Fighting Fire with Fear

Luís Sarmento
Faculdade de Engenharia da Universidade do Porto
Rua Doutor Roberto Frias
4200-465 Porto, Portugal
las@fe.up.pt

Daniel Moura
Faculdade de Engenharia da Universidade do Porto
Rua Doutor Roberto Frias,
4200-465 Porto, Portugal
dcm@fe.up.pt

Eugénio Oliveira
Faculdade de Engenharia da Universidade do Porto
Rua Doutor Roberto Frias,
4200-465 Porto, Portugal
eco@fe.up.pt

Abstract

In this paper, we propose an Agent Architecture including Emotion-like mechanisms to improve the global performance of the Agent in real-time environments. Emotions play a significant role in allowing resource-bounded Agents to manage multiple simultaneous goals, to adapt to dynamical environments and to optimize their own resources allocation. We start by presenting a computational model of Emotion-like mechanisms and explaining how this model can be usefully applied in Agent Architectures. We will then introduce Pyrosim, a simulation platform where teams of Agents cooperate to fight a forest-fire, which has been the base scenario for our development. We will proceed by presenting the current implementation of the Emotion-based Architecture and explaining the role of Emotion-like mechanisms. In particular, we will show how Emotion-like mechanisms such as "Fear" "Anxiety" or "Self-Confidence" can be used (i) to dynamically adapt the global computational effort of the Agent to the current situation, (ii) to change priorities and previous scheduling of the Agent's Goals, (iii) to adapt internal parameters according to the Agent's own success and (iv) to improve team coordination by influencing leadership decision-making. We will also discuss the possibility of interaction between Emotion-based mechanisms and Reinforcement Learning.

1. Introduction

In the last years, there has been a resurgence of the topic of Emotion in Artificial Intelligence after the pioneer work of Herbert Simon [1] nearly 40 years ago. This interest has been motivated by a deeper comprehension of Emotion developed in the field of neurology and psychology during the 90´s. Researchers have been able to establish significant connections between Emotion and specific cognitive processes such as decision-making [2], risk assessment [3], coding and retrieval of information in memory [4], and Goal management. In fact, Emotions are thought to play an essential role in making some Human cognitive processes feasible in practice, especially considering that Humans are resource-bounded Agents that operate in very complex, real-time environments. Emotion is also thought to be crucial in the promotion of specific information processing contexts *(Processing Strategies* [4]) enhancing the functional coordination amongst the various cognitive capabilities of the Agent. All these conclusions about the role of Emotion in Human cognition have led several AI researchers to include Emotion-based concepts in their Agent Architectures trying to enhance Agent capabilities. Emotion-based concepts have been included in Agents in order to improve specific processes such as decision-making [5][6], learning [7] and memory management [8]. Additional research lines have been followed in trying to develop complete Agent Architectures in which Emotion-based concepts play a central role in the global behaviour of the Agent [9][10][11].

The work developed in [12][11] has tried to bring a deeper understanding of how Emotion-based concepts may be applied to Agent Architectures to improve Agent capabilities, especially for those that work in real-time environments and have multiple simultaneous Goals. The Architecture proposed in [11] includes multiple Emotion-based mechanisms that are able to influence various parameters of the Agent Architecture. Specifically, the role of Emotion-based mechanisms ranges from altering specific parameters of individual functional modules to influencing the global processor allocation of the Agent capabilities. In this paper we will try to explain how these Emotional-mechanisms may be implemented and included in Agent Architectures.

2. Emotional-like Mechanisms

The notion of a Basic Emotional Structure has been developed in [11] and is a central concept in the proposed

Architecture. Briefly, a Basic Emotional Structure is a time dependent process exhibiting a similar dynamical behaviour to that of an Emotion. A Basic Emotional Structure (figure 1-a) is composed of two elements: an Emotional Evaluation Function (EEF), and an Emotional Accumulator (EA). EEFs are functions that receive as inputs a vector <E>, containing the information perceived by the Agent about the outside environment, a vector <I>, which contains information about the internal state of the Agent, and returns a scalar value that is related to the success chances of a given goal G. EAs are time (t) dependent processes that incrementally store a percentage (P_{input}) of output value of an Emotional Evaluation Function, EEF_G. The value stored by an Emotional Accumulator decays exponentially with a specific time constant (T_d).

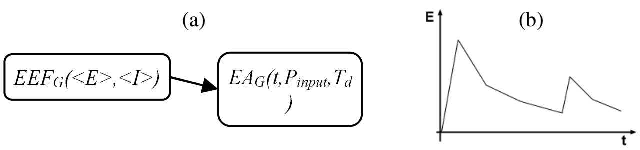

Figure 1. The Basic Emotional Structure (a), and the time behaviour of an Emotional Accumulator (b).

The essential notion about these Emotional-like mechanisms is that they are closely related to Goal success. Emotional Evaluation Functions should be able to reflect how relevant are environmental (internal and external) conditions to a given Goal. They should implicitly evaluate how favourable is the environment to the achievement of a specific Goal considering the current capabilities of the Agent. The result of this evaluation stimulates the associated Emotional Accumulator that provides a time varying profile like illustrated in figure 1-b. Values stored in Accumulators reflect the global success of the Agent in pursuing a given set of Goals. Reusing these values internally is the core of the Emotional Mechanisms.

3. Pyrosim Platform

We now will introduce the Pyrosim Agent platform that we have been using in our experiments. The Pyrosim platform simulates a forest-fire environment where a team of Agents (firemen) cooperates to control and extinguish the fire, while simultaneously trying to minimize the overall damage and losses. In Pyrosim, Agents have to deal with very dynamic fire fronts, terrain constraints and their own physical and logistic limitations. Each Agent needs to ensure its physical safeness while trying to fight the fire and, at the same time, help colleagues to remain safe. Agents are equipped with a water jet with limited power that allows them to put out the fire, but they are not normally able to do it individually so there is an obvious need for cooperation. Agents may communicate with each other in order to organize team efforts (broadcast and 1-to-1 messages). Agent's Perception System provides information about his own state (physical energy, speed, acceleration, position in the terrain, status of the personal water jet) as well as several matrix structures named *Vision Maps* that describe close range and medium range surroundings. Visual Maps contain information with different *levels of detail* and *noise* (depending on the distance) about terrain, vegetation, level of destruction, and fire cells. Agents also receive information about visible parameters of other Agents (location, approximate energy level and action). We have developed a communication layer - the *AgentSkeleton* - that deals with the low-level details of the simulator, allowing Agents to be developed from a higher level.

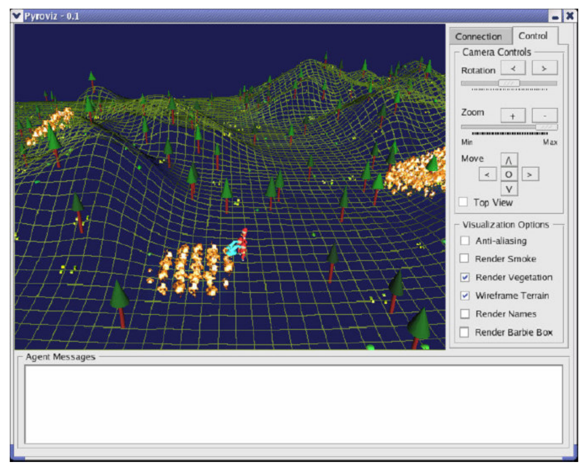

Figure 2. A visualization (in the Pyroviz visualizer) of a simulation in the Pyrosim simulator.

4. Architecture Overview

The proposed architecture is composed of two layers built on top of the basic Agent Skelton layer. Each of these layers is related to a set of specific *Goal*s and has a set of increasingly high-level capabilities. Despite this conceptually layered disposition, our Architecture is composed of several distributed functional modules whose place within the layers is not so conceptually self-contained.

4.1. Functional Modules

Functional modules are the building blocks of our Architecture and may be grouped in five distinct categories:

- **Perceptors** – transform sensory data provided by the Agent Skeleton (e.g. visual maps) in higher-level Beliefs;

- **Reasoners** – higher-level modules that operate on Beliefs produced by preceptors and are mainly concerned with decision-making, planning and high-level analysis of data;
- **Managers** - one *Goal Manager*, responsible for allocating processor time to Goals, and one Action Manager that pipes actions to the Skeleton Layer.
- **Controllers** – transform high-level commands into lower level actions (ex: rotation, acceleration, etc.).
- **Emotional Mechanisms** – Emotional Accumulators are constantly updated by Emotional Evaluation Functions according with the environment and the state of the Agent.

The proposed Architecture also includes a memory structure named *Working Memory* that works similarly to a blackboard. Functional modules may use Working Memory to store and retrieve Beliefs indexed by keyword. For example, after processing local sensory data, Perceptors can store in Working Memory all the new Beliefs produced, which then become available as input for Reasoners to produce supplementary higher level Beliefs (e.g.: a plan, a global fire analysis).

Finally, our Architecture also comprises another memory module called *Execution Memory*. The *Execution Memory* is responsible for keeping track of the execution state of the Agent, allowing the Agent to reason about it past actions and helping to decide the current and future actions. The Execution Memory stores the sequence of actions performed by the Agents as well as information about specific events related with those actions.

4.2. Layered Architecture

In our architecture, layers should be seen as an aggregation of information processing modules, whose overall processing time is tightly connected with a specific set of *Goal*s. Lower level layers correspond to groups of modules that are connected to more basic and essential goals. On the other hand, higher-level layers include modules that are related to less urgent goals or to those that depend on information produced in lower-level layers. The significant feature of the proposed architecture is that the management of processing time is done upwards, from lower level layer to higher level. Modules that are related to higher-level layers will only run after those inside lower-level layers. Higher-level modules will usually only run a fraction of the times lower-level do.

The global idea of this architecture, which is thought for environments where Agents have to deal with multiple and simultaneous concerns, is that all modules run in a pseudo-parallel fashion and share the existing computational resources according to the importance of the *Goal*(s) that they help achieve. As will be explained later, Emotions play a crucial role in this process.

4.2.1. The Basic Control Agent Layer. Immediately above the Agent Skeleton layer, we have placed the *Basic Control Agent (BCA) Layer,* which is built around one major *Goal*: to ensure Agent "survival" (must not get itself caught by the fire). BCA Layer includes several functional modules, needed to provide the basic capabilities for achieving such Goal. Starting with Perceptors, the BCA Layer includes a *Pain Perceptor,* which detects rapid decreases in Agent energy that normally result from harmful events, a *Temperature Perceptor*, capable of analyzing environment temperature and its variations and two *Fire Perceptors* that analyze close range and medium range Visual maps to extract data about the surrounding fire cells (intensity of burning cells; distances; the starting of new fire spots). Processing tasks associated with these Perceptors are executed during each BCA Layer Execution Cycle, and produce (or revise) Beliefs that become globally available through the Working Memory.

Also in this layer is the *Goal Manager* module that is responsible for scheduling the Agent's *Goal*s. In our architecture, each *Goal* is basically a program that consumes sensory information, Beliefs stored in Working Memory and values of Emotional Accumulators, and produces high-level commands or new Goals. The main *Goal* at this layer is the *Survival Goal,* which is constantly monitoring relevant Beliefs (temperature, intensity of fire, the value of "Fear" Accumulator explained later). According to certain conditions, it generates other *Goals* including one that will make the Agent run away from fire.

Finally, the *BCA Layer* is also responsible for providing processing time to its upper layer. The execution cycle of the upper layer, the Basic Deliberative Agent Layer, is chained in the execution cycle of this layer as if it was another functional module. Therefore, the Basic Deliberative Agent Layer will only be given a share of BCA Layer processing time that needs to be divided among its own functional modules.

4.2.2. The Basic Deliberative Agent Layer. The Basic Deliberative Agent (BDA) Layer is essentially focused on specific task related Goals. The functional modules contained in this layer provide the Agent with the mental capabilities needed to achieve fire-fighting related Goals. The first group is directly concerned with fire-fighting operations and includes operational goals like "Move to a specified location", "(Re)Approach a given fire segment", "Fight a fire in a close range location". The second group is related to the global monitoring of the situation and includes goals such as "Track fire evolution in close range locations", "Track Fire Evolution in Medium Range

Location", "Check if repositioning is needed". The BDA Layer is equipped with Reasoner modules able to operate on the Beliefs produced by lower-level modules. This layer includes the following Reasoners:

- Pocket Map Reasoner: generates path plans taking into account several factors (terrain geometry, fire location, etc.).
- Fire Evolution Reasoners: analyses visual maps to extract data about evolution of the fire (e.g. speed of spread).
- Fire Map Reasoner: analyses global fire estimates to extract features about fire segments (size, intensity, distance, etc.).
- Fire Segment Evolution Reasoner: tracks the evolution of fire segments (speed and direction).

The BDA Layer also includes the Execution Memory module to allow reasoning about previous actions. Information stored in Execution Memory is used to decide new goals coherently (ex: return to a location that the Agent had previously ran away from).

5. The Emotional Mechanisms

Before beginning to explain the exact function of Emotional Mechanisms, it is important to emphasize some properties about our architecture concerning functional modules, Goals and the execution cycles of layers.

First, it must be taken into account that the operation of functional modules is usually highly configurable and subjected to several parameters. For example, the Fire Map Reasoner calculates the path plan using an oriented depth-first search in the action-state space. This planning procedure involves at least 3 parameters: the depth and the breadth of the search and the function used to evaluate the cost of each plan step (taking into account risk, completion time, physical effort, etc). Changing the values of the parameters has impact both on the final plan (e.g.: quicker plan but involving some risky steps) as well as on computational cost of the search process itself (processing time, memory required). Goals also have configurable parameters. For example, the Goal "Fight a fire in a close range location" involves three threshold temperatures that are used by the Agent to decide whether it should approach the fire, keep its position or step back to a less dangerous position. Altering these values will change the average distance of the Agent to fire and, consequently, the global efficiency of fire combat since the efficiency of the water jet decreases with distance from target. There is also possibility of varying the internal behaviour of each layer. Layers are responsible for deciding which functional modules will be scheduled to run in each execution cycle. In practice, layers decide how much time is spent in analysing perceptions, in controlling current actions, in general decision-making or in exploring possible scenarios.

Since many configurations for the available parameters are possible, an efficient method to set these parameters over the entire architecture is required. It would be particularly interesting to obtain configurations that efficiently allocate the limited processing capabilities of the Agent in response to the environmental conditions: processing resources should be spent where most needed. This is precisely where Emotional-mechanisms come into play. We have included in our Architecture 3 different Emotional Mechanisms, which, for the purpose of visualization, we will be referring to by the functionally correspondent Human Emotions: "Fear", "Anxiety", and "Self-Confidence".

5.1. Fear

We start by analysing the "Fear" Emotional Mechanism because it is probably the most immediately applicable to the Architecture. "Fear" is elicited whenever a possibly dangerous or uncontrollable situation is detected, i.e. whenever the Survival Goal is at stake. "Fear" should reflect the inability of the Agent to cope with the current situation with its current resources. For instance, if the temperature becomes too high or if it increases suddenly, the "Fear" level should be increased because the Agent is probably facing a survival-threatening situation. The Emotional Evaluation Function of "Fear" includes factors such as temperatures, fire distance and intensity, the level of the "Pain" Perceptor (which results from energy losses) and the current energy level of the Agent.

It seems reasonable in dangerous situations that the Agent should concentrate its resources in coping with the possible threat: the processing time attributed to analyzing local data and controlling actions should be increased. Additionally it is important to increase the frequency of perception updates to ensure that the Agent is operating with the most recent data. On the other hand, less urgent Goals (e.g.: global monitoring of the fire evolution) and certain higher-level modules should be scheduled less frequently to decrease competition for processor time. The following snippet of pseudo-code illustrates how these conditions could be implemented:

```
IF (FEAR.level >= MEDIUM) THEN
    PERCEPTORS.frequency++;
    BDALayer.scheduling_frequency--;
    GOALMANAGER.min_scheduling_priority++;
```

The "Fear" Emotional Mechanism has also effects at other more specific levels. For example, parameters related to path planning are changed as a function "Fear". Whenever the "Fear" is high, the complexity of planning procedure is reduced in order to obtain a plan more rapidly: both depth and breadth of the associated search algorithm are decreased. The plan thus obtained is possible not the "best" one, but is certainly good enough given the urgency of the situation and the eventually scarce processing resources available at that moment. In addition, since "Fear" signals that the Agent is facing a dangerous situation it seem natural that it should adopt a more pessimist posture towards the environment for a certain period. Acting pessimistic will make the Agent avoid other risky situations. For instance, the Agent will keep a distance from fire that depends on the surrounding temperature. Initially, the Agent is set not to move backwards until the temperature reaches 60º C, allowing him to fight the fire from shorter distances and therefore more efficiently. However, if "Fear" is high the threshold temperature is decreased and the Agent will start moving back sooner, keeping a larger and safer distance from fire. Naturally, "Fear" Emotional Accumulator will decay with time and if no other fearful episodes occur, the pessimistic attitude will tend to disappear.

5.2. Anxiety

"Anxiety" is an Emotional Mechanism related to the anticipation of situations that will probably be difficult to cope with. Contrary to the case of "Fear" that is connected to very urgent or immediate life-threatening situations, "Anxiety" results from the evaluation of the environment and of the Agents own capabilities concerning future possibly threatening situation. For example, "Anxiety" will increase whenever the Agent receives information about the existence of large fires in a nearby location (environmental factor) because a difficult situation will possibly be around. If the Agent starts running out of water (Agent internal factor), Anxiety will also increase. Other factors include the energy level, wind speed, terrain geometry, number of surrounding fellow Agents and Beliefs produced by Reasoners, especially those related with Fire analysis (e.g.: Fire Map Reasoner). In a certain way, "Anxiety" works as a "prevention" mechanism. High "Anxiety" levels will promote a very intensive processing state so that the Agent will be able to detect possible Goal threatening situations or favourable opportunities. The Agent will invest a great deal of its processing resources in analysing perception data and producing higher-level Beliefs to use in decision of new Goals. This is generally achieved by increasing the execution frequency of the Basic Deliberative Layer where Reasoners are scheduled to run. As a result, the Agent will be configured to produce updated Beliefs about the areas where the fire is progressing faster, which ones are the most dangerous (where fire seems to be converging), and which ones seem more worth fighting (e.g.: less damaged). The Agent will also increase the breadth and depth of the path planning procedure in order to increase the quality of the final plan. At the same time, plans are revised frequently to integrate the knowledge of the updated Beliefs produced by Reasoners.

The increase in the processing load needs to be compensated by reducing the processor time allocated to other modules. Since the Agent is not going through a very urgent situation, which would be signalled by "Fear", it is possible to decrease the frequency of the Perceptors and Controllers and thus relax the load of the Basic Control Layer. This extra computational power may now be spent in the Basic Deliberative Layer and allocated to Reasoners. However, this situation may not be sustainable for long periods because the Agent will eventually need to focus all its resources on a nearby fire cell. But in that case, "Fear" increases and subsumes the effects of "Anxiety". The following snippet of code tries to illustrate the effects of "Anxiety" and its dependency to "Fear":

```
IF (ANXIETY==HIGH) AND (FEAR<MEDIUM) THEN
    PLAN_BREADTH = HIGH
```

5.3. Self Confidence

Self-Confidence is an Emotional Mechanism directly related with the success achieved in previous Goals. If the Agent is successfully accomplishing Goals, the level of Self-Confidence should be increased to reflect that the Agent has enough capabilities to cope with the environment. Therefore, high-levels of "Self-Confidence" signal that the current environment poses no significant difficulties to the Agent. In such situation, the Agent could simply relax its information processing strategies and use the released processing resources in other tasks or Goals that are not urgent but may become advantageous later. Additionally, Agents should adopt a more optimistic approach towards the environment and stretch its own previous limits (e.g.: decrease distance to fire, increase distance to other firemen). For the sake of brevity, we will not discuss in greater detail the influence of "Self-Confidence" on the internal parameters of the Agent. Alternatively, on the next section we will address the possibilities of Emotional Mechanisms regarding the coordination of teams of Agents.

6. Emotion & Coordination

In fire fighting scenarios, team coordination implies two goals: fighting fire efficiently and safely. Such coordination goals impose several decisions difficult to make, such as, (i) how to position the team geographically, (ii) when to change team positioning, (iii) when to call for backups or (iv) when to evacuate. These questions are hard to solve, especially considering that changing positioning is often slow and backups are limited and delayed.

6.1. Strategy in fire-fighting scenarios

When attacking wild-forest fires, it is difficult to find strategies that can both quickly return positive results and maintain their effectiveness for long periods. Most of the times, firemen are outnumbered and a compromise between short-term results quality and long-term results quality must be established. Strategies that provide mid and long-term results are preferred because they are more effective despite being much more difficult to carry out.

Table 1. Typical strategies behaviours

Strategy	Short-Term	Mid-Term	Fire Propagation
A	Bad	Good	
B	Medium	Medium	
C	Good	Bad	

In table 1, we present a set of typical strategy behaviours for this domain. Strategy A is characterized by good mid-term results despite of the considerable overhead preceding the strategy settlement. If the leader is able to predict that the fire does not grow excessively during this overhead period, then this type of strategies are better because of their long-term results. An example of this strategy is the splitting of the team in two in order to attack a "V-shape fire" from behind in both of the edges. This strategy could involve a considerable positioning time but would enable to fight fire more effectively. On the other hand, strategy C could be used when short-term results are required. This kind of strategy aims for a quick settlement but it is only capable of producing results temporarily or locally. However, using strategies like C often jeopardizes long-term incomes. Strategy B provides a compromise between the other two as for example when attacking a V-shape fire from behind concentrating firemen in the nearest edge.

6.2 Strategy selection

For discussing strategy selection let us consider a specific scenario. Figure 2 illustrates the evolution of a fire-fighting situation, in the leader's perspective, based on four indicators: estimated fire propagation, Agent satisfaction (calculated using an utility function regarding coordination goals fulfilment), Agent "Self-Confidence" and Agent "Anxiety". When situation reaches point *A*, firemen start to lose control of the fire because of a scenario alteration (e.g. wind speed raised) or a strategic action of the leader (e.g. changing to a type A strategy). After detecting that the fire is expanding, the leader must decide how long it should wait until a new strategic action becomes inevitable. When faced with such low satisfaction levels, a Deliberative Agent based on a utility function would probably change strategy before reaching point *B*, missing the strategy turning point. This would happen because the aborting decision is based on a rule without taking into account the previous success of the Agent. On the other hand, the decision of an Emotion-based Agent would depend on its Emotional State. As we can see in figure 2, the Agent "Self-Confidence" is high enough to handle the pressure and to achieve the strategy turning point (point *B*). These high levels of "Self-Confidence" reflect that the leader is being successful in its strategy decisions, which motivates him to believe that new decisions will also turn out well. In contrast, a leader with lower levels of "Self-Confidence" reveals that its strategy decisions are not working, probably because of adverse fire fighting conditions. Such leaders would be more risk-adverse and would try to use strategies that result at short-term.

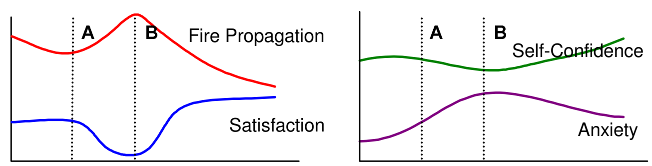

Figure 3. Scenario Evaluation: Satisfaction vs. Emotions

At the end of the situation illustrated in figure 2 the leader is more "Self-Confident" because it handled a difficult situation with success. Also worth noticing is "Anxiety" evolution. Stressful situations, such as this, even when concluding well, raise Agents' "Anxiety". Consequently, leaders raise their attention levels in order to detect possible turn backs in the scenario evolution.

We have verified how Emotions can help strategy selection in face of the team success. However, there are also environmental factors that can help strategy decision, such as fire geometry and terrain geography. Such factors combined with the leader's Emotional State may constitute the base for strategy decisions. Because these decisions are still difficult to formulate by rules we propose using machine learning to map states (Emotional and environmental) into strategies.

Emotion & Goal-directed Learning

Emotional Mechanisms provide goal-related information, which makes them potentially useful for goal-oriented processes. This is the case of *Reinforcement Learning* (RL), a machine learning methodology focused in goal-directed learning from interaction [13]. Currently, we are working in using the Agent Emotional State for (i) calculating reinforcements, (ii) influencing the exploration vs. exploitation trade-off, and (iii) defining the state. We will now briefly focus on the later topic using as example a scenario where a leader is learning how to select strategies (from a pre-defined set of strategies).

As we have seen before, “Self-Confidence” may be used to decide the kind of strategy to apply (section 6). By using this Emotional-Mechanism, Agents may skip the analysis of several environmental properties that would lead to similar conclusions. This could be a major advantage to the state definition because one Emotional Variable replaces several environmental variables, which may result in a decrease of possible states. With fewer states to explore, the learning algorithm may converge faster. However, most of the times the state will also include environment variables because of specific environment information that may be important to the skill being learned (e.g. fire geometry when learning strategy selection).

7. Conclusion

In this paper we have presented an Agent Architecture intended for resource-bounded Agents that operate in complex environment and have simultaneous goals. The Architecture is composed of several distributed processing modules, organized in two layers and includes a set of Emotional mechanisms whose activity is directly related to the performance of the Agent in achieving its Goals. We have shown how such Emotional mechanisms may be functionally used (i) to increase the adaptation of the Agent to the environment by influencing several internal parameters, (ii) to optimise the allocation of processing resources and (iii) to improve strategic decisions. We have finished by pointing out possible uses of Emotional Mechanisms in Reinforcement Learning.

8. References

[1] H. Simon, Motivational and emotional controls of cognition, *Psychological Rev.*, 74, (1967).

[2] A. Damásio, *Descartes Error - Emotion, Reason and the Human Brain*, Gosset/Putnam Press, New York, 1994.

[3] N. Schwarz, ‘Emotion, Cognition, and Decision Making’, C. Smith (Ed) *Cognition and Emotion*, V. 14, I. 4, Psychology Press, 2000.

[4] G. Bower and J. Forgas, ‘Affect, Memory and Social Cognition’, Eich, E. et al. (Eds). *Cognition and Emotion*. Oxford U. P., 2000.

[5] J. Velásquez, ‘Modeling Emotion-Based decision making’, D. Cañanero (Ed.), *Emotional and Intelligent: The Tangled Knot of Cognition*, 1998.

[6] D. Canãmero, *Designing Emotions for Activity Selection*, TR - DAIMI PB 546, Univ. of Arhaus.

[7] S. Gadanho, ‘Emotional and Cognitive Adaptation in Real Environments’, *Symposium ACE'2002 of the 16th European Meeting on Cybernetics and Systems Research*, Austria, 2002.

[8] L. Botelho and H. Coelho, ‘Machinery for Artificial Emotions’, *Journal of Cybernetics and Systems*, V. 32, Nº 5, 464–506, (2001).

[9] I. Wright, *Emotional Agents*, Ph.D. thesis, School of Computer Science, The University of Birmingham, 1997.

[10] A. Sloman, B. Logan. ‘Evolvable Architectures for human-like Minds’, G. Hatano, N. Okada and H. Tanabe (Eds), *13th Toyota Conference - Affective-Minds*, 169-181, Elsevier, Amsterdam, 2000.

[11] E. Oliveira and L. Sarmento, Emotional Advantage for Adaptability and Autonomy, *Proceedings of the second international joint conference on AAMAS*, 305–312, (2003).

[12] E. Oliveira and L. Sarmento, ‘Emotional Valence-based Mechanisms and Agent Personality’. Proc. *SBIA'02*, LNAI, Springer Verlag, 2002.

[13] R. Sutton, A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, Massachusetts, 1998.