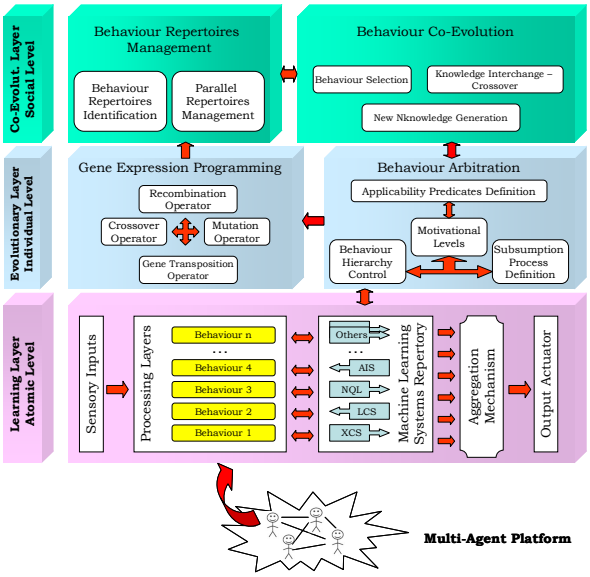


Fig. 1. Hybrid and evolutionary architecture for cognitive agents

The Fig. 1 shows an hybrid architecture from which all the constraints mentioned before can be solved. An internal architecture based on subsumption principles but with few variations can be observed in every agent:

- Each processing layer is connected with a different RMLS (XCS, LCS, AIS, CQL, and scalable to others) in a random fashion (due to all RMLSs have the same interface and purpose, no matter which RMLS connects with each processing layer). This approach replaces the typical Augmented Finite State Machines (AFSM) proposed by Brook's architecture in [1].
- After being trained, each agent's behaviour is sent to a behaviour repertoire according to its type, where a co-evolutionary mechanism is applied so, every behaviour not only will learn in a local way inside of each agent but also will evolve in a global way, to be selected afterwards by another agent in the next generation.
- There is an evolutionary process driven by a Gene Expression Programming algorithm (GEP) [12], which is in charge of self-configuring the agent (defining the number of layers, the behaviours that agent will use, the connections and hierarchies between them -inhibit, suppress, aggregate-, the applicability predicates that determine which behaviour is activated at a certain situation and an activation time controlled by a timer).

A. Hybrid Learning Layer: Behaviours driven by different Reinforcement Machine Learning Systems

Every behaviour layer in the multilayered architecture will be associated to one RMLS, that allows the architecture being hybrid and not only reactive since each behaviour will be able to exert deliberative processes using the acquired knowledge. Besides, this mechanism gives plasticity to the architecture because every behaviour "learns" in an unsupervised, independent and parallel way through its interaction with the environment, generating internal representations, rules and both specific and generalized knowledge. This mechanism is favored by the RMLS`s characteristics: robustness, fault tolerance, use of bio-inspired techniques, adaptability and they do not require a previous definition of knowledge (unsupervised learning).

There are two principles formulated by Stone [13] that have motivated the proposed layered learning approach:

- Layered learning is designed for domains that are too complex for learning a mapping directly from an agent's sensory inputs to its actuator outputs. Instead the layered learning approach consists of breaking a problem down into several behavioral layers and using RMLSs at each level. Layered learning uses a bottom up incremental approach to hierarchical task decomposition.
- RMLS is used as a central part of layered learning to exploit data in order to train and or adapt the overall system. RMLS is useful for training behaviors that are difficult to fine-tune manually.

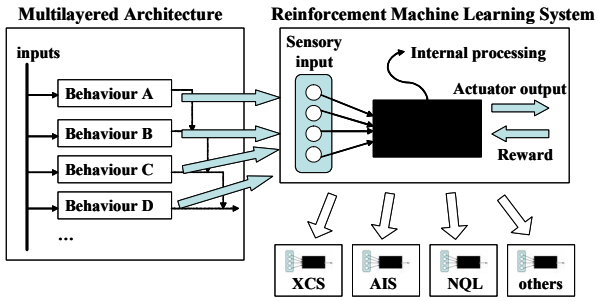


Fig. 2. Multilayered Architecture connecting with RMLS interface

The sensory inputs of each RMLS read the objects sensed around the agent while the actuator outputs indicate actions that the agent must to execute on the environment.

Accordingly, a common interface for all RMLSs (XCS, AIS, CQL, LCS, etc.) is proposed so, although each RMLS has a different internal process, they all have a similar structure that it lets the system to be scalable introducing new RMLSs if is required and connecting them in an easy way with each behaviour layer in the agent's multilayered

architecture, as depicted in Fig. 2.

Each RMLS has its advantages and disadvantages. However, no one RMLS is always better than others, so it is difficult to determine a good RMLS to drive every behaviour. On the other hand, the cognitive agent determines which behaviors must be inhibited or suppressed in a specific situation, but not always just one behaviour remains activated after inhibition, sometimes the Agent can require several behaviours to be activated concurrently. In order to merge the outputs of these behaviours activated in just one output, we propose an Aggregation Mechanism similar to proposed by Jiang in [11] but using different RMLSs. This mechanism is based on Borda Counting Method [15]. Aggregation of RMLSs can improve the learning qualities as a whole because they can share some knowledge and utilize the strengths of the others to alleviate individual weaknesses.

B. Evolutionary Layer: Behaviour Arbitration

If each agent has an arbitrary behaviour set, how to determine: the interaction between them, the hierarchy levels, the subsumption process (inhibition and suppression) and the necessary layers to do an adequate processing? These questions are solved next.

The internal multilayered structure of each agent is decomposed in atomic components which can be estimated and used to find the optimal organization of behaviors during the agent's lifetime [4]. The main goal is that the agent in an automatic way self-configures its own behaviours structure. The model proposed by Ferreira in [12] called GEP is used to evolve internal structures of each agent and generate a valid arbitration of behaviours.

GEP uses two sets: a function set and a terminal set. Our proposed function set is: AND, OR, NOT, IFMATCH, INHIBIT, SUPRESS and AGGREGATE. The AND, OR and NOT functions are logic operators used to group and exclude subsets of elements. The conditional function IFMATCH is an applicability predicate that matches with a specific problem situation. This function has five arguments; the first four arguments belong to the rule's antecedent: they all indicate motivational levels in the agent (internal states, moods, etc.), for instance: energy level, bravery/cowardice level, hunger/thirstiness level, etc. If the first four arguments are applicable then the fifth argument, the rule's consequent, is executed. The fifth argument should be an INHIBIT/ SUPPRESS/AGGREGATE function, or maybe an AND/OR function if more elements are necessary. The INHIBIT, SUPPRESS and AGGREGATE functions have two arguments (behaviourA, behaviourB) and indicate that behaviourA *inhibits/suppresses/aggregate* behaviourB.

On the other hand, the terminal set is composed by the behaviour set and the motivational levels set. Additionally "do not care" elements are included so whichever behaviour or motivational levels can be referenced. Behaviour arbitration is driven by agent's motivational levels which try to simulate moods or humor states in the cognitive agent. These moods are changing continuously whereas the agent interacts with the environment.

Each agent has a chromosome with information about its self structure, e.g. Agent A can have a chromosome as: *[{IFMATCH}, {ml$_1$}, {ml$_2$}, {ml$_3$}, {ml$_4$}, {INHIBIT}, {behaviour$_1$}, {AND}, {behaviour$_2$}, {behaviour$_3$}]*, and this chromosome is a valid rule because both the antecedent and the consequent of IFMATCH function match to each required argument type, where *{ml}* is the abbreviation for motivational level. The above chromosome traduces in the following rule:

IFMATCH:
m1$_1$, ml$_2$, ml$_3$, ml$_4$
THEN:
behaviour$_1$ INHIBIT behaviour$_2$ AND behaviour$_3$.

Analyzing this rule we can infer that the agent has three behaviour layers: *behaviour$_1$, behaviour$_2$,* and *behaviour$_3$*, and the two last ones are inhibited by the first one when agent has the motivational levels *m1$_1$, ml$_2$, ml$_3$, ml$_4$*. However, these chromosomes (applicability predicates) do not have always a valid syntax, so the GEP mechanism is used to evolve the chromosome until it becomes in a valid syntactic rule.

Each individual (agent) has a multigenic chromosome, that means, each chromosome has a gene set where each gene is an applicability predicate like the example, so the agent has several rules (genes) as part of its genotype and each one is applied according to the situation that matching the rule antecedent. Each gene is become to a tree representation and then a genetic operator set is applied between genes of the same agent and genes of other agents as in [12]: selection, mutation, root transposition, gene transposition, two-point recombination and gene recombination, in order to evolve chromosomal information. Fig. 3 shows how GEP Fixes chromosomes with invalid syntax using genetic operators.

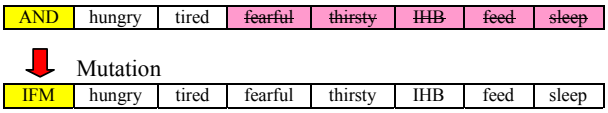


Fig. 3. Invalid syntax chromosome fixed by GEP

After certain number of evolutionary generations, valid and better adapted agent's configurations are generated. A roulette-wheel method is used to select individuals with most selection probability derived from its own fitness. Fitness represents how good interaction with environment during agent's lifetime was.

C. Behaviour Co-evolution Layer: evolving globally

A co-evolutionary mechanism is proposed to evolve each type of behavior separately in its own genetic pool. Most evolutionary approaches use a single population where evolution is performed; instead, the behaviours are discriminated in categories and make them evolve in separate behaviour pools without any interaction, as proposed in [14].

First, each agent defines a specific set of behaviours that builds its own multilayered structure. For each required agent's behaviour, a behaviour instance is chosen from the pool (this instance is connected with one RMLS). Subsequently each agent will interact with the environment and each agent's behaviour will learn a set of rules and generate an own knowledge base.

After certain period of time a co-evolutionary mechanism is activated. For each behaviour pool is applied a probabilistic selection method of behaviours where those behaviours that had the best performance (fitness) will have more probability to reproduce. Then, a crossover genetic operator is applied between each pair of selected behaviours: a portion of knowledge acquired by each agent's behaviour (through its RMLS) is selected and interchanged with the other one.

Finally, new random rules are generated until complete the maximum size of rules that behaviours can have in their own knowledge base, so a new pair of behaviors is created and left in the corresponding behaviour pool to be selected by an agent in the next generation.

D. *Emergent Properties of the Architecture*

Brooks postulates in his paper [3] the possibility that intelligence can emerge out of a set of simple, loosely coupled behaviours, and emergent properties arise (if at all) due to the complex dynamics of interactions among the simple behaviours and that this emergence is to a large extent accidental.

The proposed architecture articulates a behaviour set that learns about environmental conditions in an independent and parallel fashion, and on the other hand evolve inside a categorized pool. Each simple behavior can be applied to a subset of specific situations but not to the whole problem space, however the individual level interaction between behaviours (inside each agent) allows covering multiple subsets of problem states and some characteristics are generated: robustness, redundancy in acquired knowledge, fault tolerance and a big plasticity level, so emergent properties in the individual and inside of the society (multi-agent system) appear. So, the emergent properties arise from three points of view in a bottom-up approach:

- Atomic Level: in each behaviour of the multilayered architecture, when the associated RMLS learns from the environment how to associate sensory inputs and actuator outputs, in an automate way.
- Individual Level: when the agent self-configures its internal structure (chromosome), hierarchy and arbitration of behaviours through an evolutionary process driven by GEP.
- Social Level: when an hybrid behaviour co-evolution mechanism is applied to all agent's behaviours, so behaviours learn not only themselves via the RMLS associated but also cooperating with other agents and communicating the acquired knowledge between them.

It is important to notice that emergence in different levels, from atomic to social point of view, provokes an overall emergence of the system, where some kind of intelligence we hope to arise. The experimentation focused on discovering some emergent characteristics in the agents. Nevertheless, expected emergent properties can vary according to the environment and the behaviour set.

III. EXPERIMENTATION

In order to evaluate the proposed architecture, following aspects were considered in each level:

- Learning convergence rate of each proposed systems: XCS, AIS, LCS and CQL.
- Learning and evolution convergence rate of each behaviour pool.
- Variation of success rate vs. number of genes in GEP
- Syntactically well-formed gene convergence rate

About overall System:

- Subsumption architectures obtained on individuals after n iterations and emergent properties identified.

An artificial life environment called *Animat* (animal + robot) described in [6] is proposed to test the experiments. The environment simulates virtual agents (prey-depredator model) competing for getting food and water, avoiding obstacles, hunting, escaping from depredators, etc. Each animat driven by an agent in the environment disposes a set of 10 proximity sensors (see Fig. 4) simulating a limited sight sense. 8 sensors read a safe zone and 2 sensors read a danger zone (to avoid collisions) as proposed by Romero [8].

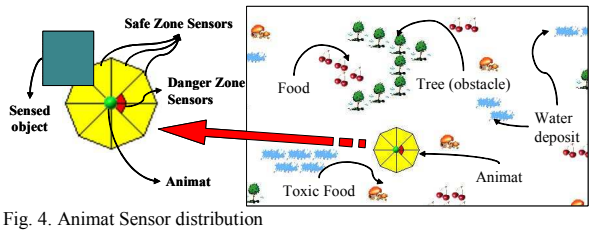


Fig. 4. Animat Sensor distribution

Some environmental changes were introduced during the experimentation, like new elements: initially environment only was inhabited by preys, and then depredators were introduced. Toxic food was introduced after animats had learned to eat food, objects were relocated, etc.

Thus, some experiments designed to evaluate the performance aspects mentioned above are described next.

A. *Learning convergence of each RMLS*

In this experiment we chose an environment where the animat has to interact with using one different RMLS on a time. Table I shows the learning parameters used.

TABLE I
LEARNING PARAMETERS OF EACH RMLS

Parameter	XCS	AIS	CQL	LCS
Life Tax	-	0.005	-	0.005
Bid Tax	-	0.003	-	0.003
Cloning Rate x rule	1	4	-	1
Mutation Rate x rule	2	2	-	1

Similarity Threshold	-	0.8	-	-
Alpha α	0.1	-	0.1	-
Beta β	0.2	-	-	-
Delta δ	0.1	-	0.02	-
Gamma γ	0.95	-	0.8	-
Lamda λ	-	-	0.8	-
Layers in NN	-	-	5	-
Number of Epochs		50		
Nº runs x epoch		20		

Fig. 5 shows a chart of the learning curve of the RMLSs: XCS, AIS, LCS, simple CQL and multilayered CQL. Each RMLS was executed 50 epochs, and in each epoch every RMLS reach a certain number of learning iterations.

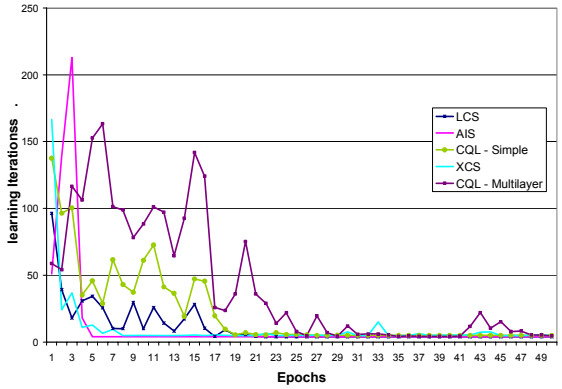


Fig. 5. Learning Curve of each RMLS

Fig. 5 illustrates that AIS and XCS are more adaptive and robust than the others converging more quickly when changes in the learned environmental pattern are introduced. The peaks were registered because of changing patterns, but each RMLS adapted to new conditions in the environment quickly.

B. Learning and evolution convergence of each behaviour pool.

The goal of this experiment is to examine if the fitness of every separate behaviour pool increments gradually until reaches a convergence point while evolution takes place. The experiment was carried out with the parameters on Table II.

TABLE II
CO-EVOLUTION LEARNING PARAMETERS

Parameters	Value
Epochs	50
Nº runs x epoch	50
Crossover Prob.	0.7
Mutation Prob.	0.3
Mutation Rate η	0.85
Mutation Rate θ	0.25
Mutation Rate κ	1.03
Mutation Rate γ	0.01

Three behaviour pools were selected for the experiment: *Avoiding-obstacles, Looking-for-food* and *Escaping-From-Depredators*, the results are depicted in Fig. 6.

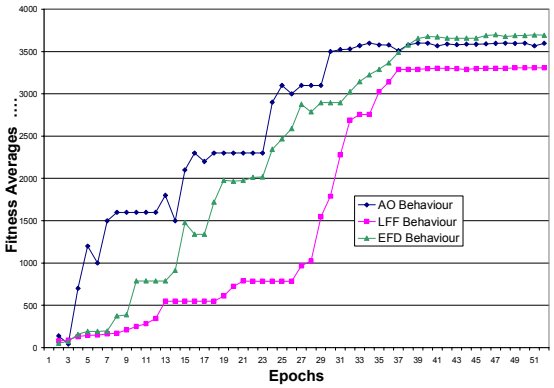


Fig. 6. Evolution convergence rate in 3 behaviour pools

Fig. 6 depicted some differences in each learning curve, due to environmental conditions, however the pools always tried to converge and reach certain knowledge stability in the same number of epochs (approximately after 30 epochs), that means the evolution has been effective and each behaviour pool has established a coherent knowledge base getting a consensus between its own behaviour instances, about what the "behaviour category" should do.

C. Syntactically well-formed gene convergence

In this experiment, the progression of the number of syntactically well-formed structure (multigenic chromosomes) of each individual was analyzed. Fig. 7 shows how the number of valid chromosomes increments whereas generations evolve through the time. The experiment was executed with a population of 300 individuals.

Fig. 7 shows that a convergence point (that means all chromosomes in population are valid) is given in the generation 27 approximately. Then, the system will need between 25 and 30 generations to evolve all individuals in the population.

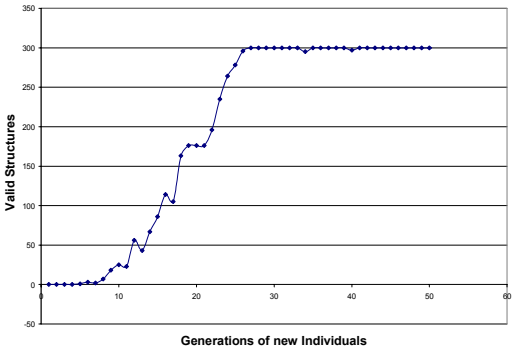


Fig. 7. Valid Structures (chromosomes) through several Generations

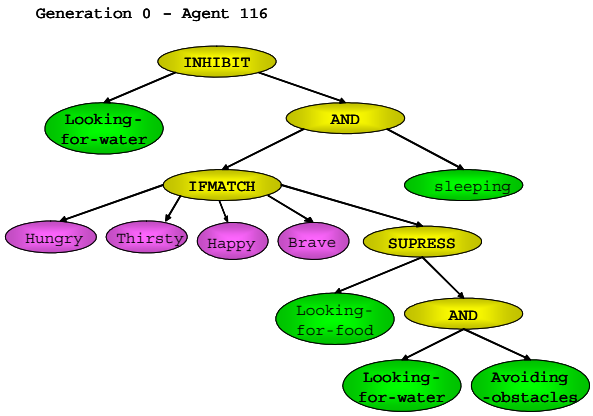
D. Analysis of evolved architectures

Finally, after the whole system has evolved during a specific number of generations, we have analyzed the final

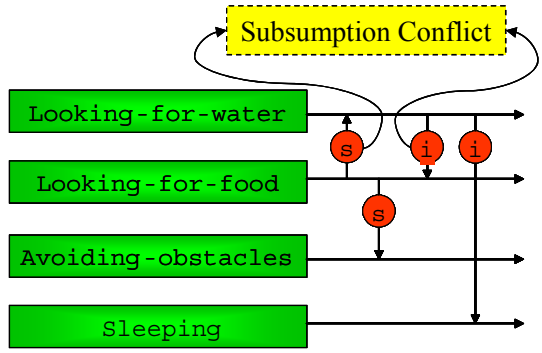
structures of the best adapted agents where emergent properties arose.

Fig. 8 shows the genotype (Expression Trees ETs) and phenotype respectively of an initial architecture of a random agent without any evolutionary phase; in contrast, Fig. 9 shows the genotype and phenotype respectively of the evolved architecture of the same agent.

In Fig. 8 the chromosome represents four behaviours: looking-for-water LFW, looking-for-food LFF, avoiding-obstacles AO and sleeping SL, where LFW inhibits LFF and SL, and LFF suppresses AO, but there is a contradictory process when LFF tries to suppress LFW and LFF has been inhibited by LFW already. This is solved with the evolved architecture in Fig. 9, which proposes a new structure adding escaping-from-depredators EFD behaviour and excluding SL behaviour.



a) Genotype of invalid chromosome

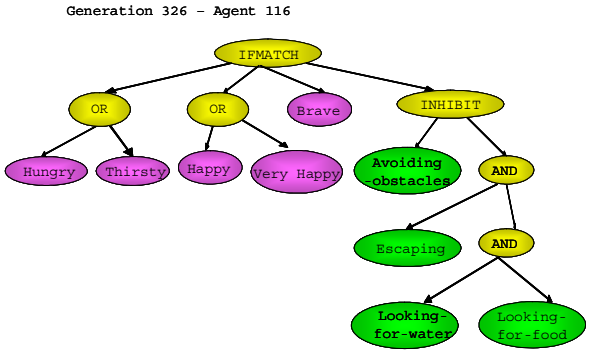


b) Phenotype of invalid chromosome

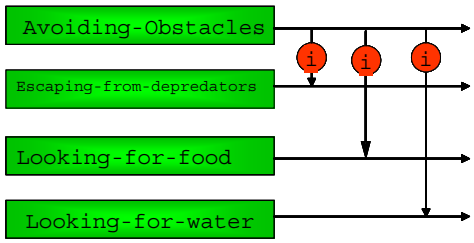
Fig. 8. Genotype and Phenotype of an initial Agent's Architecture

As depicted in Fig. 9, the initial contradictory inhibitory/suppressor processes in the agent's architecture are solved, and only hierarchical inhibitory processes are proposed by the evolved architecture. Furthermore, we can deduce too that evolved architecture has collected a set of specific behaviours becoming the agent to an animat with a prey identity.

It is important to notice in evolved architecture that EFD behaviour inhibits both LFF and LFW behaviours, but if the animat is escaping and its sensors read a "wall" or a "tree", then EFD behaviour is inhibited by AO behaviour until the obstacle is not in front of the animat anymore, and after that the animat continues its getaway, so we can say that emergent behaviour arises.



a) Genotype of valid chromosome



b) Phenotype of valid chromosome

Fig. 9. Genotype and Phenotype of the Agent's Architecture after 326 evolutionary generations

Finally, the experimentation demonstrates that specific parameter configurations in RMLSs, GEP and Co-evolutionary mechanism are required to reach certain robustness, adaptability and learning capacities in the overall system. Nevertheless, emergent properties did not arise always or in a fast way, in several experiments animats died quickly and they could not learn to survive, but this allows appearing more robust individuals later.

IV. Conclusions

The integration of multiple RMLS in controlling the behaviours layers of an hybrid Subsumption Architecture approach instead of using the typical Augmented Finite State Machines, have demonstrated through using bio-inspired techniques, important advantages in learning about the agent`s world (matching sensory inputs and actuator outputs), making internal knowledge representations (inside chromosomes, classifiers, antibodies, neurons, etc.) and adapting to environmental changes (evolutionary

mechanisms).

The evolutionary mechanisms used in this work, provided a plasticity feature allowing the agent to self-configure its own multilayered behaviour-based architecture; thus it can avoid creating exhaustive and extensive knowledge bases, pre-wired behaviour-based multilayered structures and pre-constrained environments. Instead of this, a cognitive agent using our architecture only needs to interact with an arbitrary environment to adapt to it and take decisions in a reactive and deliberative fashion.

In the experimentation, the emergent properties were difficult to discover because it takes a lot of time to evolve the overall system despite of using a multi-agent platform in a distributed configuration. Maybe, it can be similar to the natural evolution where adaptation occurs slowly and sometimes produces poor adapted creatures.

In our future work we expect to continue working on designing more adaptive and self-configurable architectures, using fuzzy techniques in the RMLSs to improve the sensors readings and to manipulate motivational levels (moods). One concrete application of this research will be the development of a cognitive module for Emotive Pedagogical Agents where the agent will be able to self-learn of perspectives, believes, desires, intentions, emotions and perceptions about itself and other agents, and the present approach will be responsible of driving the reactive layer in a three-layer cognitive architecture.

ACKNOWLEDGMENT

Supported by the Programme Alban, the European Union Programme of High Level Scholarships for Latin America, scholarship No. E05D056455CO".

REFERENCES

[1] R.A. Brooks, A Robust Layered Control System For A Mobile Robot, IEEE Journal Of Robotics And Automation, RA-2, 1986, 14-23.
[2] M.J. Mataric, Behavior-based control: Main properties and implications, Proceedings of the IEEE International Conference on Robotics and Autonomation, Nice, Francia, 1992, 2-8.
[3] R.A. Brooks, How to build complete creatures rather than isolated cognitive simulators, Architectures for Intelligence, 1991, 225-239.
[4] J. R. Koza, Evolution of subsumption using genetic programming, Proceedings of the First European Conference on Artificial Life, Paris, 1992, 110-119.
[5] M.V. Butz, T. Kovacs, S. Wilson, How {XCS} evolve accurate classifiers, Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, 2001, 927-934.
[6] J.H. Holland, Induction, Processes of Inference, Learning and Discovery, (Mich:Addison-Wesley, 1953).
[7] L. N. de Castro, J. Timmis, Artificial Immune Systems: A New Computational Intelligence Approach, (Ed. Springer, 2002.)
[8] D. Romero, L. Niño, An Immune-based Multilayered Cognitive Model for Autonomous Navigation, IEEE Congress on Evolutionary Computation, Vancouver, 2006, 1115-1122.
[9] C. Watkins, Q-learning, Machine Learning 8, Boston, 1992 – pp. 279-292.
[10] V. Kuzmin, Connectionist Q-learning in Robot Control Task, Proceedings of Riga Technical University, 2002, 112-121.
[11] J. Jiang, M.S. Kamel, Aggregation of Reinforcement Learning Algorithms, IEEE Congress on Evolutionary Computation, Vancouver, 2006, 68-72.
[12] C. Ferreira, Gene Expression Programming: A new adaptive algorithm for solving problems, Complex Systems, Vol. 13, 87-129, 2001.
[13] P.Stone, Layered Learning in Multiagent Systems, (Doctor Thesis CMU-CS-98-187, 1998)
[14] A. Farahmand, Hybrid Behavior Co-evolution and Structure Learning in Behavior-based Systems, IEEE Congress on Evolutionary Computation, Vancouver, 2006, 979-986.
[15] P. Emerson, Designing an All-Inclusive Democracy, Collective Decision-making: The Modified Borda Count, Springer Verlag, 2007, 15-38.