
KLAUS RAIZER

# EXECUTIVE FUNCTIONS FOR LEARNING AND DECISION-MAKING IN A BIO-INSPIRED COGNITIVE ARCHITECTURE

# *FUNÇÕES EXECUTIVAS PARA APRENDIZADO E TOMADA DE DECISÃO EM UMA ARQUITETURA COGNITIVA BIO-INSPIRADA*

Campinas
2015

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

KLAUS RAIZER

# EXECUTIVE FUNCTIONS FOR LEARNING AND DECISION-MAKING IN A BIO-INSPIRED COGNITIVE ARCHITECTURE

## *FUNÇÕES EXECUTIVAS PARA APRENDIZADO E TOMADA DE DECISÃO EM UMA ARQUITETURA COGNITIVA BIO-INSPIRADA*

Thesis presented to the School of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering, in the area of Computer Engineering.

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: Engenharia de Computação.

Supervisor: Ricardo Ribeiro Gudwin

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL
DA TESE DEFENDIDA PELO ALUNO KLAUS RAIZER,
E ORIENTADA PELO PROF. DR. RICARDO RIBEIRO
GUDWIN

Campinas
2015

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

R137e

Raizer, Klaus, 1982-
Executive functions for Learning and decision-making in a bio-inspired cognitive architecture / Klaus Raizer. – Campinas, SP : [s.n.], 2015.

Orientador: Ricardo Ribeiro Gudwin.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Inteligência artificial. 2. Ciências cognitivas. 3. Robôs móveis. I. Gudwin, Ricardo Ribeiro,1967-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Informações para Biblioteca Digital

**Título em outro idioma:** Funções executivas para aprendizado e tomada de decisão em uma arquitetura cognitiva bio-inspirada
**Palavras-chave em inglês:**
Artificial intelligence
Cognitive sciences
Mobile robots
**Área de concentração:** Engenharia de Computação
**Titulação:** Doutor em Engenharia Elétrica
**Banca examinadora:**
Ricardo Ribeiro Gudwin [Orientador]
João Eduardo Kögler Junior
Henrique Elias Borges
Fernando José Von Zuben
Fernando Antônio Campos Gomide
**Data de defesa:** 27-02-2015
**Programa de Pós-Graduação:** Engenharia Elétrica

## COMISSÃO JULGADORA - TESE DE DOUTORADO

**Candidato:** Klaus Raizer

**Data da Defesa:** 27 de fevereiro de 2015

**Título da Tese:** "Executive Functions for Learning and Decision-Making in a Bio-Inspired Cognitive Architecture (Funções Executivas para Aprendizado e Tomada de Decisão em uma Arquitetura Cognitiva Bio-Inspirada)"

Prof. Dr. Ricardo Ribeiro Gudwin (Presidente): Ricardo Gudwin

Prof. Dr. João Eduardo Kögler Junior:

Prof. Dr. Henrique Elias Borges:

Prof. Dr. Fernando José Von Zuben: Fernando José Von Zuben

Prof. Dr. Fernando Antônio Campos Gomide:

ABSTRACT

This work's goal is the development of executive functions for a codelet-based bio-inspired cognitive architecture. One of the major challenges every creature faces, being biological or artificial, is to define the next action to be taken, at each time step, as a function of how it perceives its surrounding environment. This decision can be made by a reactive algorithm, which always repeats the same decisions for a given situation, or by an adaptive process, which is able to make use of learning mechanisms in order to make distinct decisions based on past experience. In this work, deliberative decision-making and reinforcement learning mechanisms have been integrated into a single framework. In cognitive science literature, these functions are known as executive functions. The solution proposed here is part of our group's central line of research, which is the investigation and development of a codelet-based cognitive architecture. In this context, a central contribution made by this work is the development and implementation of algorithms capable of providing this cognitive architecture with a group of executive functions, which in turn can be used to implement complex solutions with arbitrary granularity.

Functions for deliberative decision-making have been implemented in the form of a modified behavior network, while the learning component was developed in the form of a new algorithm called GLAS (*Gated-Learning Action Selection*), based on stimulus gating and known computational neuroscience models. This framework has been validated with problems in mobile robotics and in action selection by reinforcement learning.

The cognitive architecture under development, when incremented by the contributions presented in this work, has the potential to serve as a base for future work and research in the fields of artificial intelligence, robotics and artificial cognition.

**Keywords:** Artificial intelligence, Reinforcement learning, Cognitive architectures, Mobile robots, Assistive technology

RESUMO

O objetivo deste trabalho é o desenvolvimento de funções executivas para uma arquitetura cognitiva bioinspirada baseada em codelets. Um desafio que toda criatura (seja ela artificial ou biológica) enfrenta é definir qual a próxima ação a ser tomada, a cada instante de tempo, em função da percepção de um determinado ambiente. Essa decisão pode ser definida por um algoritmo que sempre repete as mesmas decisões em função de uma determinada situação, ou pode ser uma decisão adaptativa, que utiliza de mecanismos de aprendizagem para assumir decisões distintas, em função das experiências em situações passadas. Neste trabalho, buscou-se a integração dos processos de tomada de decisão deliberativos e mecanismos de aprendizado por reforço em um mesmo framework. Estas funções são conhecidas na literatura de ciências cognitivas como funções executivas.

A solução aqui proposta insere-se dentro do contexto de nosso grupo de pesquisa, onde se busca o desenvolvimento de uma arquitetura cognitiva baseada em codelets. Nesta perspectiva, uma das contribuições deste trabalho é desenvolver algoritmos e implementações computacionais dotando a arquitetura cognitiva desenvolvida pelo grupo de funções executivas diversas, que poderão ser utilizadas para implementar soluções complexas com granularidade arbitrária. As funções de tomada de decisão deliberativa foram implementada na forma de uma rede de comportamentos modificada, enquanto que o componente de aprendizado foi desenvolvido na forma de um novo algoritmo (GLAS - *Gated-Learning Action Selection*) baseado em *stimulus gating* e inspirado em modelos de neurociência computacional conhecidos da literatura. Este *framework* foi validado em problemas de robótica móvel e de seleção de ação por aprendizado por reforço.

A arquitetura cognitiva sendo desenvolvida, incrementada com as contribuições deste trabalho, tem o potencial de servir de base para futuros trabalhos de pesquisa nas áreas de inteligência artificial, robótica e cognição artificial.

**Palavras-chave:** Inteligência artificial, Aprendizado por reforço, Arquiteturas cognitivas, Robôs móveis, Tecnologia assistiva

# Contents

To my family.
For their support and
understanding.

À minha família.
Pelo apoio e compreensão.

# Acknowledgement

I would like to thank:

My family for all the support during these years of PhD work. I am specially grateful to my wife Eliane for her love, companionship and unconditional support during this journey.

Prof. Dr. Ricardo Ribeiro Gudwin for his guidance.

My colleagues from the CogSys group at FEEC. Specially to André L. O. Paraense for our combined effort at developing the foundation of the cognitive architecture.

All researchers at LCA laboratory who helped me along the way. Special thanks to Dr. Eric Rohmer for his help in all things related to robotics.

My coworkers at CPqD (Centro de Pesquisa e Desenvolvimento em Telecomunicações) for providing such a rich learning environment. And specially to my manager Cláudia Piovesan Macedo for supporting my effort at completing this work.

All the professors at FEEC (Faculdade de Engenharia Elétrica e de Computação) and FEM (Faculdade de Engenharia Mecânica). And also to Dr. Bernard J. Baars, for the long emails we exchanged about his work and his valuable insights.

"Never can we sit back and wait for miracles to save us. Miracles don't happen. Sweat happens. Effort happens. Thought happens."

Isaac Asimov

# List of Figures

# List of Tables

# Acronyms

| | |
|---|---|
| ANOVA | Analysis of Variance |
| BCI | Brain–Computer Interface |
| BG | Basal Ganglia |
| BN | Behavior Network |
| BP | Behavior Proposition |
| CCS | Central Control Server |
| CS | Cognitive Skill |
| ECG | Electrocardiogram |
| EEG | Electroencephalogram |
| EMG | Electromyogram |
| GA | Genetic Algorithm |
| GUI | Graphical User Interface |
| GLAS | Gated Learning Action Selection |
| JSON | JavaScript Object Notation |
| MLP | Multilayer Perceptron |
| PBWM | Prefrontal Cortex Basal Ganglia Working Memory |
| PFC | Prefrontal Cortex |
| POI | Point of Interest |
| RFID | Radio-Frequency Identification |
| SARSA | State Action Reward State Action |
| spBN | Soft-Precondition Behavior Network |
| SSVEP | Steady State Visually Evoked Potential |

# Chapter 1

# Introduction

One of the most important goals any intelligent agent has is to decide which action to take next (Franklin, 1995). However trivial this task might seem to us on our day to day experience, it took nature millions of years of evolution to come up with the particular solution for action selection we call "consciousness".

Given the challenge of deciding which action to choose next, evolution has produced in the vertebrate brain the mechanism responsible for what we perceive as being our conscious experience [1]. In the context of artificial intelligence, the mathematical and algorithmic study of animal consciousness has produced a line of research called "machine consciousness". Even though there is still no consensus on what machine consciousness is, its study has produced a number of interesting results in a computational point of view (Baars and Franklin, 2009; Dubois, 2007; Bogner, 1999).

However, conscious action selection is very sophisticated and as such depends on a number of other cognitive functions, such as those associated with the concept of a central executive. The main functions of a central executive in the modern mammal brain are action selection, planning, selective attention and rule learning (Baars and Gage, 2010; Fuster, 2008; Frank and Badre, 2012), and are believed to be enabled by the prefrontal cortical regions of the brain. For humans these functions are intimately linked to consciousness, by influencing its content and in turn being influenced by it.

Currently, some of the most promising research developments in the area of human-like action selection are those made in the field of bio-inspired cognitive ar-

[1] It is still unknown if consciousness is present in the brains of animals other than highly evolved mammals like humans. Some works suggest that cephalopods and some species of birds, might have some rudimentary levels of consciousness Seth et al. (2005).

chitectures. Cognitive architectures are *frameworks* for intelligent agents, which are composed of computational processes mimicking human cognition. These processes can be defined either in terms of memory, or in terms of functional capacity to perform computation (Langley et al., 2009), and are designed to provide support for other cognitive mechanisms such as perception, emotions, action, adaptation and motivation (Vernon et al., 2007).

Recently, many research groups have worked with bio-inspired cognitive architectures. Each group, however, has its own view on how to implement those architectures, which is why there are so many differences among their frameworks.

Current architectures most closely related to our work are Stan Franklin's LIDA (Faghihi and Franklin, 2012; Baars and Franklin, 2003), ACT-R (Anderson et al., 2004, 2010) by John R. Anderson's group at the University of Carnegie Mellon, the Leabra-Emergent framework (O'Reilly, Hazy and Herd, 2012; Hazy et al., 2007) by Randall C. O'Reilly, CERA-CRANIUM (Arrabales et al., 2009a,c) by Raúl Arrabales, and the SOAR cognitive architecture (Laird, 2012; Laird et al., 2012) developed by John Laird at the University of Michigan.

Architectures such as LIDA, ACT-R and Leabra are strongly biased towards biological plausibility. One of their central tenets is to respect the biological inspiration in order to serve as platforms for the study of human cognitive processes and mechanisms. Cognitive architectures such as CERA-CRANIUM and SOAR are also bio-inspired. However, they are not restricted to implementing biologically plausible mechanisms (Langley et al., 2009), which therefore renders them free to explore a wider range of technological solutions.

However, at the time of this writing none of the aforementioned works implement in a single framework the capacity to deliberate sequences of behaviors and actions and learning from stimuli and rewards far apart in time.

## 1.1 Reactive and deliberative decision-making

The emergence of complex behaviors out of the interactions between simple components is a major research field in the area of computational intelligence and intelligent agents. In this context, it is important to briefly review the relationship between simple "reactive" behaviors and more complex "deliberative" behaviors.

Reactive behaviors are generally direct mappings between perception and action. As soon as the agent's sensors capture relevant stimuli in the environment, its actuators produce the most relevant response for the given situation. Being reflective as they are, reactive behaviors are usually faster and do not require a complete world model in order to make decisions. Deliberative behaviors on the other hand require considerable knowledge about the world. This knowledge provides the agent the information necessary to estimate probable consequences for chosen actions. This allows a deliberative agent to choose a sequence of actions which is able to take it closer to its goals. Figure 1.1 shows the characteristics of reactive and deliberative behaviors.

The more reactive a system, the higher its response speed, and the lower its capacity to predict the outcomes of its choices and dependence on a world model. On the other hand, the more deliberative a system is, the lower its reaction speed, and the higher its capacity to predict future outcomes and its dependence on world models. To go beyond purely reactive behaviors, the mammal brain is capable of interfering on reflexive behaviors and enhance them in order to orchestrate more complex behaviors able to reach its goals.

## 1.2 Motivations and Goals of this Research

In this study, we have embedded a neuroscience inspired cognitive architecture with an executive algorithm, which is able to learn and perform reactive and delib-

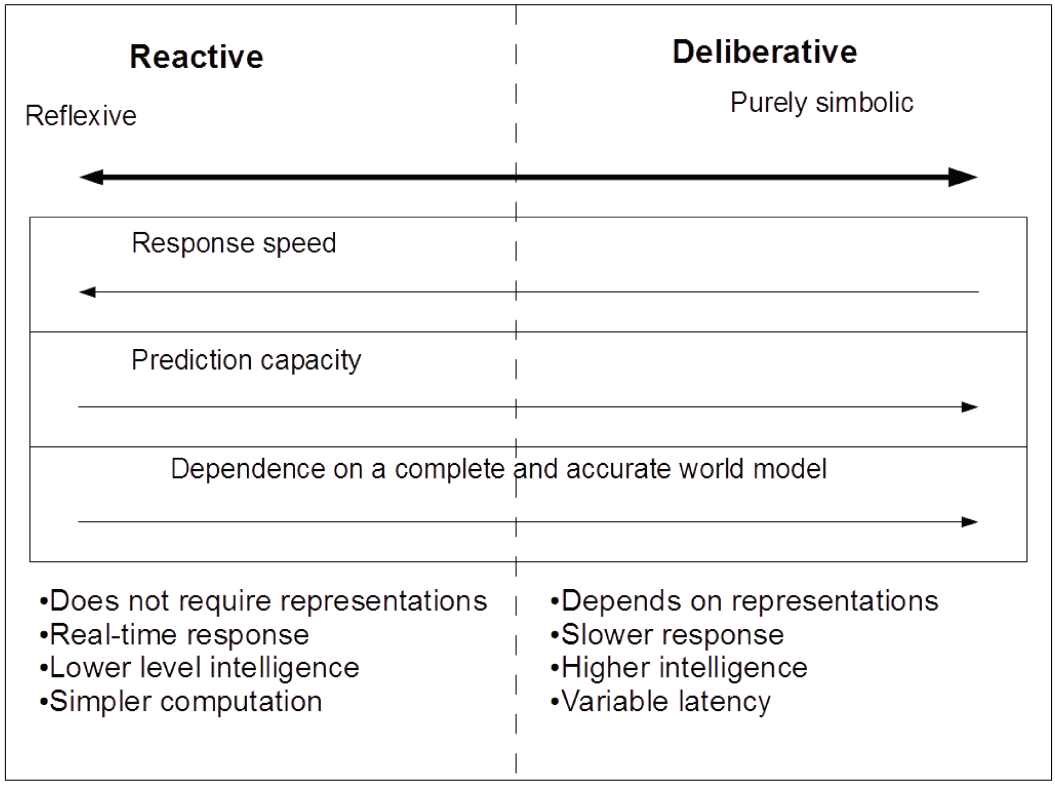



Figure 1.1: Balance between reactive and deliberative systems. Adapted from Arkin (1998).

erative action selection.

The motivation is the importance for all creatures, biological and artificial alike, of choosing the next action to be taken in a given environment (Franklin, 1995). In this context, we have integrated a deliberative action selection mechanism and an algorithm for reinforcement learning into a bioinspired cognitive architecture, which is intended to be more sophisticated than those algorithms traditionally used for controlling artificial intelligent agents.

To perform this integration, we started the development of a codelet-based cognitive architecture. Codelets are small pieces of code, responsible for performing simple tasks in parallel, and which are able to communicate among each other in order to produce more complex cognition (Hofstadter and Mitchell, 1994).

A well known cognitive architecture is LIDA, which is a highly modular architecture - but not completely codelet-based - and strongly based on cognitive neuroscience. It aims, among other things, at being a tool for generating testable hypotheses about human and animal cognition, which might make its use at simpler applications problematic.

The architecture proposed here deals with this applicability problem by being decomposable into three distinct architectures, each with a level of complexity more suitable to a particular application. In other words, this work employs a technological approach, by drawing inspiration from neuroscience in order to develop better intelligent artificial systems. Many works of this type also aim at having a contribution toward taking the scientific side of this research forward, hoping to better understand or make important discoveries about biological consciousness by building successively more complex artificial agents with cognitive architecture. This is not the case of this work, which aims at taking advantage of the new findings in science to build better technologies.

For the development of its deliberative component we have looked for inspiration at the works of Franklin et al. (2014) and Dorer (2010) with behavior networks, which are capable of reactive and deliberative action selection. For the learning component, we chose to integrate this behavior network with a mechanism inspired by the work developed by Randall C. O'Reilly's group (Hazy et al., 2007).

By implementing these executive functions we expect to contribute to the process of developing artificial cognition which is closer to the natural one. Historically, studying the solutions nature has found to many problems has produced a number of problem solving techniques in the realm of artificial intelligence (such as artificial neural networks, genetic algorithms and swarm intelligence), which are also able to control artificial creatures in diverse environments.

It is known that classical artificial intelligence has studied deliberative action

selection in the form of planning algorithms, and learning in the form of traditional reinforcement learning algorithms such as Q-Learning and SARSA (State-Action-Reward-State-Action) (Russell and Norvig, 2010).

More recently, an action selection mechanism capable of both deliberative and reactive action selection called "behavior network" has been studied by Faghihi and Franklin (2012) and Dorer (2010), both based on the original work by Maes (1989). Learning mechanisms for connecting stimuli and rewards separated in time, and which are capable of selective attention, have been investigated in the works by Bakker (2002) and Hazy et al. (2007).

At the time of this writing however, we have no knowledge of any work attempting to implement those two more sophisticated mechanisms under a single computational framework. This is the central issue of this thesis.

## 1.3 Research work

Research work for this thesis followed the steps described in Figure 1.2, where the major contributions are within the dotted boxes named "deliberative decision-making" and "learning and decision-making".

Chapter 2 reviews the literature and presents a theoretical introduction do the central concepts of executive functions and the central executive. It also establishes their relation to modern mammal's prefrontal cortex and human consciousness. This knowledge will clarify which functions we are going to investigate in this study and which functions are outside the scope of this thesis.

Chapter 3 presents our neuroscience inspired cognitive architecture, which was developed in conjunction with André Luis Ogando Paraense and Prof. Dr. Ricardo Ribeiro Gudwin. The architecture was designed to serve as a framework for the development of intelligent agents, and its modular structure made it easier to implement other algorithms.

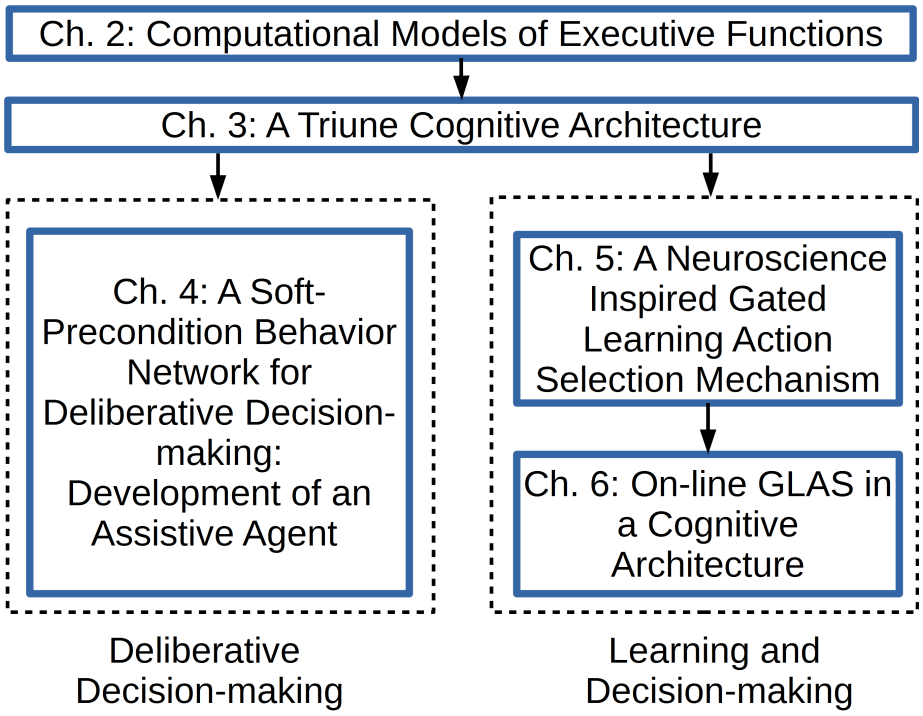



Figure 1.2: Research work

Chapter 4 describes the implementation of a modified behavior network as our cognitive architecture's deliberative action selection mechanism. This behavior network was validated in a simulated robotic application, which is responsible for providing interesting action suggestions to a human patient in the context of an assistive environment.

Chapter 5 presents GLAS: a gated learning action selection mechanism, which is capable of learning the relation between stimuli and rewards distant in time.

Chapter 6 presents the integration of GLAS and our cognitive architecture. The system is then validated in a benchmark application called 1-2-AX Working Memory Task, which was designed to evaluate working memory and executive capabilities.

Finally, Chapter 7 presents a summary of this work, highlights its main contributions and discusses future work for the continuation of this research.

# Chapter 2

# Models of Executive Functions

"Essentially, all models are wrong, but some are useful."

George E. P. Box

## 2.1 Introduction

In the last chapter we mentioned that the emergence of complex behavior in the mammal brain comes from the interaction of basic reactive behaviors. These reactive behaviors are influenced by deliberative mechanisms, which are believed to be highly dependent on the role played by the prefrontal-cortex. These cognitive functions are also known as "executive functions", and studying them will pave the way to understanding high level cognition as we see in human beings and modern mammals.

Cognitive neuroscience has proposed a number of models for each executive function we are going to see in this chapter. Our objective however is not to discuss the validity or biological plausibility of those models, but rather to use them as inspiration for the development of new technologies and algorithms.

In this chapter we investigate the major executive functions identified and studied in the field of cognitive neuroscience. In Section 2.2 we provide a brief description of what executive functions are. We explore the core executive functions identified in studies about human behavior and recent knowledge about brain structure, and comment on their relation with the mechanism of human consciousness. The chapter ends with a discussion on which executive functions are going to be approached in this thesis, and which will be left for future work.

## 2.2 Executive Functions

Executive functions are those brain functions responsible for controlling and managing other cognitive processes. They are a "macro construct" (Alvarez and Emory, 2006), in the sense that multiple sub-processes must work in conjunction to solve complex problems. The term "executive function" is therefore used as an umbrella for a wide range of cognitive processes and sub-processes (Elliott, 2003; Chan et al.,

2008).

Historically, a *central executive* was postulated by Baddeley and Hitch (1975) as being an attentional control system responsible for controlling and integrating a visual and a verbal slave systems. Since its inner workings were not known, it functioned mostly as a conceptual *homunculus*, a single entity responsible for all tasks still unexplained by the model. Baddeley (2002) himself agreed that this view was only a temporary solution, the central executive would function as a container for phenomena still not understood, but should be further decomposed into other sub-components.

In their nature, executive functions are mostly future directed and goal oriented, whilst exerting supervisory control over all voluntary activities. They deal with prospective actions and deliberate plans to achieve goals which can be defined by the executive itself. In a sense, this is what the frontal lobe, in particular the prefrontal cortex (PFC), does for humans (Baars and Gage, 2010; Chersi et al., 2011).

Important to notice however that, with the advent of new models for how the human brain works, a direct mapping between cognitive functions and brain structures is currently being questioned by neuroscience. In particular regarding the PFC, Fuster (2008) pointed out that no prefrontal area can be ascribed to a particular executive function. He claimed that executive functions operate over a system of cognitive networks (or 'Cognits'), spread along the cerebral cortex.

Cognits are formed by the connection of neural groups. These neural groups can be close or far apart in the brain. When these connected groups are associated to action selection and behaviors they are also called "executive networks", and extend across the frontal-cortex. Some executive cognits however represent sequences of actions which aim at reaching goals. These cognits have networks that extend themselves along the prefrontal-cortex. In other words, the prefrontal-cortex can be understood as a repository for executive networks called "cognits", which represent

past and future actions. The orderly activation of those cognits produces cognitive phenomena such as attention, working memory and planning (Fuster, 2008).

In a structural perspective, these neural groups are generally classified into three categories: hypercolumns, macrocolumns and microcolumns. The smallest unit of relevance here is the microcollumn, which is formed on average by 20 pyramidal neurons. Each microcollumn generally codifies similar information. In the prefrontal-cortex we can find groups of about 100 microcollumns, which were called "stripes" (Levitt et al., 1993). Each stripe is connected to other ten or more stripes, which in turn is called a "stripe cluster", as seen in Figure 2.1.

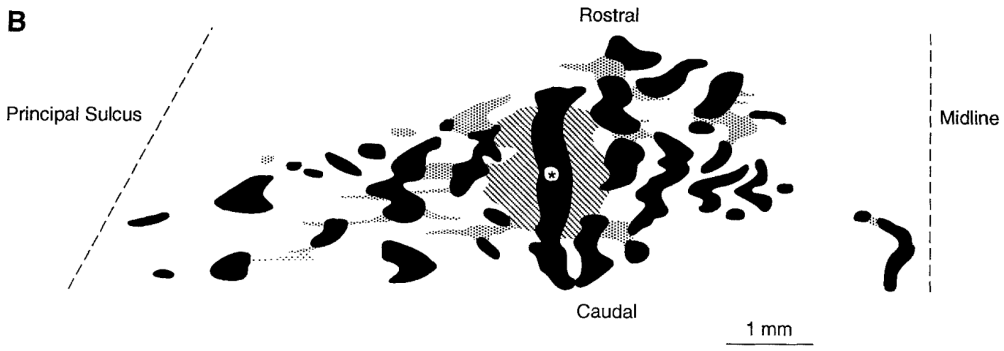



Figure 2.1: "Stripe *clusters*" in the prefrontal-cortex (Levitt et al., 1993).

According to Hazy et al. (2007), information in each stripe can be independently updated by a gating system, which is controlled by structures at the basal ganglia.Nonetheless, even if no prefrontal area can be ascribed to a particular executive function, the correlation between PFC and executive functions is so strong that it is useful to think about it as one of the major structures responsible for higher level cognition in humans. In fact, according to Fuster (2008), the PFC can be seen as a major holder of the aforementioned cognits.

In this context, we propose here a review of cardinal executive functions. The selection of those functions was based mostly on the works by Fuster (2008), Miyake et al. (2000) and Baddeley (2002), and should satisfy the demands of a central

executive whose neural correlates are strongly mediated by the human PFC.

We have adopted a separation between *low-level* and *high-level* executive functions as described by Miyake et al. (2000). Executive functions such as *monitoring*, *shifting* and *inhibition* are considered low-level when compared to *voluntary attention*, *planning* and *decision-making*. With the former three being directly related to the performance of the latter ones. This does not mean that they are the basic units of higher-level functions, but rather that they are simpler and easier to discretize by means of experimental analysis (Miyake et al., 2000). This taxonomy is summarized as seen in Table 2.1. The following sections briefly describe each of the aforementioned executive functions.

Table 2.1: Diagram with low-level and high-level executive functions.

| | | |
|---|---|---|
| Low level | Monitoring | Tests whether reality is as predicted.<br>Checks whether actions have expected results.<br>Evaluates whether plans are leading to goals. |
| | Shifting | Shifting between tasks or mental sets.<br>Also known as "mental flexibility" |
| | Inhibition | Inhibition of automatic responses.<br>Filters out what is not in the focus of attention.<br>Have deliberative and reactive components. |
| High level | Voluntary attention | Selects working memory content.<br><br>Perceptual search.<br>Goal oriented.<br>Avoids interference. |
| | Planning | Forms coherent behavioral sequences to achieve goals.<br>"Remembered future" |
| | Decision-making | Selects action, or course of actions, based on drives.<br>Three kinds:<br>Rational: deliberative, slow goal driven<br>Emotional: value-guided, fast<br>Reactive: stimuli driven, very fast |

## Monitoring

Monitoring is the executive function responsible for testing whether reality is as predicted. It does so by making sure the results of our actions, and their effects in the environment, are as expected. The nature of monitoring is feedback (neural reentry), and is vital for the individual to be able to follow plans and evaluate decision-making in order to achieve goals. Fuster (2008) concisely described the function of monitoring as the ability to answer three questions:

1. Was the action correct?
2. Was the action successful?
3. Did the action conflict with other actions/inner conditions?

Item (1) is important for the individual to know if the action intended to be done was actually performed. Without it, it would be impossible to learn a proper model of the world in early childhood, and correct new or malformed behaviors. Item (2) is vital for learning and adaptation and item (3) is responsible for detecting conflict. All items involve monitoring information, either coming from the senses or from memory, in order to revise the relevance of elements held in working memory (Miyake et al., 2000; Cooper, 2010).

## Shifting

Shifting, also known as 'mental set shifting' or 'context shifting', is the cognitive capacity to alternate back and forth between tasks or mental states (Miyake et al., 2000). This mechanism is vital for the individual to adapt to the unexpected, and makes possible higher level executive functions, such as executive attention and planning.

Mental shifting is important for individuals to succeed in a dynamical environment. Take the example of a creature that lives in a natural environment, such as in the woods. To survive, this creature might have a plan of actions to go from one place to another in order to look for food. If it notices a predator in its way, it must be able to abandon its previous plan and come up with a new one that includes this new contingency.

Alternating between different tasks or mental sets usually has the effect of a delayed response when compared to well rehearsed behaviors. This effect is also known as "switch cost" (Monsell, 2003). Being highly dependent on frontal lobes, shifting is sometimes referred to as 'mental flexibility', and has a major role in what we understand as creativity and originality.

## Inhibition

Prepotent responses are those associated to immediate reinforcement. In other words, they are reactive responses selected due to their immediate rewards, which can be either positive or negative. One of the most important low-level executive functions is the inhibition of these prepotent responses (Miyake et al., 2000). It is the capacity to deliberately inhibit dominant or automatic responses in order to achieve goals which require deliberation. This ability is crucial for all high-level executive functions, such as voluntary control, decision-making and planning (Fuster, 2008). Voluntary control depends on it to filter out irrelevant information, while decision-making needs inhibition in order to avoid undesired actions. In order to stick to a single plan of action, unsuitable plans must be inhibited from reaching the motor apparatus. Important to notice, however, that the inhibition mechanism mentioned in this section is limited to voluntary inhibition, and does not refer to neural inhibitions occurring in hierarchically lower structures of the brain.

### Voluntary attention

Also known as 'top-down attention' or 'executive attention', it is an executive function of major importance to high level cognition. It interacts with a 'bottom-up' attention system in order to select which stimuli are most important for the agent to be aware of, and should therefore be available in working memory. This bottom-up attention mechanism is directly related to stimuli coming from the senses. Each species has its own set of bottom-up attention mechanisms which react to very specific stimuli.

In human vision for instance, bottom-up attention reacts to particular characteristics such as lines orientation, brightness intensity, color changes and movement (Itti et al., 2000). However, the crucial distinction to make between bottom-up and top-down attention is that the former is mostly inherited by the individual, while the later must be learned and mastered in its lifetime. Learning what to keep in working memory is not an easy task. Our brains, more specifically our PFCs, must often find a way to relate actions we took in the distant past with rewards and consequences in the present (Hazy et al., 2007; Frank and Badre, 2012). The exact process which allows the brain to bridge this gap is one of the challenges faced by modern neuroscience.

The voluntary attention mechanism acts somewhat like a filter. It helps focusing our awareness in things which are of immediate interest to us, protecting us against potentially distracting irrelevant information. In this sense, it helps consciousness select material for working memory, which should hold limited information, in a temporarily accessible state, in service of cognition (Baddeley, 2002). Important to notice however that top-down attention acts not only on perceptual input, but also on long term memories relevant to the current situation. Under our control, it also serves as an inward directed voluntary attention system, directing attention to internal representations that must be integrated in our immediate memory (Fuster,

2008).

Take for instance the example of driving a car in a highway, while trying to find a particular traffic sign indicating where to turn right. Every time we pass a traffic sign by the road, that sign pops up into our visual bottom-up attention mechanism as something potentially relevant. That happens because traffic signs are designed to attract our attention regardless of what we are looking for. At the same time, our top-down attention mechanism is actively searching for the specific traffic sign representing right turn, which is described by an arrow pointing right. Were we not looking for the right turn sign, we might have missed it. It would not even register in our conscious mind. But since we were indeed looking for it, bottom-up and top-down attention acted in tandem, making us aware of it.

## Planning

The capacity to form coherent behavioral sequences toward the attainment of goals is called planning (Fuster, 2008). To plan a sequence of actions we must first create a mental representation of the current situation. We must then attend to goals that must be achieved and explore the possible actions which would get us closer to achieving them (Baars and Gage, 2010).

The ability to plan is found in animals with higher levels of cognition. It is present in a number of species such as corvids and cephalopods, but reached its maximum current potential in mammals like cetaceans and primates, being even more expressive in human beings. The exact reasons for such an expressive difference in human planning capability, when compared to other animals, are not completely understood. But it is widely believed that our highly developed PFC plays an important role in it.

Fuster (2008) proposed that goal-directed sequences of actions, in particular if these are novel or prospective plans, are formed by a system of cognitive networks,

distributed throughout the cortex. These cognits are present at the frontal lobes and extend into the PFC. They represent past and future actions, being both executive (operational) and also memory (representational) in nature. In this context, planning consists of "recalling" possible future outcomes for prospective actions. Our brain takes certain elements of prior experiences from long term memory, and reconfigure them in a way that makes it possible for us to achieve our goals. Ingvar (1984) called this property 'memory of the future', because we "remember" (or foresee) the possible future and its outcomes.

This is a paradigm shift in the way classical artificial intelligence usually approaches planning. In classical planning, an agent decides what to do by examining different possible sequences of actions that lead to states of defined value, and then chooses the best sequence. It works by using an algorithmic brute force approach, in a serial process called *search* (Russell and Norvig, 2010). However, despite the apparent seriality of our perceptual experience, our brains work in parallel. Our consciousness is like a serial software running in parallel hardware (Dennett, 1991). Therefore, simple *search* methods are not suitable to understanding how our brains perform planning.

Take the game of chess as an example[1]. When one first starts learning how to play chess, everything is mostly raw search through the space of possible movements: "If I move my rook two squares to the left, I'll threaten capturing my opponent's bishop, but at the same time it would weaken my king's defenses...". The deeper your search, the higher your chances of outperforming your opponent.

But as the player becomes more and more experienced, deep and slow searches are replaced to a large extent by parallel recall of similar board positions, how the player solved the dilemma in the past and what the outcome was. The player then

[1]In chess there are two players, each starting with a total of sixteen pieces. Each piece has its own starting position in the chessboard and also a particular behavior, defined by it being either a king, a queen, bishop, rook, knight or pawn. The end goal is to capture the opponent's king, and the players must plan how to move their pieces in order to achieve that ultimate goal.

uses these past memories to form new ones with good prospective outcomes, and in a fraction of a second he knows what the right move is.

## Decision-making

Decision-making differs from planning in the sense that it deals with choice rather than with deliberating a sequence of actions. Even so, the line separating planning and decision-making might become blurred by the fact that they work together most of the time. Fuster (2008), defined decision-making as the intent to execute an action, or course of actions, based on a drive[2]. Deciding on a discrete action or a plan of actions demands a minimum level of that drive.

Another way to classify human decision making is into three distinct, but complementary, components: reactive, emotional and rational decision-making.

Reactive decision-making can often be described as fast and automatic responses to stimuli. In its most basic form it is composed of phyletic memory (memories and knowledge available to the individual from birth, which were acquired by its species along millions of years of evolution), but can also be learned during an individual's lifetime through a process called "automatization". Emotional decision-making is located mostly in the ventromedial prefrontal cortex, with strong connections to the limbic system, and is responsible for value-guided decisions. Rational decision-making on the other hand takes part largely in the lateral prefrontal cortex. It deals with temporal integration, working memory and planning. Important to keep in mind though that this classification is somewhat artificial. There is no purely rational, reactive or emotional decisions being taken in our minds. They are mechanisms which are complementary in nature, and should be understood as such.

One particular model which tries to explain how these functions came into being was the triune brain concept proposed by MacLean (1990), which states that the

[2]We understand drives as being a set of deficits or needs which urge the individual to action in order to maintain its homeostatic regulation (Avila-Garcia and Cañamero, 2004).

brain developed into a three-layered organ composed of the reptilian layer, the paleomammalian layer and the neomammalian layer. The reptilian layer is composed of the oldest structures that dominate in the brains of snakes and lizards, with a major role on fixed, instinctive behaviors. The paleomammalian layer grew on top of the reptilian brain, it has a major role in emotions (emotional valence and salience) and is better at learning from experience. According to MacLean (1990), it was the development of the paleomammalian formation, also known as the limbic system, which made it possible for animals to experience emotions. In this context, emotions are the capacity to turn up or down the activation of drives that guide behavior for survival (Baars and Gage, 2010). Finally, the most recent layer is the neomammalian, which is the home of complex cognition, deliberative planning, social abilities and language. We explore in more detail Maclean's triune brain in Chapter 3, where we use it as a model to develop a bio-inspired cognitive architecture as a framework for the development of intelligent agents.

Within the scope of decision-making, it is possible to understand the interactions between limbic and neocortical layers with the old brain in terms of what LeDoux (1998) called a 'low road' and a 'high road'. The 'low road' bypasses the cerebral cortex, giving perceptual information directly to limbic structures such as the thalamus and the amygdala, prompting an emotional response. This 'shortcut' allows fast processing and reaction, in the expense of acuity. The 'high road', on the other hand, takes a longer path through the thalamo-cortico-amygdala connection, but is able to perform more complex and precise processing of stimuli, while subjecting them to deliberation before response (Baars and Gage, 2010).

## Consciousness and Executive Functions

Before investigating the question of how conscious processes relate to the aforementioned executive functions[3], we have to agree on a definition for consciousness.

Consciousness is an overloaded term, in the sense that it became synonym with a broad spectrum of phenomena not only in science but also in popular culture. Since we are interested in the former but not in the latter, we understand consciousness as the serial impression humans, and possibly other animals (Seth et al., 2005; Edelman et al., 2005), get from experiencing the world. The idea that we perceive the world as a serial succession of events might seem trivial at first. But that changes once we consider the fact that the brain is actually a parallel machine, and that this seriality must somehow emerge from the interaction of a multitude of parallel processes. As proposed by Dennett (1991), consciousness is "a Von-Neumannesque virtual machine implemented in the parallel architecture of a brain".

Details of how seriality emerges from the inherently parallel structure of the brain is still an open question and is beyond the scope of this work.

In this context, low-level executive functions play an important role in how consciousness works as a cognitive process. For instance, one of the major functions of consciousness is the identification of new situations in the environment. Our brain's monitoring system is constantly making predictions about how the world should be in the immediate future (Hawkins and Blakeslee, 2005), and when prediction and reality do not match (e.g., the brakes stop responding while driving a car) an alarm is issued, and the novel contingency reaches consciousness.

Consciousness is also deeply affected by mental set shifting and the neural mechanisms underlying it. As will be discussed latter, shifting attention away from a particular stimulus generally makes it fade from consciousness (Koch and Tsuchiya,

[3]In this work we assume that, although it exerts fundamental influence in executive cognition, consciousness is not itself considered to be an executive function (Tallon-Baudry, 2011).

2007). At last, the voluntary inhibition of conscious stimuli, behavior and memories work in tandem with our voluntary attention mechanism. By doing so, it keeps in the spotlight of attention only what is most important for the individual at a given moment.

Up until now we have reviewed the interaction between low-level executive functions and consciousness. It seems that these lower-level functions do not directly depend on consciousness to exist, at least not at an involuntary level. *Monitoring, inhibition and shifting do not seem to require consciousness in order to happen in the brain.*

Higher-level executive functions, however, are harder to picture without consciousness. Attention, for instance, is a concept often confused with consciousness, but they are actually different mechanisms. Part of the confusion comes from the fact that attention is believed to create access to consciousness (Baars, 1997), they influence each other in order to provide us with higher level cognition, but rely on distinct neural properties and have essentially different functions (Koch and Tsuchiya, 2007). Attention in this sense works as a gateway to conscious awareness. Besides working with attention in order to select what is most important at a given moment, consciousness also has a major role in our capacity to plan for the future. Its importance comes from the evolutionary advantage of being able to "simulate" future, potential actions, and learn new knowledge without the risks of actually performing those actions (Bargh and Morsella, 2008; Dehaene and Naccache, 2001). It is clear that the seriality of consciousness contributes to this mental simulation of prospective events, but details about the interplay between conscious planning and our unconscious mind are still being investigated.

Seth et al. (2005) stressed the fact that, whilst there are certainly many kinds of unconscious knowledge, what comes through consciousness may be particularly useful for volitional decision-making and planning. Many reflexive processes can be

performed without the need of conscious mediation. Spinal reflexes, or the enaction of well rehearsed behaviors can be largely unconscious. Even so, whilst unconsciously executed, reflexive behaviors can be directly influenced by conscious deliberation. Conscious decision making is important whenever novel contingencies arise and automatic responses are no longer sufficient for solving the problem at hand. Lastly, conscious decision-making influences the contents of consciousness itself. It does so by interacting with the top-down attention mechanism in order to decide what should be attended at the moment, given the individual's goals (Baars, 1997).

## 2.3 Discussion

Executive functions are brain functions responsible for controlling and managing other cognitive processes. They are prediction-based, future-oriented and are inextricably linked to the cognitive processes enabled by our prefrontal-cortex.

The models currently adopted to explain the major executive functions are shifting, monitoring, inhibition, voluntary attention, planning and decision making. In order to achieve high level cognition such as what we humans are used to experience, all these functions must be working together.

However, in this thesis I have focused on exploring the implementation of two of the high level executive functions into a biologically inspired cognitive architecture, namely planning and decision-making. As we will see in the next chapter, the proposed architecture also includes a road map for the integration not only of the remaining executive functions, but also for the implementation of a number of other cognitive mechanisms that should take it closer to presenting human-like cognition.

# Chapter 3

# A Triune Cognitive Architecture

"What I cannot create, I do not understand."

Richard Feynman

## 3.1 Introduction

The present chapter describes the development of a bio-inspired cognitive architecture, with different levels of cognition, targeting the control of artificial creatures and deliberative decision-making. As a standard guideline, cognitive neuroscience concepts were applied to define an incremental development for the cognitive architecture, following the evolutionary steps taken by the animal brain. The triune brain theory proposed by MacLean, together with Arrabale's "ConsScale" serve as road-maps to achieve each developmental stage, while iCub - a humanoid robot and its simulator - is used as an example application. The architecture's "Core" is fully described, and the first step at endowing it with executive functions is taken in the form of a behavior network implemented inside its "Central Executive" module. This behavior network is able to make a decision on which behavior is more relevant at a given situation, and it has the capacity of rudimentary planning by deliberating the most appropriate sequence of behaviors to achieve its goals.

The work described in the following sections lays the foundation for constructing a cognitive agent, which should in turn allow us to better understand cognition itself.

Looking for inspiration in nature has been a successful way of discovering new solutions for problems in the fields of control, optimization, classification and artificial intelligence. Our long term goal is to develop artificial creatures with different levels of machine cognition. To fulfil this goal, we propose the application of neuroscience concepts to incrementally develop a cognitive architecture following the evolutionary steps taken by the animal brain.

The motivation to propose and implement yet another cognitive architecture (considering that there are so many of them already available) lies in the requirement for an architecture coherent with our proposal of a codelet-based artificial mind, able to implement the animal brain in its different evolutionary steps along history. Our

proposition, in this chapter, is to investigate the results of natural selection through time, in order to derive a developmental feature set for the architecture. Most of what is presented in this chapter can also be seen in the works published by Raizer et al. (2011) and Raizer et al. (2012).

## 3.2 Statement and Background of Research

An artificial creature is an autonomous agent, a system embedded in an environment, sensing and acting on it, over time, in pursuit of its own agenda (Baars and Franklin, 2009). It can be controlled by a cognitive architecture, which includes aspects of the creature such as memory and functional processes (Langley et al., 2009), providing a framework to support mechanisms for perception, action, adaptation and motivation (Vernon et al., 2007).

As previously discussed, "consciousness" is not considered to be an executive function itself. However, since we intent this cognitive architecture to serve as a framework for the development of artificial agents with high levels of cognition, a roadmap for the development and evaluation of "machine consciousness" has been proposed.

Even though there is not a consensus on what exactly is meant by "machine consciousness" (as different authors indeed have different perspectives on what they mean by "consciousness"), in a previous work from our group (da Silva and Gudwin, 2010), we investigated a particularly interesting proposal which we called Baars-Franklin architecture[1], a general model developed by Stan Franklin's group on top of Bernard Baars's Global Workspace Theory of consciousness. During this investigation, we evaluated the possible benefits that such "consciousness" technology could bring while applied to the control of autonomous agents. According to our findings

[1]In the literature this is usually simply referred to as LIDA, or the LIDA model. We proposed the general name "Baars-Franklin Architecture" to refer not just to the actual LIDA model, but to the general architecture comprising LIDA and also all its predecessors: CMattie and IDA.

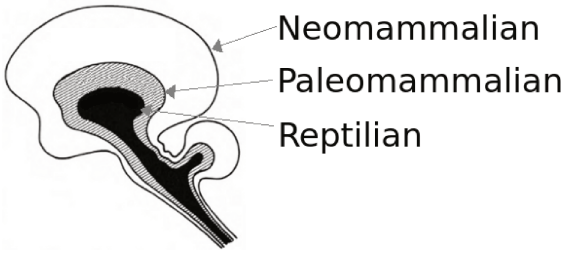



Figure 3.1: The triune brain model as proposed by MacLean. Source: MacLean (1990)

these are:

- The creation of an executive summary of perception (stream of consciousness)
- The possibility of behavior automatization (due to unconscious automatic behavior learning)

According to this perspective, consciousness is the emergence of a serial stream on top of a parallel set of interacting devices. In the Baars-Franklin architecture such devices are called "codelets", following Hofstadter and Mitchell (1994), which are small pieces of code specialized in performing simple tasks (Negatu and Franklin, 2002) in parallel. There are several kinds of codelets and they generally wait for interesting information until the situation is right for them to run.

Since cognitive architectures tend to model functions performed by structures of the animal brain, we must take into account that these structures have changed over millions of years of evolution. One model used to explain this process was the triune brain concept proposed by MacLean (1990), which states that the brain developed into a three-layered organ composed of the reptilian brain, the paleomammalian brain and the neomammalian brain, as can be seen in Figure 3.1.

The reptilian brain is composed of the oldest structures that dominate in the brains of snakes and lizards, with a major role on fixed, instinctual behaviors and control for survival. The paleomammalian brain is layered over the reptilian brain,

with a major role in emotions (emotional valence and salience) and is better at learning from experience. Finally, the most recent layer is the neomammalian brain, which is the home of complex cognition, deliberative planning, social abilities and language.

Recent authors consider this model to be an oversimplification of how human neuropsychology is structured (Smith, 2010). But however controversial this separation in three distinct layers might be today, it remains a helpful way to think about the mammalian brain (Baars and Gage, 2010), especially in a computational sense due to its modular approach. For instance, Ng et al. (2011) reported on work to apply MacLean's framework as a hierarchical structure for the construction of a cognitive architecture, which models the information processing in human brain and is able to handle various types of human-like knowledge to solve problems.

There is often no consensus about which brain structures compose each layer, and the functions each particular structure performs are not easy to discriminate due, among other reasons, to how massively interconnected most parts of the brain are. As exemplified by Fuster (2008), attributing a particular movement control to a given area, or speech only to Broca's area, ignores the fact that both functions depend on many other neural structures. With that in mind, this work aims at avoiding direct mappings (Rodrıguez et al., 2010) between neural structures and its functions, and focuses on a framework developed over a large body of brain and psychological evidence. The proposed architecture is based on Baars and Gage's functional framework, as seen in Figure 3.2, to develop a codelet-based, biologically plausible cognitive architecture.

In the framework from Figure 3.2 each sense has a brief storage ability, also called sensory buffer. Elements in the sensory buffer are modified by bottom-up selective attention, which happens to vision for instance when confronting particular patterns or to hearing when there is a loud noise. There is a top-down component

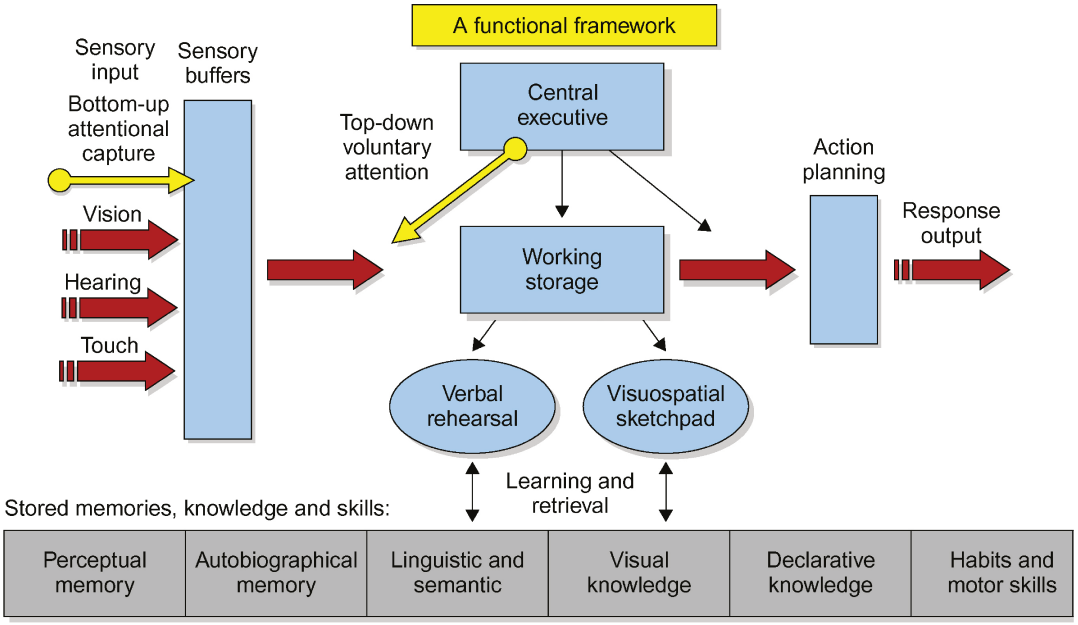



Figure 3.2: Baars and Gage's functional framework. Source: Baars and Gage (2010), with kind permission.

to selective attention coming from the central executive, which allows voluntary attention to happen. The central executive is part of working memory, as defined by Baddeley (2003), and it exerts supervisory control over all voluntary activities. Working storage is a short term and dynamic storage mechanism. It is composed of active populations of neurons which can consolidate into long term memories and is believed to have very limited capacity. The verbal rehearsal and the visuospatial sketchpad involve mental capacities used to remember things like new words (in the case of inner speech), faces, or spatial information (in the case of visuospatial sketchpad). They are both linked to long-term memory by a learning and retrieval mechanism. Long-term memory is represented by the gray boxes on the bottom and is comprised by a number of different types of memory, each with its own functions and characteristics. At the right side of the diagram, there is action planning, which can have both conscious and unconscious components, and finally an output response

that closes the perception-action cycle.

## 3.3 Architecture proposal

The work is following a path similar to the evolutionary steps taken by the animal brain as stated by MacLean (1990). The hypothesis is that such an approach should guarantee a grounded intelligent system at each developmental phase, while biasing the system towards high-level animal intelligence. [2]

The initial development of this architecture puts the emphasis on behavioral results. A low level approach to evaluate its consciousness levels - such as information integration, as seen in the works of Tononi (2008) - might be implemented in future works. ConsScale, a biologically inspired scale designed to evaluate the presence of cognitive functions associated with consciousness (Arrabales et al., 2010b), will be used to assess the different levels of control implemented within this cognitive architecture, as was described in more detail by Raizer et al. (2011).

ConsScale is a framework proposed by Arrabales et al. (2012), which is meant to be used as a roadmap for the development of artificial agents and their evaluation in terms of cognitive capacity. It is based on higher level functional aspects of the system such as behavioral capacities and skills. It was originally designed to provide a measure of the level of global cognitive power (Arrabales et al., 2009b) achieved by a certain artificial agent, and defines a quantifiable way of evaluating it. There are a total of 13 levels in ConsScale (Arrabales et al., 2010a): -1 Disembodied, 0 Isolated, 1 Decontrolled, 2 Reactive, 3 Adaptive, 4 Attentional, 5 Executive, 6 Emotional, 7 Self-Conscious, 8 Empathic, 9 Social, 10 Human-Like, 11 Super-Conscious. Levels -1, 0 and 1 correspond to agents which are neither embodied nor situated, and are often

[2] It is important to acknowledge, however, that the path taken by mammals in evolution, especially the case of homo sapiens, is not the only one which led to high-level cognition. Examples of high level cognitive behavior, and potentially conscious capabilities, have been observed in modern birds and cephalopods (Edelman et al., 2005).

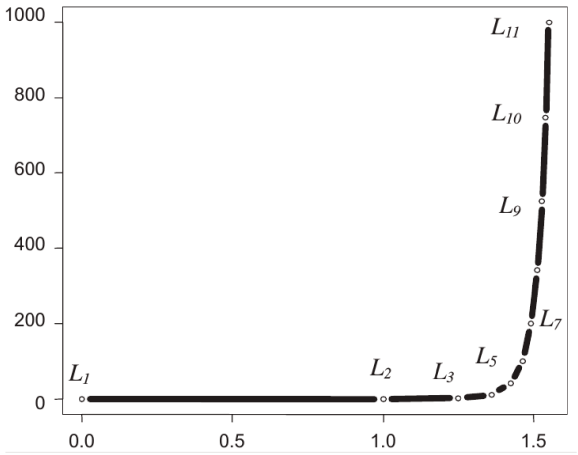



Figure 3.3: ConsScale quantitative score levels as a function of cumulative level score. Adapted from Arrabales et al. (2010b).

disregarded. The lowest level of a situated and embodied agent starts at level 2, which defines a classical reactive agent, without explicit memory or learning capabilities. At each level there is a list of cognitive skills (CS) that must be present in the agent and which must also be validated by behavioral profiles (BP). A Cumulative Level Score (CLS) is calculated as new CSs are added and validated with their corresponding BPs. This CLS is then used to calculate the agent's ConsScale Quantitative Score (CQS), which places the given agent at a particular point in the exponential scale seen in Figure 3.3 [3].

Examples of cognitive skills and behavioral profiles will be presented in the following sections.

In our work (Raizer et al., 2012), the platform used as an specific domain to exemplify the formulation of experiments is the iCub humanoid robot simulator (Metta et al., 2008), but the architecture is built so it can be applied to different platforms and applications. This particular platform was chosen at this point because it provides a "Human-Like" architectural level, as described by Arrabales et al. (2010b).

[3]See Arrabales et al. (2010b), Arrabales et al. (2009b) or Arrabales et al. (2010a) for more details on how to calculate CLS and CQS for a particular agent.

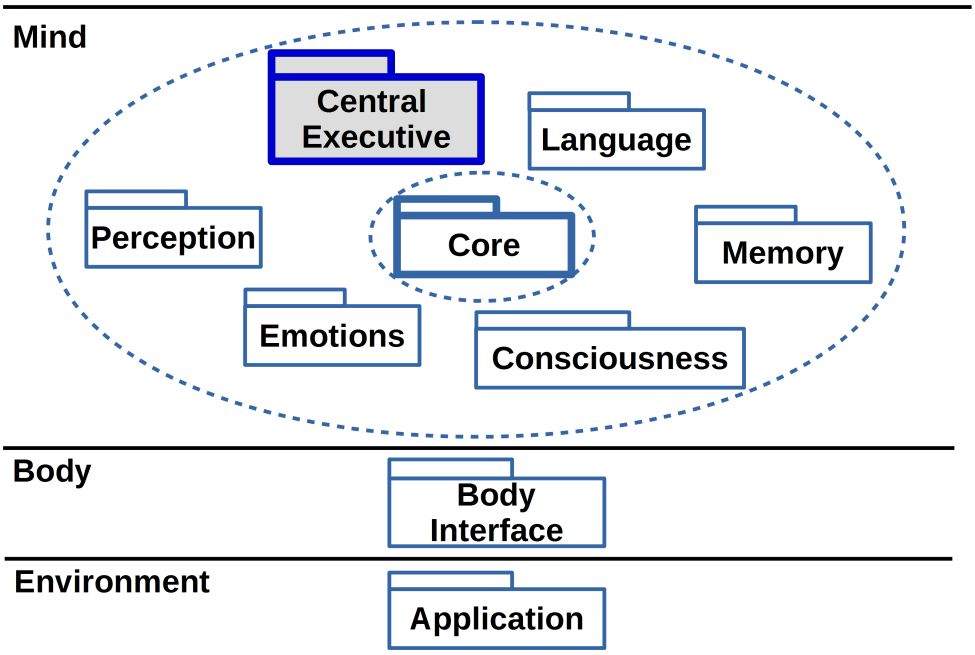



Figure 3.4: Architecture's layers and subsystems

### 3.3.1 Conscious Codelet-Based Cognitive Architecture

Figure 3.4 shows a diagram describing the proposed architecture's modules.

The relationship between modules and their features according to the functional framework of Figure 3.2 is better understood by following a full cognitive cycle, considering the neomammalian brain:

1. Bottom-up attention (Perception) acts on sensory buffer (BodyInterface), giving rise to objects from raw sensory inputs;

2. Bottom-up attention (Language) acts on sensory buffer (Perception), giving rise to symbols from objects;

3. Top-down attention (Central Executive) acts on sensory buffer's objects (Perception) and symbols (Language);

4. Top-down attention (Central Executive) brings information into working storage (Memory);

5. Learning and retrieval mechanism (Memory) consolidates to stored memory (Memory) and brings into working storage (Memory) long-term information;

6. Spotlight controller (Consciousness) acts on working storage (Memory), defining spotlight (Consciousness) content;

7. Decision-making (Central Executive) uses information under spotlight (Consciousness) to select a plan, composed of a list of behaviors;

8. Behavior sequence is sent to action buffer (BodyInterface);

9. Actuators (BodyInterface) act on the World (iCub) based on action buffer (BodyInterface).

Figure 3.5 shows a diagram depicting how the concepts of the codelet-based Core subsystem have been implemented. Following this picture, Memory Objects are single units of data in memory, which have a type (T) and some information (I). The Raw Memory contains all Memory Objects in the system. It can be logically divided in different kinds of memory, such as the stored memories from Figure 3.2. Codelets are devices which are composed of small pieces of code, specialized in performing simple tasks (proc), a list of input Memory Objects (In), the ones which are read, a list of output Memory Objects (Out), the ones which are written, an input list of broadcasted Memory Objects (B), the ones which were broadcasted by consciousness mechanisms, and an activation level (A). Coalitions are groups of Codelets which are gathered in order to perform a task by summing up their abilities. Two or more Codelets share a Coalition when they write in and/or read from the same set of Memory Objects. The Coderack, according to Hofstadter and Mitchell (1994), is the pool of all active Codelets in the system.

With the above described Core at hands, the next module to go under development was the Central Executive, as it holds most executive functions and performs a

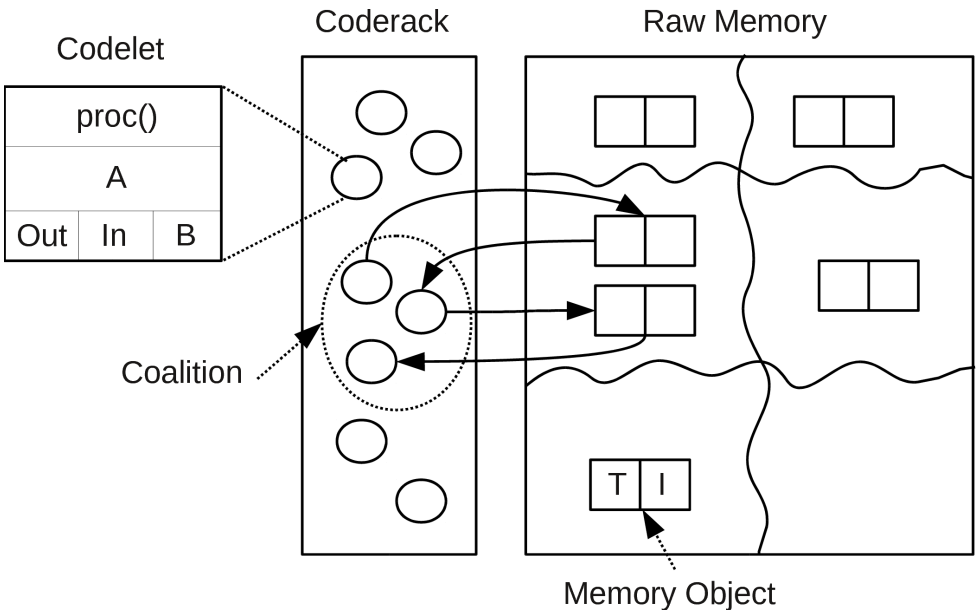



Figure 3.5: Core's concepts

central role both in Baars and Gage's diagram and also in this architecture. As defined by Baddeley (2003), some of the central executive's responsibilities are exerting supervisory control over all voluntary activities and allowing deliberative decision-making.

Focusing on deliberative decision-making and inspired by previous successful works (Negatu and Franklin, 2002), a modified behavior network based on the original action selection mechanism by Maes (1989) has been implemented. As pointed out by Tyrrell (1994), the original behavior network proposed by Maes suffers from a number of drawbacks, such as not being able to perform more than one action at the same time, using Boolean propositions instead of continuous variables or oscillating between behaviors. The last one has been addressed in the current implementation, while other issues will be taken care of during development, since parallel works have shown it to be a possible endeavor (Negatu and Franklin, 2002; Dorer, 2010).

Figure 3.6 depicts the concepts of the Central Executive subsystem as it has been implemented so far. Permanent goals are defined by the agent's most vital

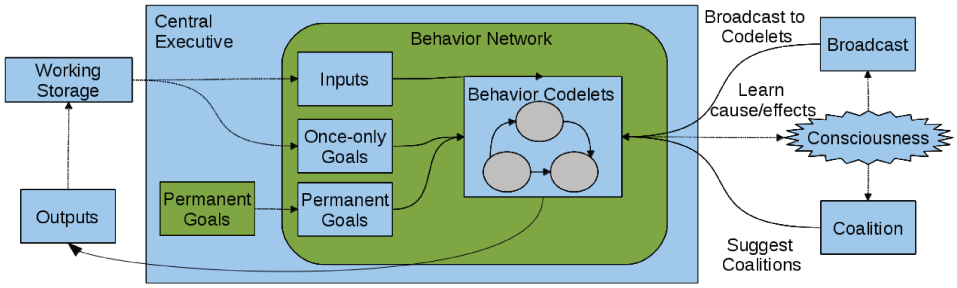



Figure 3.6: Deliberative decision-making in detail, and its relationship with consciousness mechanism

objectives and motivations. One time only goals - goals that are expected to be achieved only once - are defined depending on the interaction between the agent and its environment, as well as on its own inner state.

In this implementation, each behavior is a codelet and as such runs as an independent process. The whole network is therefore a distributed system, in which codelets communicate with each other exchanging energy, as described by Maes (1989), only when belonging to the same coalition.

Coalitions of codelets are formed by the consciousness module based on context, with a given context being defined by the world state, state of self, and each behavior's preconditions and prospective consequences.

More details on the inner working of the implemented Behavior Network will be seen in Chapter 4, where we apply the mechanism to a robotic application. The other modules are subject of future work and will be implemented according to the planned steps for the architecture, as further explained in Section 3.3.2.

### 3.3.2 Evolutionary Steps Taken by the Animal Brain

#### Reptilian

According to MacLean, the protoreptilian formation is composed of a group of ganglionic structures located at the base of the forebrain of reptiles, birds and mam-

mals. It is also known as the striatal complex and brainstem (Baars and Gage, 2010) or, as he puts it, the R-Complex. It was traditionally thought to be the motor apparatus under control of motor cortex and reveals a number of fixed behaviors (25 special forms of behaviors and 6 forms of what he calls "interoperative" behaviors (MacLean, 1990)), involved in the regulation of the animal's daily routines. In this sense, the R-Complex is composed of pre-programmed regulators for homeostasis and survival, lacking an advanced learning mechanism as the one seen latter in evolution. On the one hand, it excels at performing sensory categorization, such as identifying a particular smell as being harmful or not, and then generating reflexive messages about what to do, like running or biting (Goleman, 2006). On the other hand, it is constrained by its daily master routine, destined to perform a limited number of behaviors. Reptiles and lizards however need to "learn" their territory in order to know which hole to escape into in case of a predator or just to find their way home. This, and other examples of very basic "memory/learning" mechanisms, is what MacLean called protomentation, which are "rudimentary mental processes that underlie a meaningful sequential expression of prototypical patterns of behavior" (MacLean, 1990).

Based on the main characteristics of a creature with an R- Complex, a number of skills from ConsScale[4] are proposed for this level of development:

- CS2-1 : Fixed reactive responses.

  BP2-1 : Basic reflexes such as blinking and contraction of limbs as responses to pain.

- CS3-3-5 : Selection of relevant sensory/memory/motor information.

[4]Here, CS stands for "cognitive skill", which is the definition of the cognitive capacity that must be implemented. BP stands for "behavioral profile", which is an experimental implementation of the equivalent cognitive skill. Each CS must, therefore, be validated by its corresponding BP.

BP3-3-5 : The robot reacts to predefined sensory inputs and stores basic information in fixed memory.

- CS3-6 : Evaluation (positive or negative) of selected objects or events.

  BP3-6 : The robot evaluates sensory input comparing it with predefined patterns to evaluate sensory inputs as being good or bad for it.

- CS4-2 : Directed behavior toward specific targets like following or escape.

  BP4-2 : The robot selects grabbing action towards regions that seem good for it.

According to ConsScale, the aforementioned selection of skills constitutes a level 2 (Reactive) agent, with a CQS (ConsScale Quantitative Score) of 0.21 in a scale from 0 to 1000. This set of skills motivated the creation of a Body Interface module, responsible for dealing with all somato-sensory data exchange - sensory input and response output from Figure 3.2 - between agent and environment. This module also holds a fixed number of reactive responses and autonomic behaviors that have as a primary concern the survival and self-preservation of the agent. Perception at this level should be very basic, with low resolution pattern recognition and a bottom-up attention mechanism that depends on the agent's nature and objectives. The output functions are organized in the Central Executive module, which at this level of development is responsible for decision-making with a repertory of fixed predefined behaviors. The agent at this level lacks a general Memory System, counting only on predefined memory slots for performing specific tasks.

There is a great debate on whether creatures other than humans do or do not possess consciousness as we experience it. One major problem faced by such debate is the lack of an accurate definition of what consciousness is and what is needed for its emergence. In this work, machine consciousness is the implementation of

Global Workspace (GW) theory (Baars and Franklin, 2009) as a mean to achieve primary consciousness, in which percepts are united into episodic scenes (Edelman, 2004). In this sense, it is assumed here that a protoreptilian brain lacks the reentrant interactions in the thalamocortical system needed to sustain consciousness.

### Paleomammalian

The next evolutionary step in the development of the vertebrate brain is the paleomammalian formation. With this new set of neuronal structures - some notable examples being the amygdala, hippocampus and hypothalamus - also known as the limbic system, animals became able to experience emotions (Baars and Gage, 2010), which are essentially the capacity of turning up or down the activation of drives, with one of its roles being to guide behavior for survival. This emotional "skill" greatly affects and communicates with the aforementioned autonomic system, provoking marked physiological changes within the organism. Learning and memory have also shown remarkable improvement. An animal with a limbic system mounted on top of its R-Complex was able to discriminate good things from bad ones also by looking into its past memories (Goleman, 2006). MacLean emphasized that this part of the mammalian brain is responsible for a number of behaviors and characteristics that were absent in ancient reptiles such as nursing, audio-vocal communication for maternal contact and play (MacLean, 1990).

The ConsScale skills added at this level are listed as follows:

- CS3-1 : Autonomous acquisition of new adaptive reactive responses.

  BP3-1 : Learns to "eat" certain kinds of "food" and reject others.

- CS3-2 : Usage of proprioceptive sensing for embodied adaptive responses.

  BP3-2 : Looks for "food" when hunger state reaches a certain level. Plays to get happier.

- CS3-7 : Selection of what needs to be stored in memory.

  BP3-7 : Emotional valence influences selection of what is relevant to be stored in memory.

- CS4-1 : Trial and error learning. Re-evaluation of selected objects or events.

  BP4-1 : The robot learns what is good (good food) or bad (rotten food) for him by trial and error.

- CS4-3 : Evaluation of the performance in the achievement of a single goal.

  BP4-3 : Evaluates how successful it is in pursuing a single goal, such as looking for "food" and uses this information to get better at it.

- CS4-5 : Ability to build depictive representations of percepts for each available sensory modality.

  BP4-5 : The robot can discern particular objects and some of its properties and calculate their relative positions.

- CS5-1 : Ability to move back and forth between multiple tasks.

  BP5-1 : An interrupted behavior is resumed latter if still relevant. For example: playing with certain objects in the environment can be resumed after stopping this behavior to satiate hunger.

- CS5-4 : Autonomous reinforcement learning (emotional learning).

  BP5-4 : The robot calculates a "reward" based on how good it is at a task and improves its performance. It might throw a ball at a given target and get better at it by practicing.

- CS5-2 : Seeking of multiple goals.

BP5-2 : Having more than one goal, such as satisfying hunger and play, it uses CS5-1 to alternate between them.

- CS5-3 : Evaluation of the performance in the achievement of multiple goals.

  BP5-3 : The robot evaluates its own performance at pursuing multiple goals, and alternating among them, instead of pursuing only one.

- CS5-6 : Ability to generate selected mental content with grounded meaning integrating different modalities into differentiated explicit percepts.

  BP5-6 : The contents of the conscious broadcast, defined by the consciousness module, constitute mental content with grounded meaning and it is composed of an integration of percepts from different modalities.

- CS6-1 : Self-status assessment (background emotions).

- BP6-1 : Evaluates its own inner physical and emotional state and has its global behavior influenced by it.

- CS6-2 : Background emotions cause effects in agent's body.

- BP6-2 : The emotional state is reflected into the robot's body (happy or sad faces) through its autonomic functions.

- CS6-3 : Representation of the effect of emotions in organism and planning (feelings).

- BP6-3 : Together with BP6-1, if the robot is high on health and hungry it may go look for food but if low on health and hungry it might hide at home.

This set of skills appears at the ConsScale as a level 3 (adaptive) agent, with a CQS of 7.21 in a scale from 0 to 1000. Even having a number of higher skills, such

as CS6-1 (Self-status assessment) for instance, it lacks some dependencies such as CS4-4 (Basic planning capability) that would allow it to attain a higher score.

There is an evolution in perception at this stage so the agent is able to perform higher-level pattern recognition, and discern particular objects in the environment. The central executive performs top-down attention over percepts, providing full attention selection capability. The memory module becomes generic, in the sense that memory objects are produced, stored and retrieved by means of a learning/retrieval mechanism. Those memories and percepts are marked with emotional content, influencing the aforementioned learning/retrieval mechanism.

At this point it is assumed that the reentrant interactions between parts of the thalamocortical system mediating perceptual categorization and those mediating memory have evolved in a way that allows the emergence of primary consciousness. The consciousness module implements GW theory, producing an attentional spotlight that broadcasts its contents to all the system. However, the creature still lacks the capacity to report its conscious stream, an ability human beings possess and which is used in our case to verify the existence of consciousness as we perceive it.

### Neomammalian

The neomammalian formation is the latest addition to the vertebrate animal brain. Its distinguished structure is the neocortex which is composed of many layers, with a smooth surface in small mammals and deeply grooved in larger ones. The neocortex is highly oriented toward the external world (MacLean, 1990). With it, animals are capable not only to understand their senses but also to develop a symbolic representation of those senses and inner representations. Being on top of the limbic system, it is also capable of developing feelings about these symbols and abstract ideas (Goleman, 2006). Its most distinctive role, however, lies in what is called

executive functions, which consist, among other things, on the ability to organize sequences of actions towards a given goal.

As the vertebrate brain evolved, the organism's actions became more based on its memories and prior experiences than on reflexive responses to the environment based on its needs (as can be seen in the transition from protoreptilian to paleomammalian). These actions also became more deliberate and voluntary (Fuster, 2008), especially in the transition from the paleomammalian to the neomammalian brain. With this evolution, important parts of the neocortex such as the prefrontal cortex show significant growth in proportion to more ancient brain structures, with a maximum size achieved only in the human primate (Fuster, 2008).

The ConsScale skills at the neomammalian level are as follows:

- CS4-4 : Basic planning capability: calculation of next n sequential actions.

  BP4-4 : Plans a sequence of actions to attain a goal. Example: playing for learning wastes energy, so it plans a break time to replenish it.

- CS5-5 : Advanced planning capability considering all active goals.

  BP5-5 : Takes into account information from CS5-3 to improve seeking multiple goals. Such as reducing transition time between behaviors or deciding a better order of behaviors.

- CS6-4 : Ability to hold a precise and updated map of body schema.

  BP6-4 : It has a map of its own body and can use it to plan/select behaviors.

- CS6-5 : Abstract learning (lessons learned generalization).

  BP6-5 : Its memories influences how it behaves in a general way. Differently from how it would behave without them.

- CS6-6 : Ability to represent a flow of integrated percepts including self-status.

BP6-6 : The consciousness module allows an executive summary, composed of integrated percepts and allowing the robot to represent its self-status.

- CS7-1-3 : Representation of the relation between self and perception / action / feelings.

  BP7-1-3 : Special codelets are specialized in establishing the relation between perception and action, and the robot's sense of self as an emotional agent.

- CS7-4 : Self-recognition capability.

  BP7-4 : The robot recognizes itself as an agent in the world, allowing CS7-5.

- CS7-5 : Advanced planning including the self as an actor in the plans.

  BP7-5 : It performs CS5-5 taking into account itself as an agent.

- CS7-6 : Use of imaginational states in planning.

  BP7-6 : The robot estimates future emotional state for possible outcomes due to planned actions and uses this information to select behavior.

- CS7-7 : Learning of tool usage.

  BP7-7 : Learns to use objects in the scene to perform tasks, such as throwing a ball at something out of reach to bring it down.

- CS7-8 : Ability to represent and self-report mental content (continuous inner flow of percepts/inner imagery).

  BP7-8 : The robot can report its conscious contents.

- CS8-1 : Ability to model others as subjective selves.

  BP8-1 : It will use its own mental model to predict/estimate another's actions.

- CS8-2 : Learning by imitation of a counterpart.

  BP8-2 : The robot will learn new behaviors, such as waving or select particular objects, by watching a counterpart doing it.

- CS8-3 : Ability to collaborate with others in the pursuit of a common goal.

  BP8-3 : The robot can form plans including other agents to reach a common goal. Such as pushing boulders or exchanging tools.

- CS9-3 : Advanced communication skills (accurate report of mental content as basic inner speech).

  BP9-3 : The robot is able to report inner mental state.

A neomammalian agent is registered as being level 7 (self- conscious) and scores 207.63 at the CQS scale.

The central executive at this stage is able to produce new behaviors that are added to the repertory of predefined ones. It becomes able to actually devise plans to achieve its goals. The major add on feature in perception is the creation of memory objects with symbolic content.

An agent at the neomammalian level should has a Language module which is responsible for producing an accurate report of its mental content. This allows basic reportability tests of consciousness but high-order consciousness (Edelman, 2004) should only be achieved at the *homo sapiens* stage.

***Homo sapiens*** The most distinguished part of the human brain is its big frontal lobes. These regions have shown remarkable expansion at the last stage of human evolution and can be regarded as the core machinery for what we understand as being human. The aforementioned prefrontal cortex (PFC) plays a decisive role in both social cognition and advanced planning and problem solving. The ability to

recombine and manipulate internal representations, a vital skill for the development of advanced language, and the capacity of holding "images of the future", important for tool-making, are both critically dependent on the PFC (Baars and Gage, 2010).

The ConsScale skills at the *homo sapiens* level are as follows:

- CS8-4 : Social planning (planning with socially aware plans).

  BP8-4 : The robot devises plans including groups of agents in order to improve the group's conditions as a whole.

- CS8-5 : Ability to make new tools.

  BP8-5 : The robot can combine objects in the scene to produce a new tool. For instance, bending a wire so it works as a hook.

- CS8-6 : Inner imagery is enriched with mental content related to the model of others and the relation between the self and other selves.

  BP8-6 : Robot's conscious content integrates mental imagery related to its own model and the models of other agents.

- CS9-1 : Ability to develop Machiavellian strategies like lying and cunning.

  BP9-1 : The robot is able to estimate another agent's reaction to its actions and use it in its own benefit. For instance, if the robot wants a person to get closer, it might ask this person for "food" even without being hungry.

- CS9-2 : Social learning (learning of new Machiavellian strategies).

  BP9-2 : The robot can learn new strategies as in CS9-2, not implemented a priori.

- CS9-4 : Groups are able to develop a culture.

BP9-4 : Groups of robots and other agents can develop their own cultural content and pass it on to other individuals to improve learning.

- CS9-5 : Ability to modify and adapt the environment to agent's needs.

  BP9-5 : The robot can include the altering of the environment in its plans to reach a goal. Such as moving rocks to form a barrier.

- CS10-1 : Accurate verbal report. Advanced linguistic capabilities. Human-like inner speech.

  BP10-1 : The robot should be able to develop conversations, with grammar and semantic content.

- CS10-2 : Ability to pass the Turing test.

  BP10-2 : At this point, the robot should be able to pass a domain specific Turing test.

- CS10-3 : Groups are able to develop a civilization andadvance culture and technology.

  BP10-3 : Groups of robots and other agents should be able to interact in a cultural and social way to develop new tools and knowledge about the environment.

At this stage, the agent reaches level 10 (Human-Like) in the ConsScale, with a CQS of 745.74. Higher levels could only be achieved with structural modifications in the basic architecture to allow several streams of consciousness being managed by the same agent. It is not clear, however, if being able to manage many streams of consciousness in the same body would result in an advantage, as suggested by ConsScale.

Structurally, the cognitive architecture at the *homo sapiens* level is the same as the one described for the neomammalian brain. It has the same modules and the communication between them is virtually identical. The difference lies in the codelets used to perform the new skills necessary to achieve this level of cognition.

A general picture of all levels discussed so far can be seen in Figure 3.7.

The figure shows a mapping between conscale, proposed by Arrabales et al. (2010b) and, our triune development framework. Each level of the architecture is defined by a particular color, which in turn encompasses a particular set of cognitive capabilities.

## 3.4 Discussion

The architecture proposed here, in its many development stages, aims at managing the agent's attentional resources in order to fulfill its tasks and reach its goals. Distinctively from other architectures, this model commits to a single, uniform notation for encoding knowledge, which are memory objects that hold information for different applications. This has the advantage of simplicity and may support learning and reflection more easily, since they have to operate on a single type of structure.

The codelet approach further enhances the modularity and scalability of the system. Particular codelets can be designed on demand to fulfill a given task and be readily implemented in the architecture without the need of major architectural modifications.

Due to its essentially modular structure, like seen in ACT-R and LIDA, this triune cognitive architecture differs from other well known architectures such as SOAR (Langley et al., 2009). A modular structure offers a number of advantages, such as robustness and allowing distributed processing. Moreover, ACT-R and SOAR architectures lack, at the time of this writing, a consciousness mechanism, which would allow an efficient perceptual summary and behavior automation.

In summary, this chapter has described the framework for a cognitive architecture which provides a road-map for the development of intelligent agents. This road-map served us as a guide to explore the executive functions proposed in Chapter 2, and should also be the foundation for our future research with cognitive architectures in general. In the following chapter we describe in more detail the implementation of the behavior network for deliberative decision-making, and present its validation in a simulated robotic application.

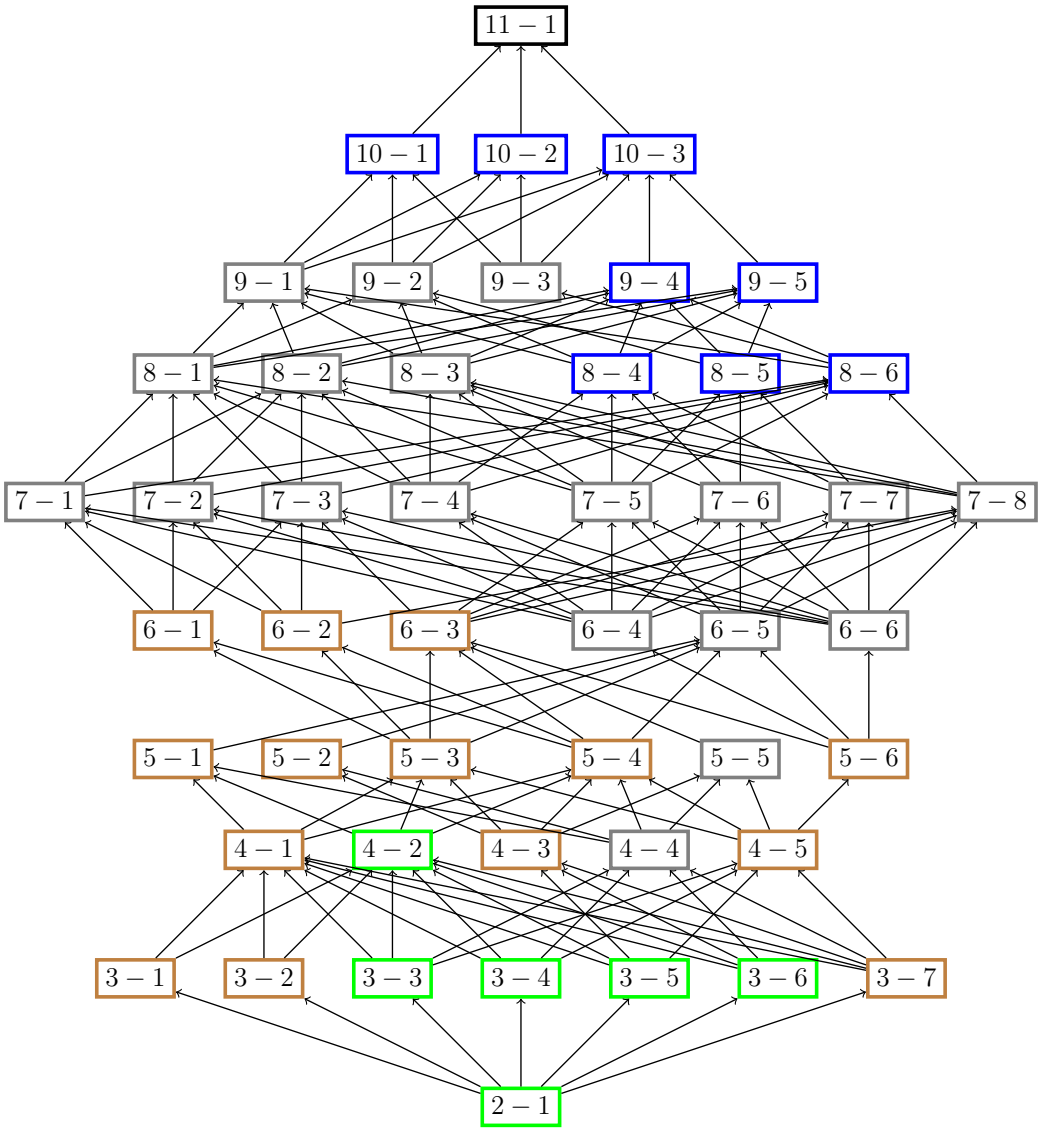



Figure 3.7: Mapping between conscale's CSs and the proposed triune cognitive architecture at different developmental levels. Each numbered box is a ConsScale cognitive skill, and arrows suggest dependency between skills, as proposed by Arrabales et al. (2010a). Green is for reptilian, brown is for paleomammalian, grey is for neomammalian, blue is for human-like and black is for super-human.

# Chapter 4

# A Soft-Preconditions Behavior Network for Deliberative Decision making

"No sensible decision can be made any longer without taking into account not only the world as it is, but the world as it will be..."

Isaac Asimov

## 4.1 Introduction

In Chapter 3 we introduced a road-map for implementing an intelligent agent, starting at lower level cognitive components and following vertebrate evolutionary development toward high level cognition. To tackle the problem of embedding it with executive functions as seen in Chapter 2 - in particular deliberative decision-making - we proposed the development of a behavior network, modified in order to work with a bio-inspired cognitive architecture.

This chapter provides further details about the proposed behavior network, and validate its implementation by applying it to a simulated robotic application. This work was developed in cooperation with the Brazilian project DesTINe (Desenvolvimento de Tecnologias da Informação para Neurologia), for assistive technologies. The chosen application was the development of an intelligent agent responsible for making interesting action suggestions to a BCI (Brain Computer Interface) user in the context of an intelligent environment. Our hypothesis is that an artificial agent capable of making decisions closer to how humans do it, should be able to make interesting suggestions to the human user.

The work presented here describes the development of an intelligent agent responsible for making interesting action suggestions to a BCI user in the context of an intelligent environment. In order to perform its duties as a suggestion system, the artificial agent should "put itself in the user's place", and choose the most desirable action given a set of goals and current world state.

A modified version of a behavior network, which was originally proposed by Maes (1989), was used to define which action suggestion took precedence over many others in a complex environment. This modification consisted on adding an extra list of preconditions, hereby called Soft-Preconditions, which work like the original preconditions list from MASM (Maes Action Selection Mechanism), but do not block a

given behavior from becoming executable. We address this modified version of the original MASM as "spBN", or soft-preconditions behavior network.

Granting mobility to victims of severe motor impairment is a major challenge being faced by a number of research groups throughout the world. The low bandwidth offered by non-invasive Brain Computer Interfaces (BCI) makes their use in complex systems difficult. One solution to this problem is to increase the autonomy of the agent responsible for executing those BCI commands, allowing the user to perform higher level control in certain situations (Bell et al., 2008).

A smart environment is a controlled environment enhanced with actuators and sensors, being able to provide services assisting the human operator and the robotic agent in their tasks. Services to the human operator can be to interact with the environment (e.g., opening doors, calling an elevator, changing the TV channel or air conditioning setting, etc.), or to provide information like reminding scheduled events, showing the temperature in a room, showing the crowding of a remote place, showing an optimal way toward a goal etc. For the robotic agents, useful services can be localization, action/path planning (Rohmer et al., 2010), and so on.

Even with a smart environment, the operator still needs to interact with the system to let it know what he wants to achieve, using a BCI to navigate through menus inside a Graphical User Interface (GUI). Each interaction with the system increases mental workload and eventually leaves the operator tired (Cowan et al., 2012).

In order to soften this drawback and increase user comfort, a possible solution is to reduce the number of interactions with the system which are needed in order for the user to express his will to the system. One way of achieving this is to cleverly organize access to menu options, therefore limiting the total number of interactions. Another known way is trying to guess the operator's will by taking into account his current state and goals. In this work, we suggest the idea of using computational

intelligence techniques, in the form of the aforementioned artificial agent, to provide a suggestion system to the user. Most of what is presented in this chapter can also be seen in the work published by Raizer, Rohmer, Paraense and Gudwin (2013a).

## 4.2 Behavior Networks

Also known as MASM (Maes' Action Selection Mechanism), a behavior network is an action selection mechanism that aims at selecting the most appropriate action at a given instant of time. In the original paper (Maes, 1989) behaviors are also addressed as "competencies", which are mutually exclusive, in the sense that only one of them can be active at a given time.

A behavior module, or competence, $i$ is described by a tuple, as described in Expression 4.1.

$$B_i = (c_i, a_i, d_i, \alpha_i) \tag{4.1}$$

In other words, if a given behavior network has three behaviors, $B_1$ or $B_{i=1}$ is the first behavior, $B_2$ or $B_{i=2}$ is the second behavior and $B_3$ or $B_{i=3}$ is the third behavior.

Where $c_i$ is a list of preconditions which must be fulfilled (observed to be true in the world) before the behavior can become active. The expected consequences for this behavior to get executed are listed in $a_i$ and $d_i$. Here, $a_i$ is known as the add list, which is a list of propositions that are expected to become true after the given behavior gets executed, and $d_i$ is known as a delete list, which holds a list of propositions that are expected to become false when the behavior gets executed. Finally, $\alpha_i$ is the activation level for behavior $i$. It is a real value number which represents how relevant a behavior is at a given time.

In other words $\alpha_1$, or $\alpha_{i=1}$, is the level of activation for behavior $B_1$, and represents

how relevant it is when compared to other behaviors.

At this point, it is also important to keep in mind that in the original MASM a behavior can only become executable if, and only if, all of its preconditions are observed to be true at a time $t$. Also, there can be only one behavior active at any given time $t$.

Given the aforementioned definition, behavior modules are, therefore, linked in a network through three types of links: successor links, predecessor links, and conflicter links:

- There is a successor link for every proposition in the add list of behavior x that also exists in the precondition list of behavior y. In other words, there is a successor link from behavior x to behavior y, if the consequences of behavior x make it more likely for behavior y to become executable.

- There is a predecessor link from behavior y to behavior x for every successor link between x and y. In other words, there is a predecessor link for every successor link between two behaviors.

- There is a conflicter link from behavior x to behavior y, for every proposition that is a member of the delete list of x and a member of the precondition list of y. In other words, if a proposition in the delete list of behavior x is present in the precondition list of behavior y, we can say that behavior x conflicts with behavior y by making it less likely to become executable.

As explained by Maes (1989), the idea is that behavior modules "*use these links to activate and inhibit each other, so that after some time the activation energy accumulates in the modules that represent the 'best' actions to take given the current situation and goals*".

Parallel to this spreading activation mechanism, there is an input of new activation energy into the network due to the current world state and the current global

goals of the agent. In MASM, goals and world states are defined as follows:

- Goals: Propositions that the agent would like to observe as being true in the world
- World States: Propositions that the agent currently observes as being true in the world

Intuitively, the spreading activation mechanism works as follows:

- There is input of energy into behavior x for each proposition observed to be true in the world that is present in its precondition list
- There is input of energy into behavior x for each proposition in its add list that matches a global goal of the agent. Therefore, behaviors that are expected to fulfill a goal get more energy because of that
- There is a decrease in energy for behavior x for each proposition in its delete list that matches a goal. In other words, behaviors that tend to undo the agents goals get their activations decreased

Maes (1989) also distinguishes goals into two types: *once-only* goals, which have to be achieved only once and are deleted from the list of global goals as soon as they are achieved, and *permanent* goals, which have to be achieved continuously.

In order to illustrate how it performs action selection we have built an example problem with a very simple network, which should be able to control an agent in a restricted environment. The network is comprised of three behaviors, namely, "EXPLORE", "FORAGE" and "EAT".

In this scenario we used six propositions, i.e., three positive propositions and their complements: "$hungry$", "$\neg hungry$", "$exploring$", "$\neg exploring$", "$foundFood$" and "$\neg foundFood$". MASM works with lists, so these propositions can be added to any of

the lists in the behavior network in order to make it perform the tasks it is supposed to.

The lists in MASM are the following: a list of goals, a list of world state, one add list for each behavior, one delete list for each behavior and one preconditions list for each behavior. For instance, if at a time $t$ the agent had its list of goals containing "$\neg hungry$" and "exploring", and its list of world state containing "$\neg hungry$", this would mean that the agent has the goals of exploring the environment and not being hungry. However, since the world state list contains just "$\neg hungry$", it means that even though it is achieving the goal of not being hungry, it is not exploring the environment as it should.

For our hypothetical example, the contents for each behavior's add, delete and precondition lists can be seen in Table 4.1.

Table 4.1: Structure for a simple behavior network which can be used to control an explorer agent.

| $Behavior_i$: | $EXPLORE_1$ | $FORAGE_2$ | $EAT_3$ |
|---|---|---|---|
| $c_i$: | $\neg hungry$ | $hungry$ | $hungry$, $foundFood$ |
| $a_i$: | $exploring$, $hungry$ | $foundFood$ | $\neg hungry$ |
| $d_i$: | $\neg hungry$ | $exploring$ | $hungry$ |

In the scenario covered by this behavior network the agent has the global goal of *exploring* the environment. However, in order to keep doing so it also needs energy and, therefore, another of its global goals is to avoid being hungry, i.e., $\neg hungry$. If it gets too hungry while exploring the environment, it must stop exploring and start foraging for food. Once it finds food, it should eat it and, if not hungry anymore, should go back to exploring.

A diagram that illustrates the activation links between those three behaviors, the agents goals and world state can be seen in Figure 4.1. In this diagram, circles represent the behavior modules, rectangles inside the "List of goals" represent global

goals and rectangles inside "List of world states" represent world states. For instance, for behavior EAT to become executable the $foundFood$ proposition must be observed to be true in the environment, which means it should be present in the list of world states, as seen in Figure 4.1.

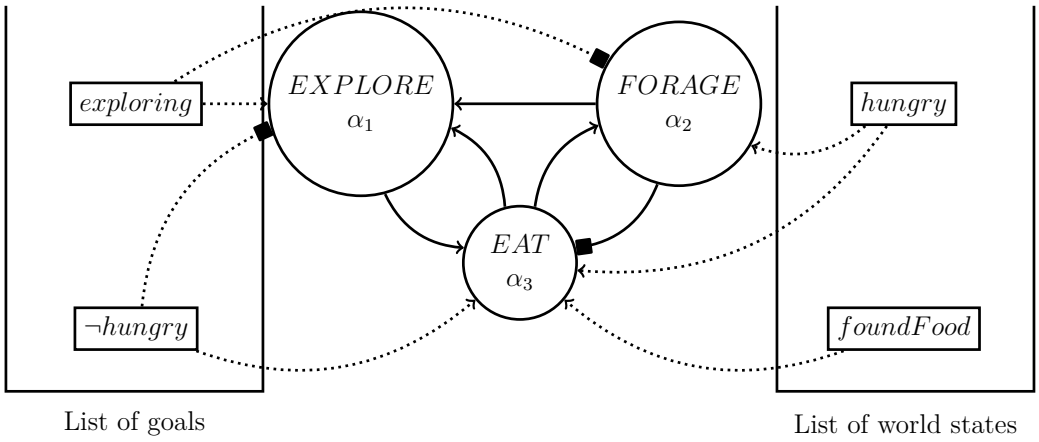



Figure 4.1: Simplified behavior network diagram at an arbitrary time $t$. Here circles represent behaviors. The list of goals is a list with the propositions that the agent would like to observe as being true in the environment. The list of world states is the list of propositions actually observed to be true in the environment at time $t$.

In this diagram, successor links are represented as full arrowed lines between two behaviors. Predecessor links are not represented in the figure because we already know that there is one of them for every successor link. Conflicter links are represented as full lines ending in a filled square. Energy input from goals and world states into the behavior network are represented as doted arrowed lines. When looking at the diagram in Figure 4.1, it is important to remember that those arrows and lines represent energy flow between behaviors, and not necessarily the sequence of execution for those behaviors.

In this scenario, the agent would go about exploring the environment. Only when it got hungry would it switch into its foraging behavior and go look for food. Despite being a toy-problem example, manufactured for explanatory purposes only, one potential drawback with this approach is that if it finds food while exploring the

environment it will simply ignore it, even if it is readily available for consumption. In this case, switching directly to the EAT behavior might in some cases be preferable, since it would avoid the need to go hungry in the first place. In MASM, however, this would not be possible for this behavior network because being *hungry* is a hard precondition for the EAT behavior to become executable.

## 4.3 A Soft-Preconditions Modified Behavior Network

A soft-preconditions modified behavior network, or "spBN", was developed to address this problem. In other words, before applying the behavior network to engineer an assistive agent, what we propose in this chapter is the modification of the original tuple described in Expression 4.1 by adding an extra term "$s_i$", as can be seen in Expression 4.2.

$$B_i = (c_i, a_i, d_i, s_i, \alpha_i) \tag{4.2}$$

The new term $s_i$, stands for a list of *soft-preconditions*. These soft-preconditions affect the flow of energy in the network the same way the original preconditions do, but do not block the behavior from being executed in case those conditions are not observed.

The behavior network described in Table 4.1 would change into the following Table 4.2.

With the aforementioned modifications, the agent is now capable of eating (in our example scenario) even if it is not hungry at the moment - because *hungry* is not a blocking precondition anymore - but only as long as it finds food on its way - because *foundFood* is still a blocking precondition - as seen in Table 4.2.

Table 4.2: Structure for a modified behavior network which can be used to control an explorer agent.

| $Behavior_i$: | $EXPLORE_1$ | $FORAGE_2$ | $EAT_3$ |
|---|---|---|---|
| $c_i$: | $\neg hungry$ | $hungry$ | $foundFood$ |
| $s_i$: | | | $hungry$ |
| $a_i$: | $exploring$, $hungry$ | $foundFood$ | $\neg hungry$ |
| $d_i$: | $\neg hungry$ | $exploring$ | $hungry$ |

### 4.3.1 Formal Description

More formally, the mathematical model of the resulting modified behavior network is similar to the one proposed by Maes (1989), and can be described as follows. Given:

- A set of behavior modules 1..n
- A set of propositions P
- A function S(t) returning the set of propositions observed as being true in the world state as time $t$
- A function G(t) returning the propositions that are a goal of the agent at time $t$
- A function R(t) returning the propositions that are goals which have already been accomplished up to time $t$
- A function executable(i,t), returning true if module $i$ is executable at time $t$, and false otherwise. A module $i$ is executable if, and only if, all its preconditions in $c_i$ are observed as being true in the environment
- A function M(j), returning the set of modules that match proposition j
- A function A(j), returning the set of modules that achieve proposition j

- A function U(j), returning the set of modules that undo proposition j
- $\pi$, the mean level of activation
- $\theta$, threshold of activation, where $\theta$ is lowered by 10% every time no module could be selected, and is reset to its initial value whenever a module becomes active
- $\phi$, the amount of activation energy injected by the state per true proposition
- $\gamma$, the amount of activation energy injected by the goals per goal
- $\delta$, the amount of activation energy taken away by goals to be protected

Given the behavior module $x = (c_i, a_i, d_i, s_i, \alpha_i)$, the input of activation to module x from the world state at time $t$ is given by Expression 4.3, with the modification here consisting on adding the $|\mathbf{s_x}|$ term in bold, where $j \in (S(t) \cap (c_x \cup s_x))$.

$$input\_from\_state(x,t) = \sum_j \phi \frac{1}{|M(j)|} \frac{1}{|c_x| + |\mathbf{s_x}|} \tag{4.3}$$

The input each module x receives from goals at time $t$ is given by Expression 4.4, where $j \in (G(t) \cap a_x)$.

$$input\_from\_goals(x,t) = \sum_j \gamma \frac{1}{|A(j)|} \frac{1}{|a_x|} \tag{4.4}$$

Removal of activation from behavior module x by goals that are to be protected at time $t$ is given by Expression 4.5, where $j \in (R(t) \cap d_x)$.

$$taken\_by\_goals(x,t) = \sum_j \delta \frac{1}{|U(j)|} \frac{1}{|d_x|} \tag{4.5}$$

Expression 4.6 defines what a behavior $x = (c_x, a_x, d_x, s_x, \alpha_x)$ spreads backward to another behavior $y = (c_y, a_y, d_y, s_y, \alpha_y)$, with $j \notin S(t)$ *and* $j \in (c_x \cap a_y)$.

$$\text{spreads\_bw(x,y,t)} = \begin{cases} \sum_j \alpha_x(t-1)\frac{1}{|A(j)|}\frac{1}{|a_y|} & if\ executable(x,t) = false \\ 0 & if\ executable(x,t) = true \end{cases} \tag{4.6}$$

Expression 4.7 defines what a behavior $x$ spreads forward to another behavior $y$, with the modification here consisting on adding the $|\mathbf{s_y}|$ term in bold, with $j \notin S(t)$ *and* $j \in (a_x \cap (c_y \cup s_y))$.

$$\text{spreads\_fw(x,y,t)} = \begin{cases} \sum_j \alpha_x(t-1)\frac{\phi}{\gamma}\frac{1}{|M(j)|}\frac{1}{|c_y| + |\mathbf{s_y}|} & if\ executable(x,t) = true \\ 0 & if\ executable(x,t) = false \end{cases} \tag{4.7}$$

Expression 4.8 defines what a behavior $x$ takes away from another behavior $y$, with $j \notin S(j)$ *and* $j \in (a_x \cap (c_y \cup s_y))$.

$$\text{takes\_away(x,y,t)} = \begin{cases} 0 & if\ (\alpha_x(t-1) < \alpha_y(t-1)) \\ & and\ (\exists i \in S(t) \cap (c_y \cup s_y) \cap d_x) \\ max\left(\sum_j \alpha_x(t-1)\frac{\delta}{\gamma}\frac{1}{|U(j)|}\frac{1}{|d_y|}\ ,\ \alpha_y(t-1)\right) & otherwise \end{cases} \tag{4.8}$$

The activation level of a module y at time $t$ is defined in Expression 4.9:

$$\begin{aligned} &\alpha(\text{y,0}) = 0 \\ &\alpha(\text{y,t}) = \text{decay}(\alpha(\text{y,t-1})).(\text{1-active(y,t-1)}) \\ &+ \text{input\_from\_state(y,t)} + \text{input\_from\_goals(y,t)} \\ &\text{- taken\_by\_goals(y,t)} \\ &+ \textstyle\sum_{x,z} (spreads\_bw(x,y,t) + spreads\_fw(x,y,t) - takes\_away(z,y,t)) \end{aligned} \tag{4.9}$$

Where x ranges over all modules in the network and z over all modules except y. The decay function is calculated in such a way that the total activation in the network remains constant: $\sum_y \alpha_y(t) = n\pi$.

Finally, the behavior module $i$ that becomes active at time $t$ is as defined by Expression 4.10. Notice that, according to this expression, there can be only one behavior active at any given time $t$.

$$\text{active(t,i)} = \begin{cases} true & if \begin{cases} \alpha(i,t) >= \theta \ (1) \\ executable(i,t) = true \ (2) \\ \forall j \ fulfilling \ (1) \ and \ (2) \ : \ \alpha(i,t) \ >= \ \alpha(j,t) \end{cases} \\ false & otherwise \end{cases} \tag{4.10}$$

In the following sections we apply this modified behavior network in the context of building an assistive agent which should provide interesting suggestions for a robotic wheelchair user.

## 4.4 Experimental Setup

The assistive agent described in this work is part of a larger project for the development of an assistive system for a robotic wheelchair user in a simulated smart environment, which is seen in Figure 4.2. Our work was performed in a simulated environment, and consists on the development of the "suggestion module" at the bottom right corner of this diagram.

In the assistive platform, the operator will be located on a robotized wheelchair equipped with a manipulator to give him back some of his lost autonomy. For the time being a pioneer P3DX or a Seekur Jr, on which a 7 degrees of freedom Schunk lwa3 anthropomorphic manipulator is mounted, are used in place of this robotic

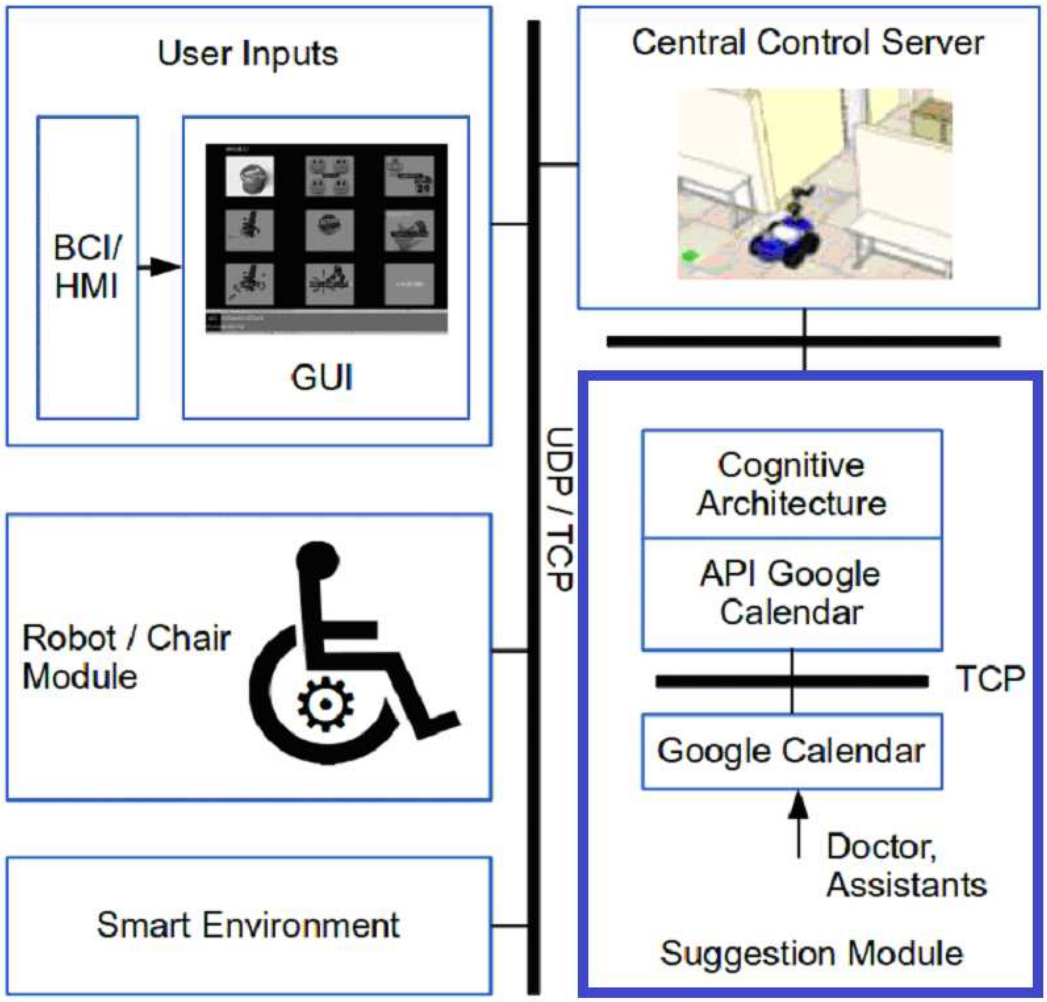



Figure 4.2: Assistive platform overview. The work described in this chapter is represented by the "Suggestion Module" at the bottom right corner of this diagram.

wheelchair. The Seekur Jr robot was chosen as it has about the same size as a real motorized wheelchair. Operation control for the robot is being developed in a smart environment that facilitates the navigation and assists the user in his life.

The smart environment is a set of places enhanced by sensors and actuators, linked together by way-points, which provides the robots and the user services like localization, path to follow in order to reach a specific location or exit, successive set of actions to be taken by the robot to do a task, environmental data (temperature, crowding of a place, Points Of Interests (POIs) location, etc.), and interaction with

devices (air conditioning, TV control, calling of an elevator, automatic door opening etc..).

A topological place or simply a place is a room or a part of a room connected to other places called neighbors. A place can be linked to another place by a single exit point. An RFID (Radio-Frequency Identification) tag or array of tags (or other detection device) determines the physical limits between connected places (i.e., a place and its neighbors) that, when crossed, let the robot and the environment knows that the robot transited into the neighbor place. Each place has a data structure defined by a name (its handle), a local frame (in which any coordinates defined inside the place are expressed), a metric map (a 3D model of the place of the static objects of the room like walls, heavy furniture), a set of POIs, and a set of connected neighbor places.

The User Input module allows the operator to interact with the robotic system and smart environment, via a Graphical User Interface (GUI), using a Brain Computer Interface (Electroencephalogram EEG) that detects his brain wave activity. The GUI gives the operator the possibility to move his robotized wheelchair and manipulator, and interact with his environment. Additional Electromyography (EMG) detecting the electrical potential generated by muscle cells, and Electrocardiogram (ECG) are linked via Blue-tooth to a smart-phone monitoring the patient's biosignals to eventually evaluate his physical and mental state and trigger alarms or actions. Eventually a remote help can be asked by the operator, to let an assistant take control of the robotic platform.

The GUI is communicating with the Central Control Server (CCS) that plans, organizes and executes the set of successive actions to be done by the wheelchair, depending on the smart environment sensory information and the operator input. The CCS owns a dynamic engine based simulator used to implement behaviors, test scenarios, or train the operator to use the platform. The CCS is here used

as a simulation platform to emulate the smart environment services (sensors and actuators) and the robot's behaviors they trigger.

The flow of use of this platform is as follows. At first, the operator navigates through the GUI's menus to select a target destination. The CCS computes an optimal topological path toward the target place using a uniform cost search algorithm (Russell and Norvig, 2010) where each action required to get from one place to another (moving toward an exit, sequence required to open a door, etc.) is weighted considering its difficulty, rate of success, and execution time. The optimal path is the one with the smallest action cost to get to the desired place. Once this topological path is generated, the robot knows the succession of places it needs to navigate through to reach the goal place.

A metric map, loaded from the CCS, is used to navigate inside each place - actually, the 3D model of the room associated with the place. The path is obtained using the Rapidly-exploring Random Tree (RRT) (Kuffner Jr. and Steven M. LaValle, 2000) algorithm. The robot follows the computed metric path toward the exit of each place, using a localization service provided by the smart environment or its gyro-odometer when no service is available. If an obstacle is detected in its way, either a Braitenberg obstacle avoidance algorithm (Braitenberg, 1986) avoids the obstacle or the smart environment re-plans the route toward the exit. When getting closer to the exit of the present place (e.g., a doorway), the robot searches for a line on the ground and a line following algorithm leads it along this line toward the next place. An RFID located under the line resets the robot odometer and let the CCS know that the robot got inside the next room.

This process is repeated until the final place is reached. At anytime the operator can change the final destination, and each time a new place is reached, the list of accessible POIs inside the place are listed in a menu. The operator can choose to interact with one of the POI (e.g., get a coffee), and then resume his trip.

When the target place is reached, the operator can select an action through the GUI shown in Figure 4.3. The layout and interaction logistic was inspired by other works seen in the literature (Ferreira et al., 2007; Holzner et al., 2009; Ferreira et al., 2012; Escolano et al., 2012).

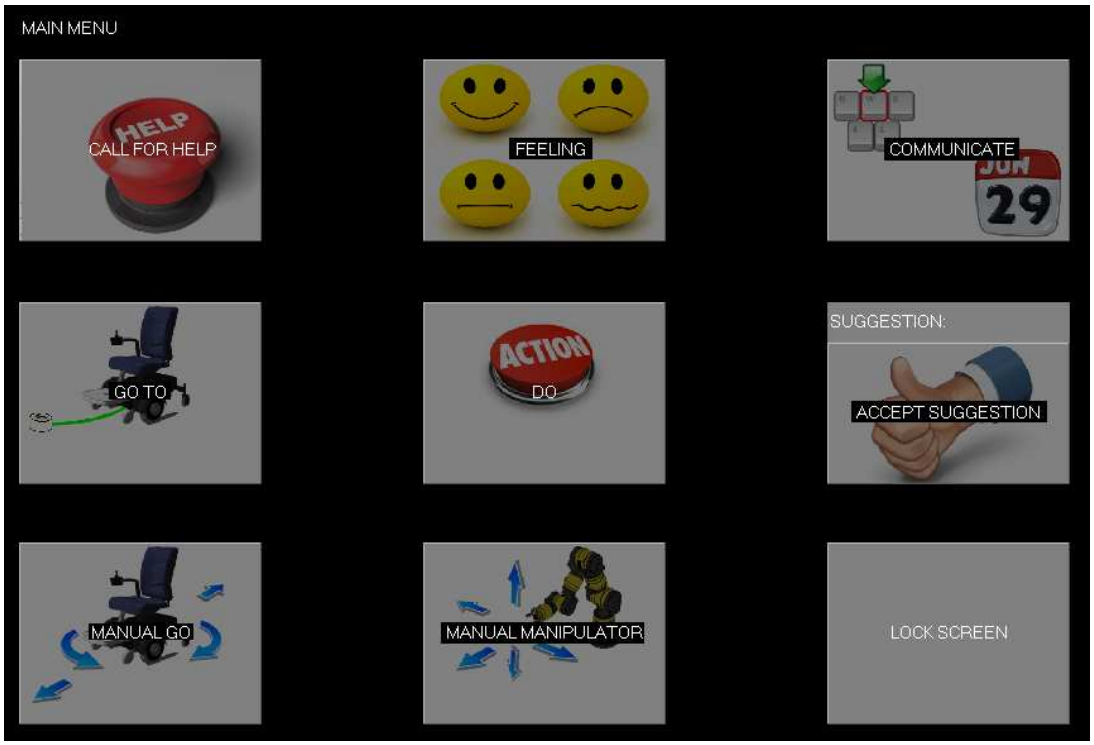



Figure 4.3: User interface main menu.

The main menu contains nine options to be chosen by the user.

- Call for help: Calls a human assistant in case of emergency.
- Feeling: Used by the user to communicate his/her current emotional state.
- Communicate: Access a speller system which is capable of sending e-mails, SMS or send text to the screen.
- Go to: Allows the user to select a room in the smart environment as a destination.

- Do: Leads to a menu with everything that can be performed (POIs) in the current room.
- Accept Suggestion: Accepts the POI suggestion given by the cognitive architecture.
- Manual go: Enters a shared control interface which allows the user to manually navigate the environment.
- Manual manipulator: Grants access to a control menu for the robotic arm.
- Lock Screen: Stops automatic sweep system.

In this work we investigate the particular case of a *motor imagery* and a *SSVEP* (steady state visual evoked potential) paradigms to generate three output channels (control channels) in order to navigate inside the menus.

With the first channel the user is able to move a selection box from option to option, starting at the first option on the top left. A second channel is used to actually select the currently marked option, whilst a third channel is reserved as a cancel command, which undoes the last selection.

## 4.5 Cognitive Architecture Suggestion System

As a framework for the development of this suggestion agent we have used the cognitive architecture described in Chapter 3.

Focusing on deliberative action selection and inspired by previous successful works (Negatu and Franklin, 2002), a modified behavior network based on the original action selection mechanism by Maes (Maes, 1989) has been implemented. As pointed out by Tyrrell (Tyrrell, 1994), the original behavior network proposed by Maes suffers from a number of drawbacks, such as not being able to perform more than one action at the same time, using Boolean propositions instead of continuous variables

or oscillating between behaviors. The last one has been addressed in the current implementation, while other issues will be taken care of during development, since parallel works have shown it to be a possible endeavour (Negatu and Franklin, 2002; Dorer, 2010). None of those drawbacks have affected the performance of the agent for this particular application.

Figure 3.6 depicts the concepts of the Central Executive subsystem as it has been implemented so far. Permanent goals are defined by the agent's most vital objectives and motivations. As described by Maes (Maes, 1989), one time only goals have the same effect as permanent goals, the difference being that once they have been accomplished they are removed from the list of current goals to be pursued. They are therefore defined depending on the interaction between the agent and its environment, as well as on its own inner state.

In this implementation, each behavior is a codelet and as such runs as an independent process. The whole network is therefore a distributed system, in which codelets communicate with each other exchanging energy, as described by Maes (Maes, 1989).

The behavior network has been modeled taking into account the relationship between the user and the intelligent environment. As can be seen in Figure 4.4, the environment is a two storey building (which could have been as small as an apartment or as large as a hospital), with each floor being divided into separated rooms. At each room there is a number of preprogrammed actions the user is capable of performing. These actions are related to POIs (Points of Interest) which define where in the room the user must move to in order to perform a given task.

The rooms and their respective POIs are organized as follows. For clarity, "ELV1" and "ELV2" are elevators linking the two floors of the building. "OJ" refers to the action of getting an orange juice from the juice machine at "ROOM2" or "HALL1". The other tags are self-explanatory, and point to the possibility of performing the indicative action at the given room.

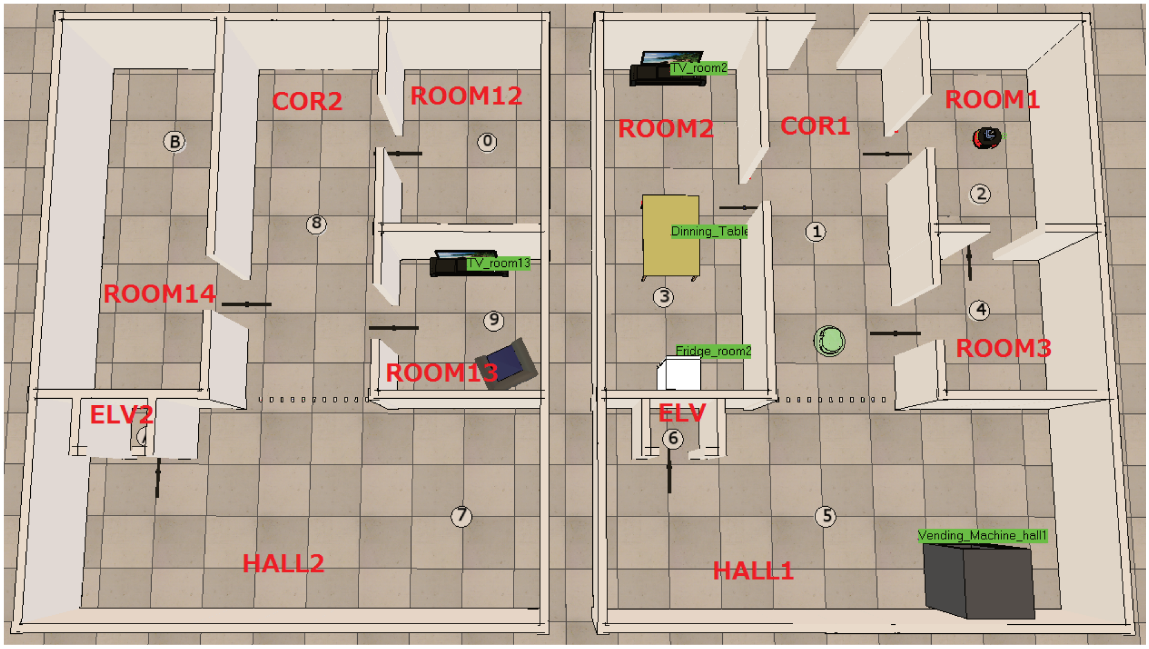



Figure 4.4: Bird view of the intelligent environment designed for the experiments.

- HALL2 : “∅”
- ROOM2 : “TV”, “LUNCH”, “DINNER”, “BREAKFAST”, “WATER”, “OJ”
- HALL1 : “WATER”, “COFFEE”, “OJ”
- ROOM1 : “RADIO”
- COR1 : “∅”
- ROOM14: “∅”
- COR2 : “∅”
- ROOM12: “∅”
- ELV2 : “∅”
- ROOM13: “TV”
- ELV1 : “∅”

- ROOM3 : "∅"

There are two types of basic behaviors the user can perform which are of major interest to the cognitive architecture, "GO" behaviors and "DO" behaviors. A "GO" behavior relates to the possibility of the user choosing to go to a specific room in the building. After the user chooses to go to a given room by selecting it at the interface, the robotic system takes control and automatically drives its way to the target. A "DO" behavior is the possibility of performing a task related to a particular POI. For instance, being at "ROOM2", the user would be able to perform a "DO_WATER_AT_ROOM2" behavior in order to have some water.

However, by examining the particular case of "DO_LUNCH" behavior, for instance, a major problem is revealed.

If the preconditions required to perform "DO_LUNCH" were "AT_ROOM2" and "TIME_LUNCH", that would mean the user would only get the suggestion for having lunch in case he or she was at room2 and also it was lunch time. This would exclude the possibility of suggesting lunch at other times of the day (the user might want to have lunch later for instance). This is the reason why we have used the modified behavior network described in Section 4.2.

For the construction of this behavior network, a "GO" behavior was created for each room in the building. An example of how their tuples where defined can be seen in the following case for the "GO_ROOM2" behavior.

| | |
|---|---|
| Preconditions: | "PERFORM_GOTO_SUGGESTION" |
| Soft Preconditions | "AT_COR1" |
| Add List | "AT_ROOM2" |
| Delete List | "AT_COR1" |

This means that it is possible to reach room 2 from corridor 1, and once the agent does so it will no longer be at corridor 1.[1] Therefore, the set of "GO" behaviors form

[1]The precondition "PERFORM_GOTO_SUGGESTION" is a preposition that can be passed

a topological map of the environment, as shown in Figure 4.5, within the behavior network's structure. This topological information is embedded into the behavior network in a top-down manner by the smart environment. In this figure, each circle is a room, and arrows show the possibility to go from one room to another.

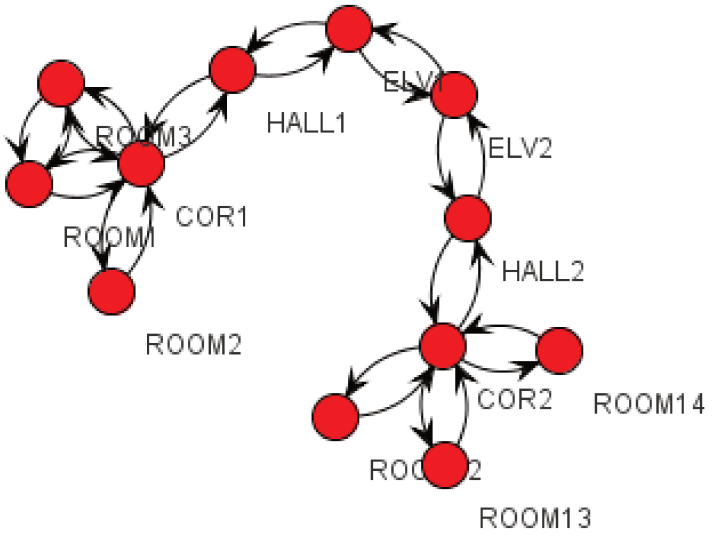



Figure 4.5: Smart environment's topological map. The topological information is embedded into the behavior network in a top-down manner by the smart environment.

An example of a "DO" behavior is "DO_LUNCH_AT_ROOM2", which can be described as the following.

| | |
|---|---|
| Preconditions: | "AT_ROOM2" |
| Soft Preconditions | "PATH_ROOM2", "TIME_LUNCH" |
| Add List | "DO_LUNCH" |
| Delete List | "$\emptyset$" |

This behavior is more likely to be activated if the agent is either at or passing through room 2, where it is possible to have lunch, but the suggestion is enacted only when the user is at room2.

to the agent so it knows if it should suggest "GO" behaviors. In this implementation it is set as false, so the agent only suggests "DO" behaviors.

The main controller from Figure 4.2 sends to the cognitive architecture agent the information of which room the user is at the moment ("AT_" propositions) and also all rooms the user is going to pass through ("PATH_" propositions) in case he or she has chosen to go to a particular place in the building.

Motivation propositions and scheduled event's times are retrieved from an online agenda using the Google Calendar API. Once a given behavior is executed, it sends a suggestion to the Main Controller with its own name. For instance, once the "DO_LUNCH_AT_ROOM2" behavior gets executed, it sends a "SU_LUNCH" string to the Main Controller, with the "DO_" prefix being replaced by a "SU_" prefix in order to denote a suggestion.

A more detailed description on how to assemble the behavior network for the assistive agent can be seen in Algorithm 1. The procedure receives a list with all selectable actions in the smart environment and another list with all rooms and their connections. It then creates "go" and "do" behaviors, and add them to the behavior network.

The final structure for the behavior network proposed in this work can be seen in Figure 4.6. Circles within the limits of the continuous line are "DO" behaviors while those within the limits of the dashed line are "GO" behaviors. Links between behaviors with arrows are "conflict" connections, while those without arrows are "successor/predecessor" connections.

## 4.6 Experimental Results

A number of simulated experiments were performed to evaluate if the behavior network was indeed presenting valuable suggestions and performing as expected [2].

First, a statistical evaluation was performed in order to assess the reduction in

[2]After a few trials we have settled, for these experiments, with the following values for the behavior network's global variables: $\phi = 0.1$, $\gamma = 0.01$ and $\delta = 0.02$.

**Algorithm 1** Assemble Behavior Network

**Input:**
A : list of all selectable actions
R : list of all rooms
**Output:**
B : assembled behavior network
**function** CREATEGOBEHAVIORS($A, R$)
    G : list of go behaviors
    **for** each *room* in R **do**
        create a *behavior* "$GO\_ < room >$
        add "PERFORM_GOTO_SUGGESTION" to its preconditions list
        add "AT_<room>" to add list
        **for** each *neighbor* of *room* **do**
            add a "AT_<*neighbor*>" to its soft-preconditions list
            add a "AT_<*neighbor*>" to delete list
        add *behavior* to G
    **return** $G$
**function** CREATEDOBEHAVIORS($R$)
    D : list of do behaviors
    **for** each *action* in A **do**
        **for** each *room* in R **do**
            create a *behavior* "$DO\_ < action > \_AT\_ < room >$"
            add "$AT\_ < room >$" to its preconditions list
            add "$PATH\_ < room >$" to its soft-preconditions list
            add "$TIME\_ < room >$" to its soft-preconditions list
            add "$DO\_ < action >$" to its add list
            add *behavior* to D
    **return** $D$
G = createGoBehaviors(A,R)
D = createDoBehaviors(R)
B = $G \cup D$
**return** $B$

the required number of interactions between user and system. Then, a qualitative evaluation of the dynamics of implemented behaviors was performed with a single example.

### 4.6.1 Reducing the Number of Interactions

Distinct scenarios were simulated and analyzed considering two different systems: a system without a suggestion agent and one with a behavior network based agent

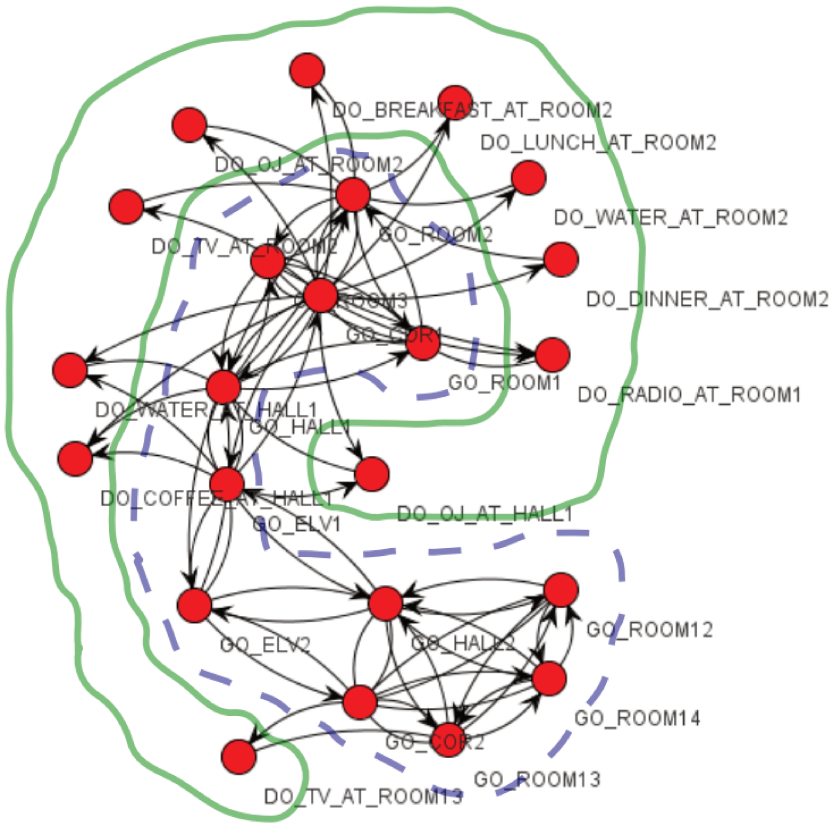



Figure 4.6: Behavior Network designed for the intelligent suggestion agent. Circles within the limits of continuous line are "DO" behaviors while those within the limits of the dashed line are "GO" behaviors. Links between behaviors with arrows are "conflict" connections, while those without arrows are "successor/predecessor" connections.

which makes interesting suggestions based on what can be done at the current room.

For both scenarios an automatic selector *bot* was developed which acted as a user with the mission of accomplishing all its objectives. Each experiment consisted first on randomly selecting three POIs as being the user's objectives and positioning the robot at a random room in the simulated environment. The automatic *bot* would then check if any of its objectives could be accomplished at the current room and would perform them. If it still had objectives to fulfill, it would go to the nearest room where it would be able to accomplish at least one of them. The cycle would proceed until there were no objectives left to perform.

In the scenario without a suggestion agent, the *bot* would randomly choose between any of the POIs available in the room that were also objectives. In the scenario with a suggestion agent, the *bot* would first look at the suggestion available and then, if it was of any interest to it, would select it instead of going into the DO menu looking for it. Choosing a POI from the suggestion option in the main menu costs less to the user in terms of interaction than selecting the DO option and looking for the desired POI inside it. For instance, starting at the main menu, selecting the suggestion would cost the user moving to the suggestion box and then selecting it. If the user chooses to ignore the suggestion however, it would cost him moving to the DO button, selecting it, moving to the desired POI and then performing another selection to finally activate it. Assuming the agent got the suggestion right, accepting it costs less in terms of user interaction than ignoring it.

A comparative cost analysis was performed by evaluating the total cost for each experiment. In order to avoid the possibility of placing the "suggestion" and "do" options at particular positions, which would alter their relative costs, their costs were generalized as being equivalent to the difficulty of reaching the middle of the user interface menu from Figure 4.3 by using the previously explained 3-channels system. The same rule was applied to reaching a POI once inside the "do" sub-menu.

Since the environment being used has a small number of POIs in each room (with some rooms having no POIs at all), a third scenario was emulated in which the cost calculation was performed considering a "do" sub menu always containing 36 (a 6 by 6 grid) to choose from.

A total of 100 executions with different initial conditions were performed for each scenario. The result can be seen in Figure 4.7, which shows a box plot with the mean costs for the scenario without a suggestion system (1-left), with a suggestion system applied to the smart environment described in Figure 4.4(2-center) and a simulation of each room having 36 options for the user to choose from (3-right).

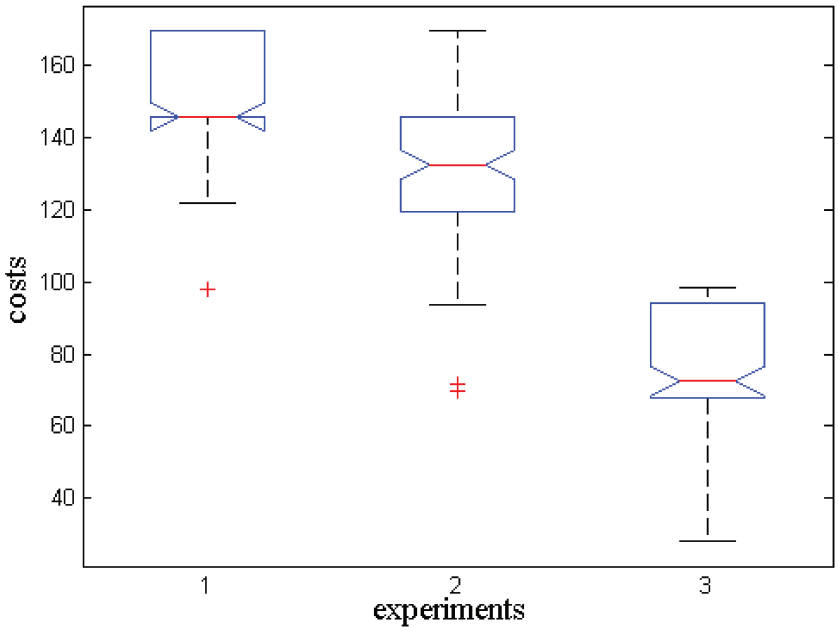



Figure 4.7: Comparison between simulated experiments without a suggestion system (1-left), a suggestion system applied to the smart environment described in Figure 4.4(2-center) and a simulation of each room having 36 options for the user to choose from (3-right). Blue box structures represent interquartile ranges.

One-way ANOVA (Moore et al., 2009) was performed on simulated results from the three scenarios producing a $p$ value of $4.52 \times 10^{-074}$, which points to the conclusion that their results are not the same. This, together with Figure 4.7 suggests that the greater the number of options are, the more expressive the reduction in number of required interactions is.

### 4.6.2 Behavioral Dynamics

Placing the robot initially at a room with no POIs, such as ELV2, without any objectives, expected path or scheduled events, the behavior network is expected to have no reason to make any suggestion. The activation for all behaviors should remain at zero and none of them would be executable.

In Figure 4.8, we show the activation levels for each suggestion along time, given the following scenario: the robot was initially placed at HALL1, which contains the

POIs for having some water, coffee or orange juice. This made it more likely for the behavior network to suggest one of those three actions to the user, as can be seen by the increase in activation levels for the corresponding behaviors. At the 10 seconds mark in Figure 4.8, after some deliberation, the threshold level theta reaches the executable behavior with the highest activation level, which prompted the behavior network to suggest an "Orange Juice" action. Near 21 seconds at the same figure, the "TIME_COFFEE" proposition was read by the network (it was detected as being true in the world), which caused "DO_COFFEE" actions to become more relevant than others such as "DO_WATER", causing it to get chosen for execution later on.

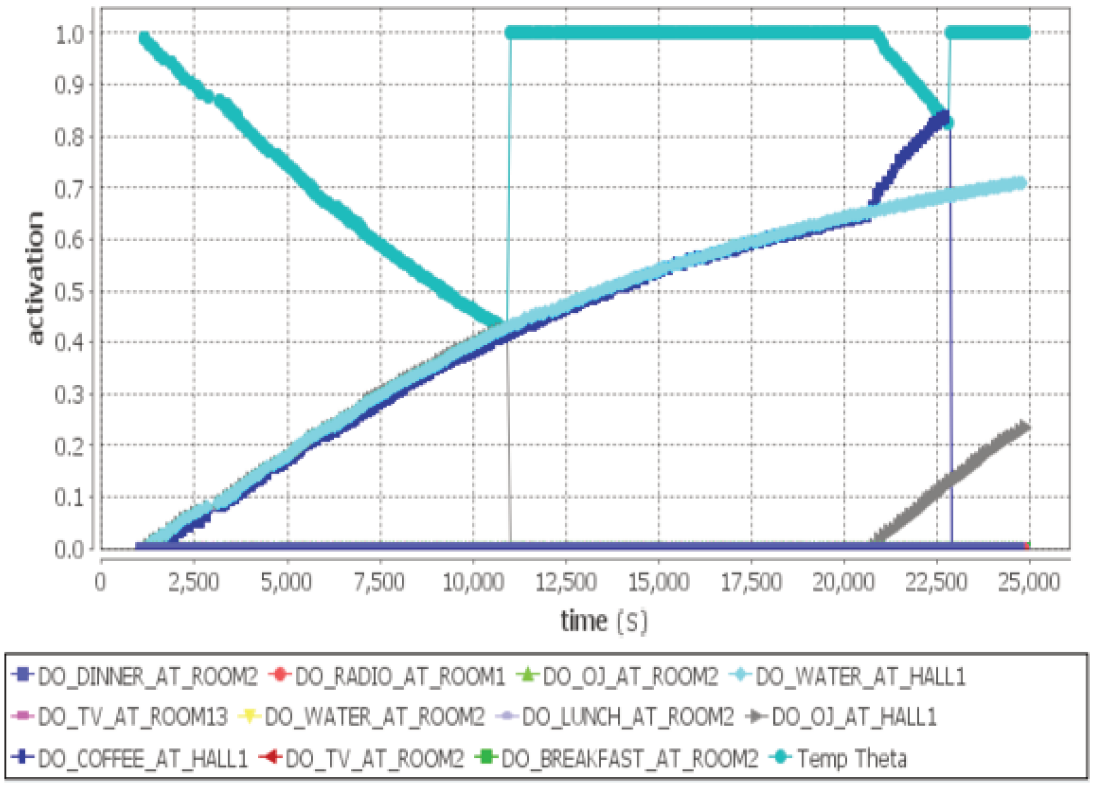



Figure 4.8: Effect of the behavior network as a suggestion system when no objectives or expected path are set. The robot is initially at a room with water, coffee and orange juice POIs and, during simulation, it becomes "coffee time" (time in seconds).

Another scenario was simulated to verify the behavior network's ability at performing goal-directed action selection. The robot was initially placed at ROOM3,

which contains no available POIs. Objectives “DO_WATER” and “DO_RADIO” were set for the agent and the user informed the architecture of its intention of going to HALL1 by first passing through COR1.

Activation levels for the described scenario can be seen in Figure 4.9, where the behavior “DO_WATER” at HALL1 initially got the highest activation. That happened because that is where the user intends to go, while at the same time it is there where the user can perform the “DO_WATER” behavior.

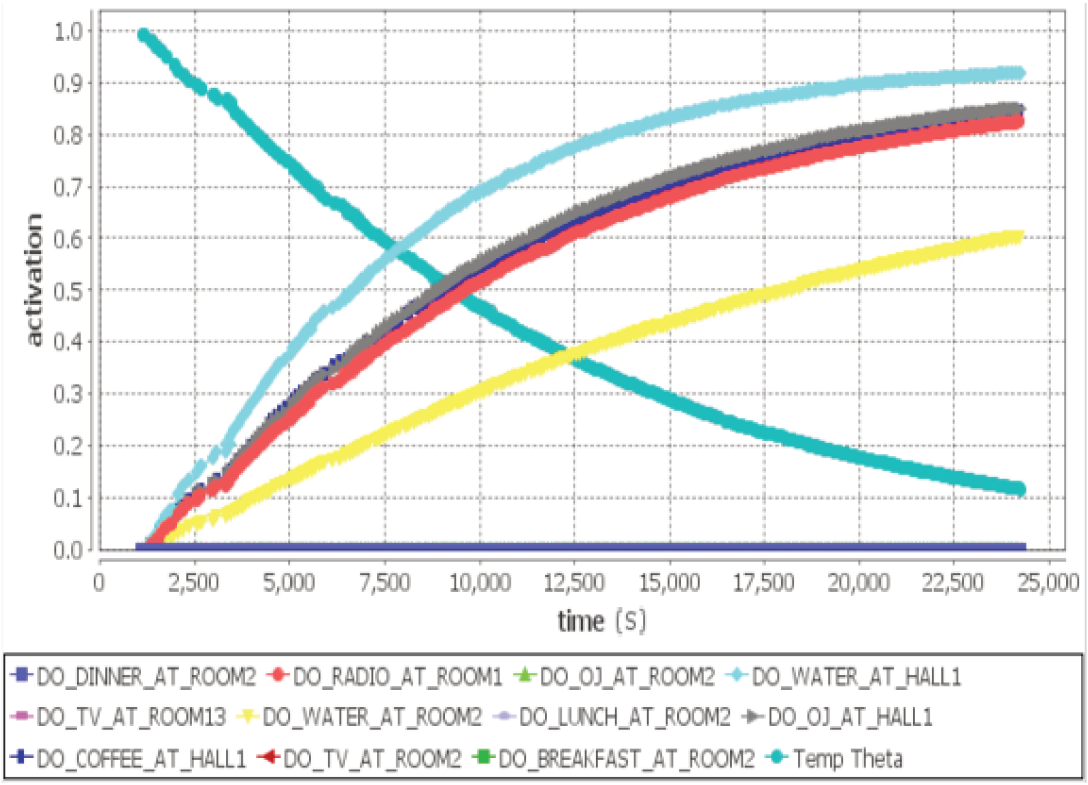



Figure 4.9: Effect of the behavior network as a suggestion system when the user has set the objectives for the day as drinking water and listening to radio. The robot is initially at a room with no POIs and, during simulation, the user plans to go to hall 1 (time in seconds).

Figure 4.10 shows a detailed view of the same system from Figure 4.9 after reaching steady state. The “DO_RADIO” behavior from ROOM1, which accomplishes one of the given objectives, had a higher activation level than “DO_WATER” from ROOM2, because ROOM1 is closer to where the user is at the moment.

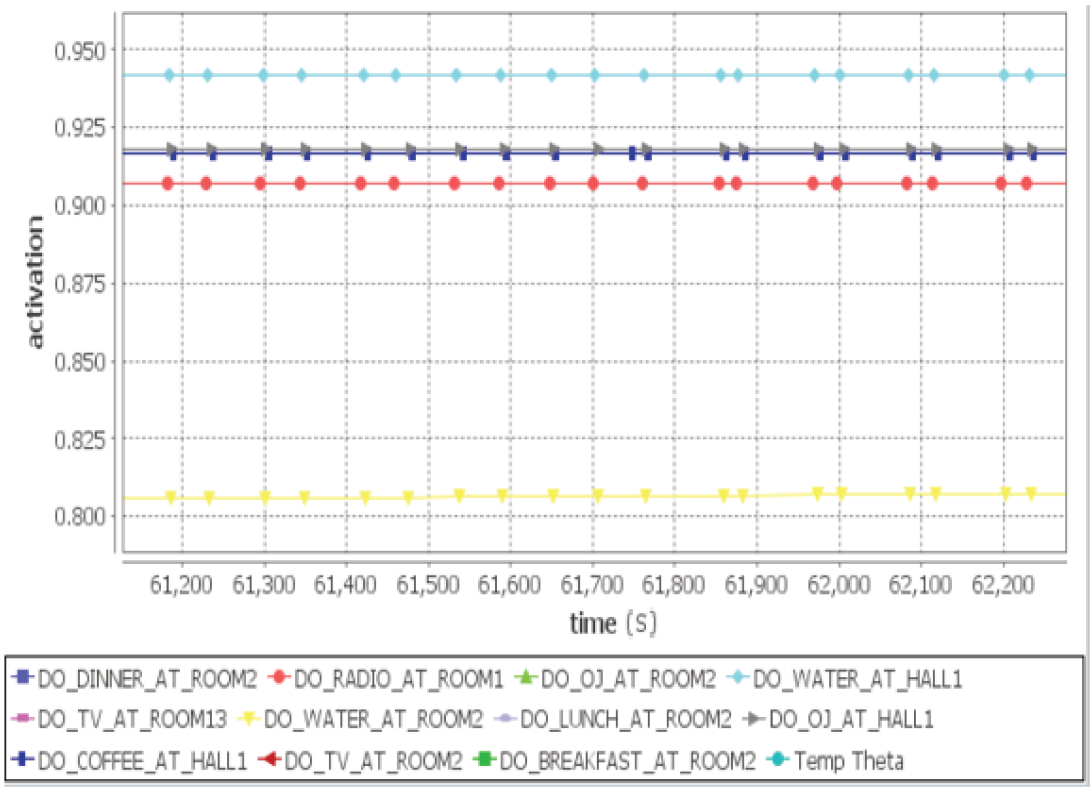



Figure 4.10: Detailed view from Figure 4.9 at steady state (time in seconds).

## 4.7 Discussion

Obtained results have shown that the behavior network is capable of producing suggestions given not only current world state but also objectives and user intentions. It was able to take into account predefined objectives, scheduled events and topological information about the environment in order to deliberate over the possible behaviors and attempt to make relevant suggestions. These results, however, apply to the simulated environment presented in this chapter. Therefore, it is important to stress that, in order to validate this solution for a real world scenario, similar experiments should be reproduced with human users in future research.

The original behavior network model proposed by Maes (1989) had to be adapted in order to make it useful in this scenario. Other modifications are bound to be necessary in order to further develop and improve the intelligent agent. Considering its scalability, the whole behavior network is automatically generated, in a top-down manner, from a JSON (JavaScript Object Notation) file containing the description of the smart environment. This allows the network to be automatically reconfigured after any relevant modification in the environment is performed.

Using our modified ConsCale presented in Chapter 3, we can say this assistive agent works with the following skills:

**Reptilian**

- CS2-1 : Fixed reactive responses.

  BP2-1 : Fixed reflexive suggestions based on the world state.

- CS3-3-5 : Selection of relevant sensory/memory/motor information.

  BP3-3-5 : The agent reacts to predefined sensory inputs provided by the central controller and stores basic information in fixed memory.

- CS3-6 : Evaluation (positive or negative) of selected objects or events.

  BP3-6 : The agent is able to evaluate sensory input as being relevant, good because helps to achieve goals, or irrelevant for the user.

- CS4-2 : Directed behavior toward specific targets like following or escape.

  BP4-2 : Selected suggestions are directed towards specific targets, such as water, orange juice, a specific room, etc.

**Paleomammalian**

- CS5-2 : Seeking of multiple goals.

  BP5-2 : The behavior network allows setting a number of different goals that influence action selection at the same time.

**Neomammalian**

- CS4-4 : Basic planning capability: calculation of next n sequential actions.

  BP4-4 : Activation spread across the behavior network allows the agent to take into account the sequence of selected actions.

- CS5-5 : Advanced planning capability considering all active goals.

  BP5-5 : The behavior network considers all active goals when deciding a better order of behaviors.

The behaviors implemented for developing an assistive agent granted the cognitive architecture full reptilian capability. It also provided it with a number of skills from the paleomammalian and neomammalian levels. However, one major drawback of this system up to this point is its inability to learn new knowledge from experience. To better serve as an assistant, the agent should be able to learn from feedback and, by doing so, adapt to its human user personal goals and habits.

# Chapter 5

# A Neuroscience Inspired Gated Learning Action Selection Mechanism

"Artificial intelligence is the science of making machines do things that would require intelligence if done by men."

Marvin Minsky

## 5.1 Introduction

In Chapter 4 we described a modified behavior network applied to the development of an intelligent agent responsible for making action suggestions to a BCI (Brain Computer Interface) user in the context of an intelligent environment.

Results have shown that the behavior network is capable of producing interesting suggestions, deliberating not only over the current world state but also over objectives and user intentions. It's major drawback however was its inability to learn new knowledge from experience.

In this chapter we present a new algorithm for action selection in the context of intelligent agents capable of learning from sparse in time and delayed rewards. Inspiration for the proposed algorithm was drawn from computational neuroscience models of how the human prefrontal cortex (PFC) works. This algorithm should improve our cognitive architecture with the ability to learn from experience, and select actions based not only on the current state of the world, but also on future implications of those actions.

The mathematical and algorithmic study of how the human conscious mind solves the problem of selecting the next action to be taken has produced many interesting results (Reggia, 2013; Baars and Franklin, 2009). As we have seen in Chapter 2 "executive functions" are those responsible for what is usually considered to be "intelligent behavior", and work by controlling and managing other cognitive processes. They are a "macro construct" (Alvarez and Emory, 2006), in the sense that multiple sub-processes must work in conjunction to solve complex problems. The term "executive function" is therefore used as an umbrella for a wide range of cognitive processes and sub-processes (Chan et al., 2008), with the most prominent being *action selection*, *planning*, *selective attention* and *learning* (Baars and Gage, 2010; Fuster, 2008; Frank and Badre, 2012).

In their nature, executive functions are mostly future directed and goal oriented, whilst exerting supervisory control over all voluntary activities. They deal with prospective actions and deliberate plans to achieve goals which can be defined by the executive itself. In a sense, this is what the frontal lobe, in particular the prefrontal cortex, does for humans (Baars and Gage, 2010; Chersi et al., 2011).

In this work we propose an action selection algorithm inspired by the computational neuroscience model described in the Leabra framework (O'Reilly, Munakata, Frank and Hazy, 2012; Hazy et al., 2007). With its PBWM (Prefrontal Cortex Basal Ganglia Working Memory) algorithm, Leabra models how the human PFC interacts with basal ganglia in order to learn from rewards separated in time and select the most appropriate action given a particular stimulus. In other words it performs, with the exception of *planning*, all major executive functions.

The PBWM mechanism strives to follow the biological cognitive process as closely as possible. The present work, however, focuses more on the development of a new algorithm, which is also biologically inspired but ultimately designed to be used in the development of intelligent agents. In order to do so, the inner workings of the PBWM mechanism (Hazy et al., 2007) were abstracted in the form of an action selection algorithm, whose behavior towards new stimuli is defined by an optimized tree structure. We have observed that this abstraction provided a number of advantages, such as:

- Representing solutions as trees, instead of neural networks, allows one to interpret the knowledge it encodes in a direct manner.
- This method turned the learning process into a combinatorial optimization problem. This potentially makes the use of different optimization techniques straightforward.

Details on how we achieved this can be seen in Section 5.3. The remainder of this chapter is organized as follows. Section 5.2 presents our initial motivations

for developing this algorithm. Section 5.3.1 provides a brief description of how the original PBWM mechanism works, while describing the "1-2-AX" working memory task used as a benchmark for validation. Section 5.3.2 then describes our proposed gated-learning action selection (GLAS) mechanism. In Section 5.4 we apply GLAS to learn the 1-2-AX working memory task and present training results given a particular sequence of events. The chapter closes with Section 5.5, where we discuss obtained results and argue the pros and cons of the proposed algorithm.

## 5.2 Motivation

The core motivation for developing this algorithm is to take advantage of what neuroscience, and more specifically computational neuroscience, has produced that could be useful for the development of artificial intelligent agents.

Specifically, adapting the PBWM mechanism to provide human-readable (which can be interpreted by humans) solutions was motivated by our previous work with behavior networks and action selection (Raizer et al., 2012; Raizer, Rohmer, Paraense and Gudwin, 2013a).

As a matter of fact, not only should an agent be able to select the most relevant action at a given time, but it should do so while taking into consideration its future consequences. Traditional reinforcement learning mechanisms, such as variations of SARSA and Q-Learning (Russell and Norvig, 2010), have often been used to solve challenging problems in engineering and computer science (Stone et al., 2005; Mahadevan and Connell, 1992; Riedmiller et al., 2009; Bagnell and Schneider, 2001; Kolter and Ng, 2011). They lack, however, an ability the mammalian brain excels at: to bridge the gap between actions and late rewards (Bakker et al., 2003).

Let us take for instance the task of teaching a dog that taking a bath can be a rewarding experience.

In Figures 5.1, 5.2 and 5.3, "stimulus from senses" represents the dog's perception

of having a bath. Figure 5.1 represents the use of a synchronous reward (reward is given while the dog is still perceiving the stimulus) and Figure 5.2 represents the attempt to use a late reward. We see therefore 4 bath episodes being represented here, and learning is represented by the dotted vertical line. If every time during bath its owner gives the dog a cookie (reward), a burst of dopamine neural firing happens in the dogs' brain. Since the stimulus of taking a bath is still active, the brain manages to correlate this reward with the beginning of the stimulus, linking the perception of taking a bath to being something good, even if the reward is not given again after the learning step (dashed reward).

However, if the owner waits to give its dog a cookie long after each bath is finished, something like what is described in Figure 5.2 could happen. In this case, there is nothing linking the moment of reward to the appearance of the stimulus.

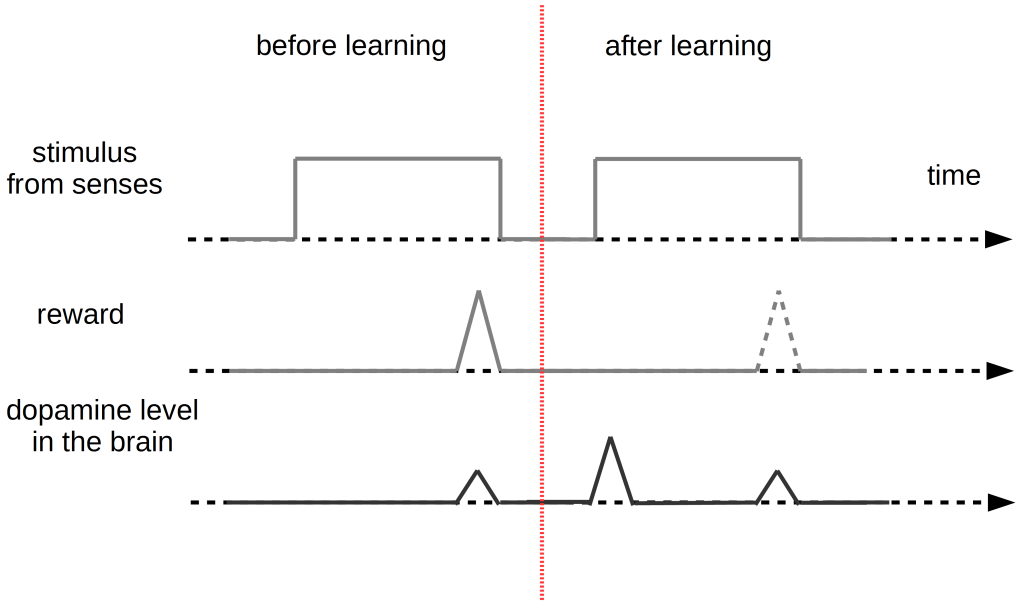



Figure 5.1: Dopamine firing propagates backwards towards continuous stimulus. Adapted from Hazy et al. (2007) and O'Reilly et al. (2007).

Dogs, however, are mammals with highly developed prefrontal cortices. The PFC works, among other things, as a temporary container for storing stimuli representations.

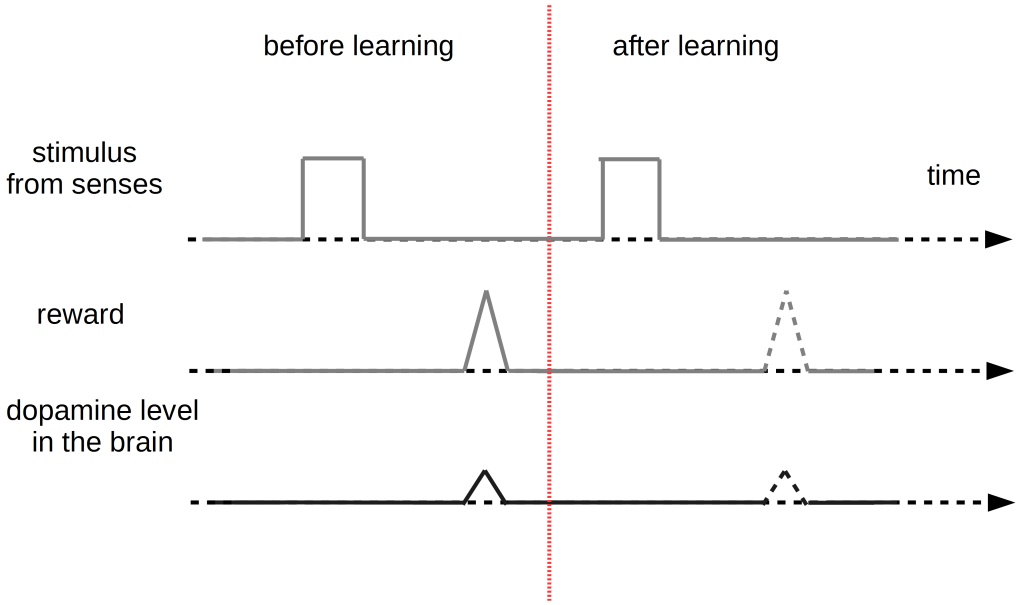



Figure 5.2: Dopamine firing can not propagate back due to gap between stimulus and reward. Adapted from Hazy et al. (2007) and O'Reilly et al. (2007).

Therefore, what really should happen in the previous case, is something like what we see in Figure 5.3.

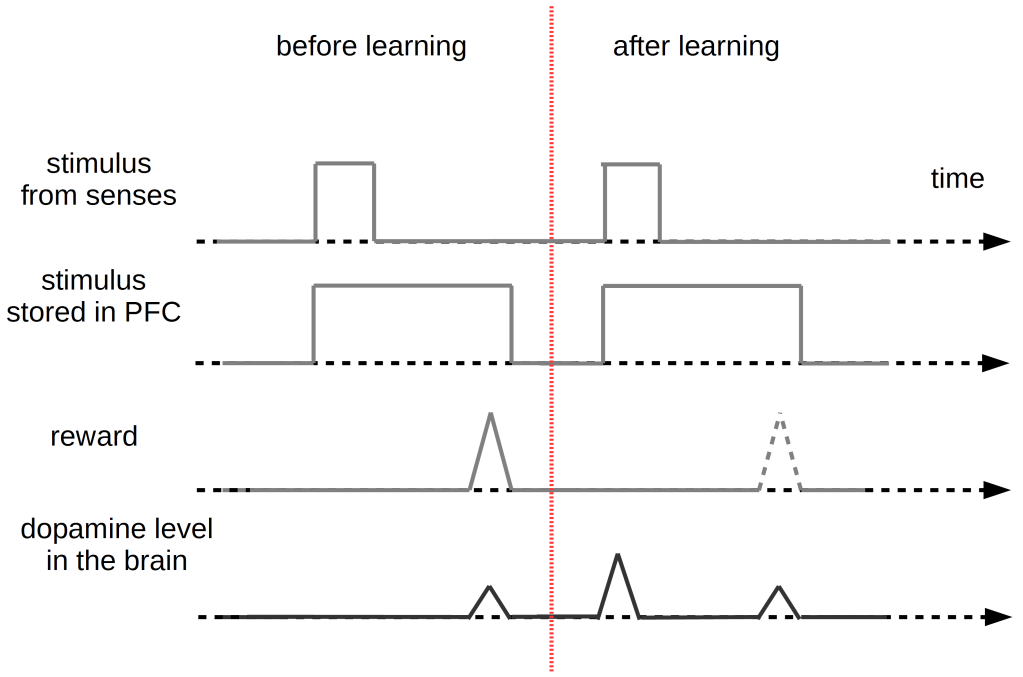



Figure 5.3: The PFC stores a temporary representation of stimuli. Bridging the gap between reward and stimulus. Adapted from Hazy et al. (2007) and O'Reilly et al. (2007).

In this case, the PFC stimulus representation was held long enough for the brain to make the association. In other words, the represented stimulus, stored in PFC, acted as if it were the perceived stimulus coming from the dog's senses.

As previously mentioned, Q-Learning and temporal differences (TD) methods, which originate from the behaviorist and cybernetic tradition of cognitive science, tend to perform poorly at this class of problems (Bakker et al., 2003). An interesting alternative from the realm of symbolic AI could be Learning Classifier Systems (LCS) (Booker et al., 1989). According to Holland et al. (2000), LCS are rule based systems which were initially intended to be used as a framework that uses genetic algorithms to study learning in condition/action, rule-based systems. In order to solve the credit assignment problem it uses an algorithm known as *bucket brigade*, which transfers part of the "strength" of the selected classifier to those with matching messages. However, Dorigo and Bersini (1994) has shown that the original bucket brigade is analogous in function and capability to Q-Learning. In other words, it is poor at solving non-markovian problems. Even though the internal messages in LCS provide the system with some level of short-term memory capability, attempts at using this mechanism for non-markovian problems suggest it leads to the eventual degradation of system performance (Smith et al., 2000). And since more recent implementations such as the XCS (initially proposed by Wilson (1995)) have no memory mechanism even in the form of internal states (Zang et al., 2014), we have therefore opted to draw inspiration from the field of computational neuroscience.

However, traditional artificial neural networks, such as MLPs (multi layer perceptrons) and recurrent neural networks, are known to be bad at establishing a link between longer time lapses, since backpropagated errors tend to either explode or exponentially decay during training (Pérez-Ortiz et al., 2003).

Computational models successful at solving this kind of problem are usually those based on gating mechanisms. A gating mechanism is responsible for holding stimuli

in what is called a "short-term memory". The information held in memory can then affect both action selection at a given instant and the learning process alike. Examples of algorithms using this sort of gating mechanism are the Long-Short Term Memory recurrent neural network (Bakker, 2002), and the aforementioned PBWM mechanism.

The choice of PBWM mechanism as inspiration for GLAS was motivated by the possibility of abstracting its core functionality into an algorithm able to perform a gated action selection learning, while at the same time keeping this learned knowledge in a human-readable form.

## 5.3 Methods

Before getting into the algorithms, we must first define a simplified problem to serve as a benchmark.

In order to study working memory in a controlled environment, a number of tasks were devised to demonstrate the demands of our brain's executive system. Examples of such tasks are the AX-CPT (AX Continuous Performance Task, widely studied in humans) and a more complex variant called 1-2-AX task (Hazy et al., 2007).

In the 1-2-AX task, a person sits down on a chair with two buttons, one to the left and another to the right. A sequence of visual stimuli is shown to this person who, in turn, must either press the right button (R) or the left one (L). Possible stimuli are: 1, 2, A, B, X and Y.

At each choice of action, pressing the correct button (according to a hidden set of rules) produces a positive reward, whilst pressing the incorrect button produces a negative reward. The challenge is to find out which button to press, given not only the current stimulus, but also a recent history of stimuli.

By the end of the experiment, if the subject's executive functions are working correctly, he should be able to learn the following rules about the 1-2-AX task:

- If the last number I saw was a 1, I must press 'R' when I see an 'X' after an 'A'
- If the last number I saw was a 2, I must press 'R' when I see a 'Y' after a 'B'
- In all other cases, I should press 'L'

An example of applying those rules to a given sequence of stimuli can be seen in Table 5.1. Notice how the subject should ignore 'Y' at step $i = 5$ by pressing 'L' instead of 'R', and correctly presses 'R' at step $i = 7$.

Table 5.1: Short example stimuli with correct action choices for the 1-2-AX working memory task.

| **i** | $\mathbf{s_i}$ | $\mathbf{a_i}$ |
|---|---|---|
| 0 | 1 | L |
| 1 | B | L |
| 2 | A | L |
| 3 | X | R |
| 4 | 2 | L |
| 5 | Y | L |
| 6 | B | L |
| 7 | Y | R |

Performing this task, even when knowing the rules, is not trivial. The subject must keep in mind at all time which number was last seen, while at the same time paying attention to the inner task sequences (AX or BY). Learning those rules in the first place is even harder[1].

We are going to use this 1-2-AX task to help us briefly explain the PBWM mechanism, and then to show how GLAS works while solving the same problem.

[1]Parallels to this kind of problem can be found in many real world engineering problems. For instance, in assistive technology, a robotic wheel chair system might need to learn that the patient chooses to go to the kitchen only on Tuesdays and Thursdays, and only if he has a visit, or there is a particular program on TV, respectively. With this information, the wheelchair system could make automated suggestions to the user in order to help him around the house (Raizer, Rohmer, Paraense and Gudwin, 2013a). Other potential applications are the development of artificial intelligent agents for video games, traffic control and mobile robotics in general.

### 5.3.1 The PBWM Mechanism

As the name suggests, the PBWM mechanism is a model for how working memory operates given the interactions between our prefrontal cortex and basal ganglia.

The basal ganglia are a set of older brain structures known to be responsible, among other things, for action selection and reward based learning (Chakravarthy et al., 2010; Stewart et al., 2010).

As we saw in Figure 5.3, the PFC is considered to be responsible for storing a temporary representation of relevant stimuli for action selection. However, how does it know what is relevant and what is not? According to PBWM, the basal ganglia is the structure responsible for deciding what should be gated into the PFC. It has, therefore, two important roles; deciding what should be stored in the PFC, and which action to select given a certain scenario. In this context, both functions should be learned from experience.

Figure 5.4 presents a snapshot of Leabra's PBWM neural network structure while solving the 1-2-AX working memory task.

As can be seen in this picture, the PBWM mechanism is based on a group of interconnected neural networks, which play the roles of input, output, hidden neural layers, PFC representations and Basal Ganglia structure. The mechanism behind Leabra's neural network is quite complex, and explaining its inner workings is out of the scope of this work. For a complete description of this mechanism, please refer to Hazy et al. (2007). Figure 5.5 shows a simplified version (diagrams were simplified for clarity) of how the already trained PBWM mechanism would work with the sequence of events seen in Table 5.1.

Square boxes represent PFC stripes (Hazy et al., 2007). Each stripe is a section of PFC, responsible for holding a single stimulus representation. The diagram depicts a unit with three stripes, but in theory they could be formed by a much longer chain of stripes. Each stripe has its own "basal ganglia component", represented here by

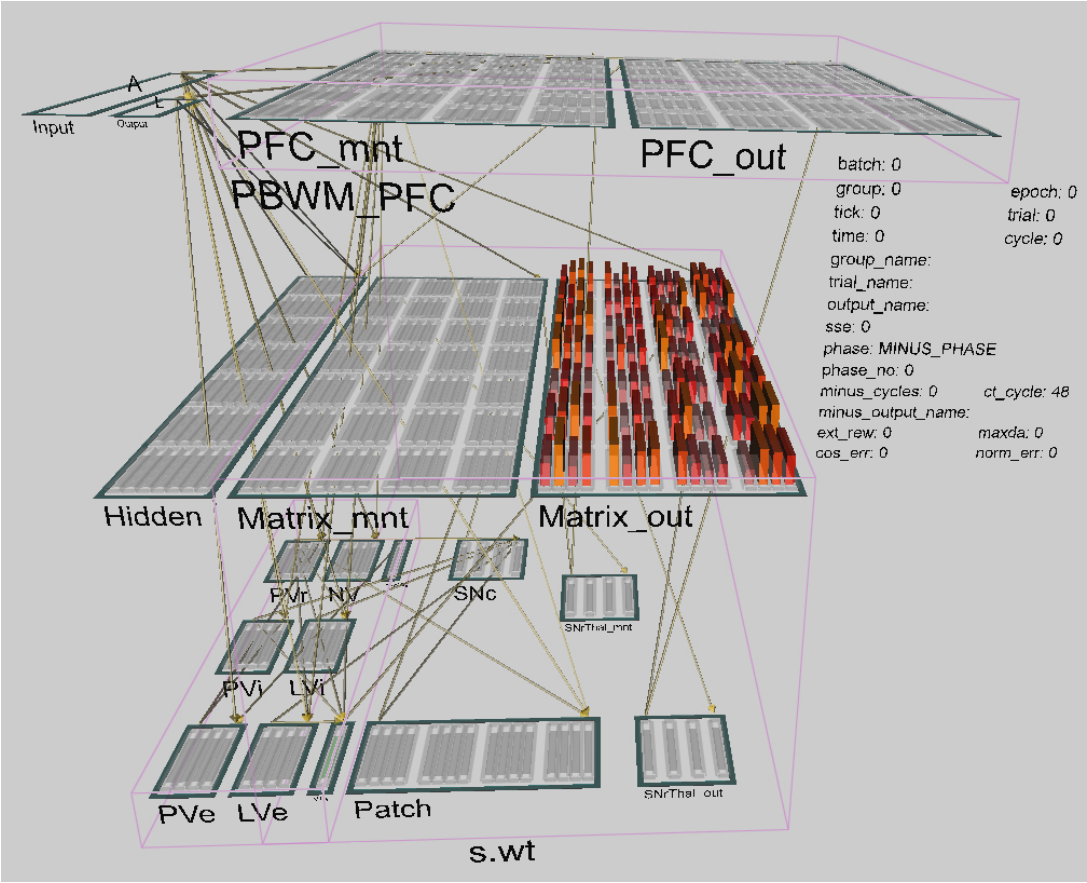



Figure 5.4: Snapshot of Leabra's PBWM neural network mechanism and connections for the 1-2-AX working memory task (O'Reilly, Munakata, Frank and Hazy, 2012).

circles. At any given time, a stimulus (the hexagon shape) is presented to the whole unit, and an action (downwards arrow) is produced.

A clear circle means that whatever comes from stimuli can fill, and replace, its neighbor's stripe content. A dark circle means its neighbor stripe is closed. Being closed means two things. First, if empty it will not allow new stimuli to enter it and will, therefore, remain empty. Second, if it is already holding a stimulus it will keep it in, preventing new stimuli from altering its contents.

As can be seen in Figure 5.5, the following would happen assuming the basal ganglia has already learned what to do:

a) No stimulus available. $b_0$ keeps $s_0$ open. $s_0$ is empty.

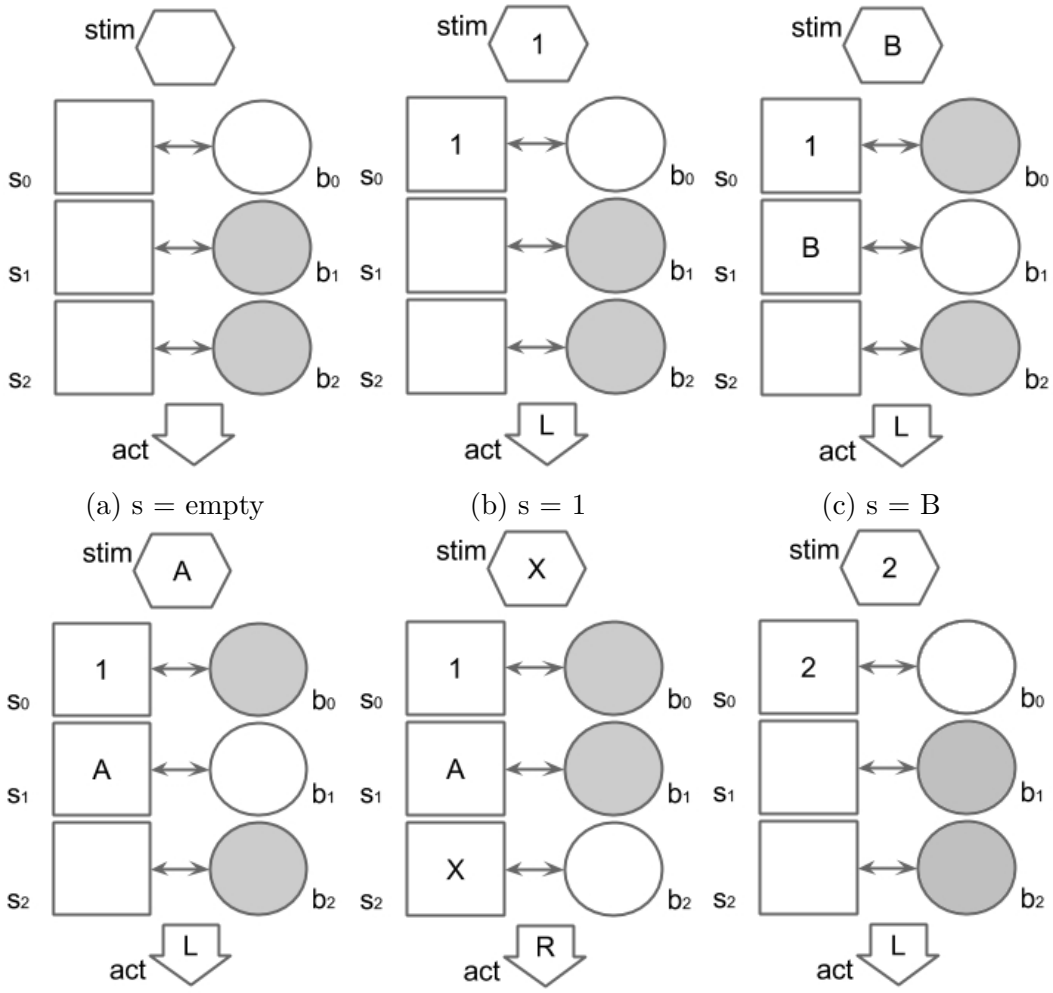



Figure 5.5: Stimuli gating and action selection for the PBWM mechanism. Simplified from Hazy et al. (2007)

b) Stimulus "1" available. $b_0$ allows stimulus to get into $s_0$.

c) $b_0$ closes gate for $s_0$. $b_1$ opens gate for $s_1$ and allows "B" to get in.

d) $b_1$ keeps $s_1$'s gate open and stimulus "A" gets in.

e) $b_1$ closes gate for $s_1$, keeping "A" inside $s_1$. $b_2$ opens gate for $s_2$, and allows "X" stimulus to get in.

f) $b_0$ detects a "2" stimulus. It opens $s_1$'s gate for it to get in and, by doing so, resets the unit.

See that the first stripe holds a special type of stimuli (for the 1-2-AX task the "1" and "2" stimuli), which are capable of resetting the unit. These are called "task stimuli" (or task sequence starters), because they identify the starting stimulus for a particular task sequence.

Up to this point we have assumed that the BG already knows what to do: what to gate in, what to gate out and which actions to select. In order to learn what to do, the PBWM model uses a reinforcement learning mechanism (a variation of temporal differences learning called PVLV - primary value and learned value (O'Reilly et al., 2007)). The net effect is that the algorithm learns working memory tasks based only on experience.

### 5.3.2 GLAS - A Gated-Learning Action Selection Mechanism

Knowledge, in the PBWM model, is stored in a group of interconnected neural networks. Because information about task sequences and gating rules are encoded in the BG's neural networks weights, it is hard for a human user to learn what the network knows.

In order to avoid this we proposed GLAS to represent knowledge in the form of a tree, with each node holding a stimulus and an action. An example for the 1-2-AX task can be seen in Figure 5.6. The tree starts at the root node, and we can see two distinct task sequences coming from it; 1-A-X and the 2-B-Y.

#### Action Selection

The algorithm starts at the root node. The current node is denoted in Figure 5.6 by a red circle surrounding a particular node. When initially presented to a sequence of stimuli, the algorithm should ignore it until it sees one of the task sequence starters, in this case either "1" or "2". If it identifies a task sequence starter in its input, it

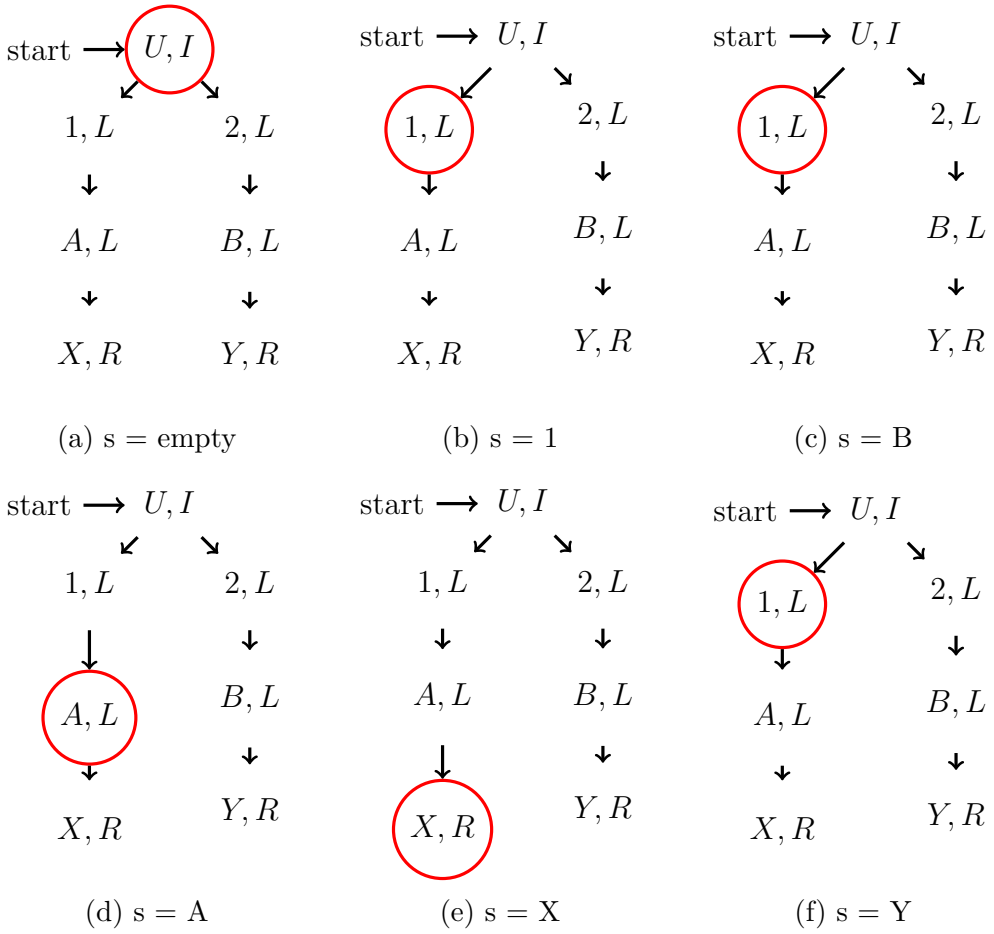



Figure 5.6: GLAS solving the 1-2-AX task. Red circle shows current node and "U" represents an initially unknown stimulus.

should move from the root node to the one related to the aforementioned input stimulus. From here on, it can only move to a child node or to another task sequence starter node (with the later taking precedence, representing the unit being reset), and only if some of them contains the currently seen stimulus. Once it reaches the last node in a task sequence (its ending node), it goes back to its respective task sequence starter. Notice that the mechanisms shown in Figure 5.5 and 5.6 produce the same action selection at each step. Pseudo-code for this algorithm can be seen in Algorithm 2.

For general cases, this action selection algorithm starts at root node in the solution tree, and at the first stimulus in the given sequence of events. The algorithm keeps

**Algorithm 2** Action Selection

```
1: function RUNAS(s)
2:     i ← 0(current stimulus)
3:     c ← 0(current node)
4:     for i = 0 to s length - 1 do
5:         if c is root then
6:             if s(i) is at a start node then
7:                 c ← start node with s(i)
8:         else if c is sequence start or intermediate then
9:             if s(i) is at start node then
10:                c ← start node with s(i)
11:            else
12:                if s(i) is at child node then
13:                    c ← next child node with s(i)
14:        else if c is sequence end then
15:            if s(i) is at start node then
16:                c ← start node with s(i)
17:            else
18:                c ← this sequence's start node
19:        actions(i) = known_actions(c)
20:    return actions
```

track of which node it is and, as new stimuli are presented to the algorithm, it behaves differently depending on what kind of node it is at any given moment. If it is at the root node, it can only go to one of the starting nodes if one of them holds the current stimulus. If at a starter or intermediate node, it can only go either to a child node or a starting node holding the current stimulus. If the current node ends a sequence two things can happen: it can go to a starting node if there is a starting node holding the current stimulus, but if this is not the case it jumps back to the node which started the current sequence. Every time a new stimulus is presented an action is selected based on which node the algorithm is currently at.

### Encoding

In the previous section we explained that GLAS represents knowledge in the form of a tree, with each node holding a stimulus and an action. We must, therefore,

encode information for three different components: the tree structure, the stimulus at each node and the action at each node.

**Encoding tree structure.** We encode the "tree structure" as an array of integers, with position $i$ in the array determining node $i$'s parent in the tree. For instance, let us take the "tree structure" seen in Figure 5.7. In this example the tree has 8 nodes, and n1 is the root node. Nodes n2 and n3 are "task sequence starters". Nodes n4 and n6 are "intermediate nodes", and nodes n5, n7 and n8 are called "task sequence ending" nodes.

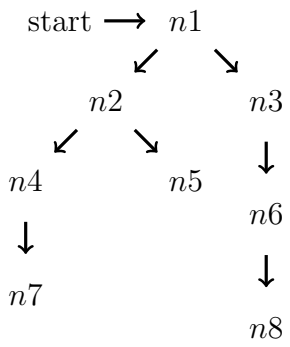



Figure 5.7: Example of a tree with 8 nodes.

We can represent this "tree structure" as the following vector of integers:

$$\begin{array}{c} \text{tree structure} = \\ i \end{array} \begin{array}{c} [\ 0\ 1\ 1\ 2\ 2\ 3\ 4\ 6\ ] \\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8 \end{array}$$

Position $i = 1$ informs that node 1 has 0 as parent, which means node 1 is the root node. Position $i = 2$ informs that node 2 has node 1 as parent. Position $i = 3$ informs that node 3 also has node 1 as parent, which means that both nodes 2 and 3 are sequence starters. Following the same logic for the remaining elements of this array allows us to encode the same information we see in Figure 5.7.

**Encoding stimuli.** Stimulus at each node is also represented as a vector of integers. A possible example for the tree presented in Figure 5.7 is as follows.

$$\begin{array}{rl} \text{stimuli} = & [\ 0\ 1\ 1\ 2\ 2\ 2\ 3\ 3\ ] \\ i & \quad 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8 \end{array}$$

In this example, the 0 at position $i = 1$ means node 1 is associated stimulus 0. The 1 in position $i = 2$ means node 2 is associated with stimulus 1, and so forth.

**Encoding actions.** Finally, action at each node is also represented as a vector of integers. A possible example for the same tree presented in Figure 5.7 is as follows.

$$\begin{array}{rl} \text{actions} = & [\ 0\ 2\ 2\ 1\ 3\ 4\ 2\ 1\ ] \\ i & \quad 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8 \end{array}$$

In this example, the 0 at position $i = 1$ means node 1 is associated with action 0. The 2 in position $i = 2$ means node 2 is associated with action 2, and so forth.

We call "solution tree", an array of integers with all three components in a row. For instance, the "solution tree" for the the previous example is the following vector of integers:

**solution tree** = [0 1 1 2 2 3 4 6  0 1 1 2 2 2 3 3  0 2 2 1 3 4 2 1]

In other words, the "solution tree" is encoded as an array of integers with three parts; tree structure, stimuli and actions. The final solution is a $3 * N$ sized vector, with N being the number of nodes: "solution tree" = ["tree structure" "stimuli" "actions"].

**Representing the solution tree for the 1-2-AX task.** For the 1-2-AX benchmark task, we have used the following codification with "U" representing an unknown stimulus and "I" an ignore action, as seen in Table 5.2. There are a total of seven stimuli, including the unknown one, and three actions, including the ignore one. The lines named "int" are the indexes representing each stimulus or action.

Table 5.2: Codification for the 1-2-AX benchmark task

For stimuli:

| stim | U | 1 | 2 | A | B | X | Y |
|---|---|---|---|---|---|---|---|
| int | 0 | 1 | 2 | 3 | 4 | 5 | 6 |

For actions:

| act | I | L | R |
|---|---|---|---|
| int | 0 | 1 | 2 |

As an example, a solution tree for the 1-2-AX task is as follows.

- tree structure = [0 1 1 2 3 4 5]
- stimuli = [0 1 2 3 4 5 6]
- actions = [0 1 1 1 1 2 2]

**solution tree** = [0 1 1 2 3 4 5  0 1 2 3 4 5 6  0 1 1 1 1 2 2]

### Learning

It is important to notice that the action selection algorithm is fixed (follows a fixed set off rules). The only thing that changes is how it deals with stimuli (what to gate in, gate out and which action to select), which is defined by the solution tree.

Therefore, the learning process in GLAS consists on finding the solution tree which maximizes reward given a sequence of events. Each event is composed of a current stimulus, a selected action and a reward.

By using the same coding presented in Table 5.2, a possible sequence of events is shown in Table 5.3. For each event "i", there is a stimulus $\mathbf{s_i}$, a selected action $\mathbf{a_i}$ and a respective reward $\mathbf{r_i}$. Given rewards can be of values "1", when it chooses the correct button for non-ending cases, and "-1" for when it pushes the wrong one. Rewards are "10", when it chooses the correct button for sequence ending cases, and "-10" when it makes the wrong choice[2].

[2]We have also experimented with values between -1 and 1, which produced the same results. We have arbitrarily chosen to use integer numbers between -10 and 10 for visualization purposes.

Table 5.3: Example of sequence of events used to learn solutions for the 1-2-AX working memory task. Rewards $r_i$ are given by the environment.

| **i** | $\mathbf{s_i}$ | $\mathbf{a_i}$ | $\mathbf{r_i}$ |
|---|---|---|---|
| 0 | 1 | 1 | 1 |
| 1 | 4 | 1 | 1 |
| 2 | 3 | 1 | 1 |
| 3 | 5 | 2 | 10 |
| 4 | 2 | 1 | 1 |
| 5 | 3 | 1 | 1 |
| 6 | 4 | 1 | 1 |
| 7 | 6 | 2 | 10 |
| 8 | 1 | 2 | -1 |
| 9 | 4 | 1 | 1 |
| 10 | 3 | 1 | 1 |
| 11 | 5 | 1 | -10 |
| 12 | 2 | 1 | 1 |
| 13 | 3 | 0 | -1 |
| 14 | 4 | 1 | 1 |

| **i** | $\mathbf{s_i}$ | $\mathbf{a_i}$ | $\mathbf{r_i}$ |
|---|---|---|---|
| 15 | 6 | 2 | 10 |
| 16 | 2 | 1 | 1 |
| 17 | 4 | 1 | 1 |
| 18 | 3 | 1 | 1 |
| 19 | 5 | 1 | 1 |
| 20 | 1 | 1 | 1 |
| 21 | 5 | 1 | 1 |
| 22 | 4 | 1 | 1 |
| 23 | 6 | 1 | 1 |
| 24 | 2 | 1 | 1 |
| 25 | 3 | 1 | 1 |
| 26 | 3 | 1 | 1 |
| 27 | 6 | 2 | -10 |

Then, we search for a tree maximizing the total accumulated reward by the algorithm as it is presented to the given sequence. In other words, this version of the GLAS algorithm is currently an off-line learning algorithm. A fixed sequence of events must be presented to it in order for it to learn a new solution. Much like how an MLP (Multi Layer Perceptron) neural network learns in batch mode. In the next chapter we will propose an improvement in the algorithm to make it an on-line algorithm.

In this work, we used a traditional genetic algorithm (GA) to explore the space of possible solutions. We named the most important parameters as:

$\alpha$ : population size

$\mu$ : number of parents

$\lambda$ : number of offspring

$n_g$ : number of generations

The GA was implemented using Opt4J framework (Lukasiewycz et al., 2011) for meta-heuristic optimization. Chromosome codification was defined as explained in Section 5.3.2, and the GA's operands and fitness function were implemented as follows.

**Operands.** Mutation is slightly different for each part of the chromosome. For actions, it is a random variation between 0 and the number of known actions available (including the ignore case). For stimuli it is the same, but in the interval ranging from 0 to the number of known stimuli (including the unknown case). For structure, each position is only allowed to vary in a range that produces a valid tree. We used the default one-point crossover (Russell and Norvig, 2010), which consists on selecting a single crossover point on both parents, and all genes beyond that point in either individual is swapped between the two parents, and the resulting individuals are the children.

**Fitness.** Each individual in the population goes through the same sequence of events, using the knowledge in its solution tree to choose the best action at each time step. If it chooses the same action in the current event, it accumulates its reward (which could be positive or negative). If it chooses a different action, it cannot judge the merit of the action it has chosen and, therefore, ignores the reward at the present event and moves on. The total final fitness is calculated as follows.

Given:

$\phi$ : total fitness

$N$ : number of nodes

$n_g$ : number of generations

$\rho$ : accumulated reward

$n_e$ : number of events in the sequence

$n_i$ : number of intermediate nodes in the history of nodes

$\iota$ : reward to avoid staying in intermediate nodes[3]

To calculate $\iota$, we first present the sequence to a given individual, and record its history of nodes. At each time step, it should be at a particular node from the solution tree. Whenever the current node is an intermediate node, we add "1" to $n_i$.

Expression 5.1 shows how to calculate the reward to those individuals who avoid staying too long in intermediate nodes.

$$\iota = 1 - \frac{n_i}{n_e} \tag{5.1}$$

Finally, Expression 5.2 gives the final fitness for a particular individual.

$$\phi = \rho + \iota - N \tag{5.2}$$

There is a penalty for larger trees (larger N), which acts as an incentive for more parsimonious solutions.

Pseudo-code for the learning component of GLAS is shown in Algorithm 3. Which is a standard genetic algorithm procedure (Russell and Norvig, 2010).

In *getFitness()*, each individual in the population is passed to *runAS()*, producing a sequence of chosen actions. This list of chosen actions is then used to calculate the individual's fitness based on Expressions 5.1 and 5.2.

We must stress, however, that the learning component illustrated in Algorithm 3 could be carried out by other optimization algorithms. Therefore, this pseudo-code should be taken as reference for how we solved the problem of finding an appropriate solution, and not as a set of necessary rules that must be followed.

[3]Adding this reward helped the algorithm escape some local maxima by encouraging individuals going through whole task sequences more often.

**Algorithm 3** Learning from sequence

```
1: function LEARN(events sequence, N)
2:     population ← α random valid solutions with N nodes
3:     for g = 1 to n_g do
4:         fitness ← getFitness(population, events)
5:         population ← selectBestFit(population, fitness)
6:         population ← applyOperands(population)
7:     fitness ← getFitness(population, events)
8:     solution ← individual with best fitness
9:     return solution
```

## 5.4 Results

In order to train a GLAS instance with the sequence of events given in Table 5.3, we used the following parameters (other parameters are the default ones): $\alpha = 100$, $\mu = 2$ , $\lambda = 2$, $n_g = 1000$.

We ran the learning algorithm for different numbers of nodes, ranging from N=2 to N=9. The result, in fitness, for the best solution found with each number of nodes can be seen in Figure 5.8. As the number of nodes grows from 2 to 7, there is an increase in the resulting fitness for the best found solution. After that, if the algorithm tries to add more nodes to the solution fitness starts to decrease, due to the effect of N in Expression 5.2, as we see in Figure 5.8 for N = 8 and N = 9. This is an indication that the best overall found solution, given this sequence of events, is the one with N=7.

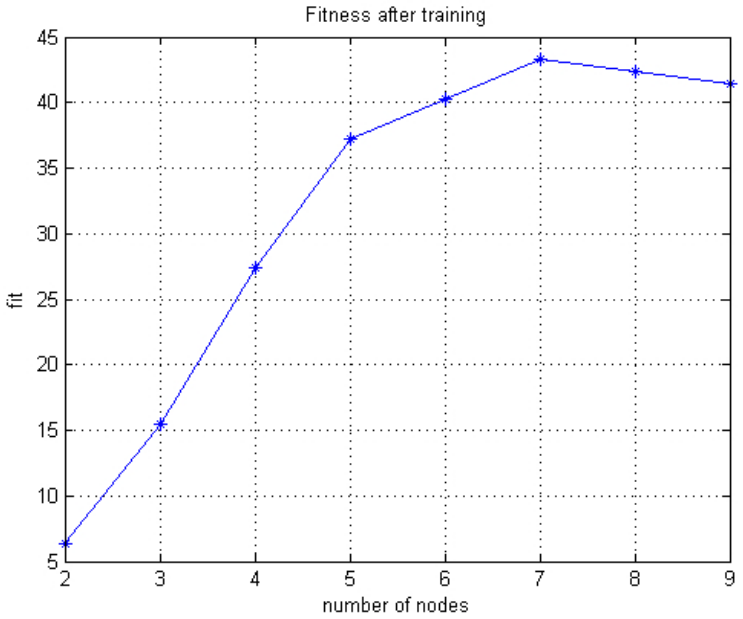
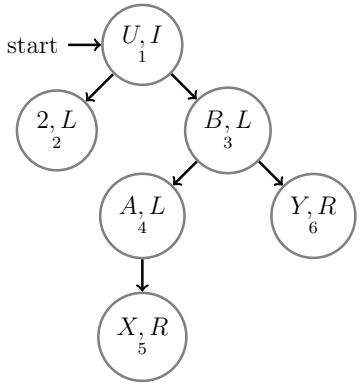
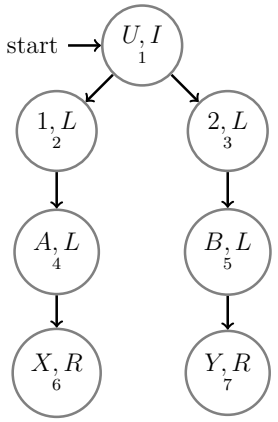



Figure 5.8: GLAS learning the 1-2-AX working memory task for different numbers of nodes.

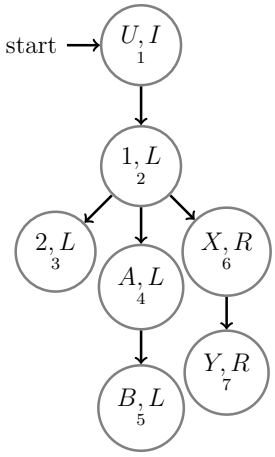
In the following figures we can see examples of solutions found by the GA. Each node is denoted by a gray circle with its number written at its lower center. Inside each circle and to the left, we see the stimulus that activates the given node, while to the right we can see the action it performs. Figure 5.9a shows the best solution found for N=6. Figures 5.9b and 5.9c show two possible solutions for N=7. Notice how fitness is slightly higher for the solution described in Figure 5.9b. At last, Figure 5.9d shows the best solution found for N=8, which is very similar in functionality to the best one for N=7, but has a penalty for having more nodes in comparison.



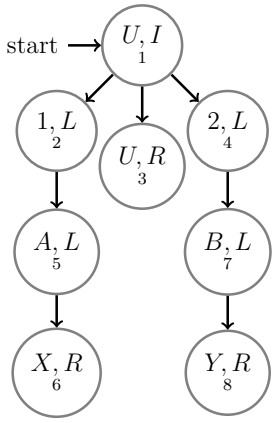

(a) Best found solution for $N = 6$ ($Fitness = 40.2857$).




(b) Best found solution for $N = 7$ ($Fitness = 43.2500$).




(c) Alternative solution for $N = 7$ ($Fitness = 43.1250$).




(d) Best found solution for $N = 8$ ($Fitness = 42.3333$).

Figure 5.9: Results for the learning algorithm with different number of nodes. "U" represents an initially unknown stimulus and numbers identify the nodes' positions in the vector of integers which encodes the tree structure.

## 5.5 Discussion

This GLAS algorithm was inspired by a computational neuroscience model of how the PFC and BG interact in order to learn action selection from stimuli and late rewards. Abstracting this mechanism as an action selection algorithm and a knowledge tree working together provided the following benefits:

- What the algorithm has learned is now in interpretable human readable form.
- As a combinatorial task, it can now be tackled by a number of optimization techniques specialized in solving this class of problems.
- The use of GA could ease its application to multi-objective problems.
- New heuristics could be specifically designed to modify the fitness function in order to improve training for different applications.

We are particularly interested in further investigating how different heuristics would influence the algorithm's capacity to correlate stimuli far apart in time. Some of the drawbacks we identify in this approach are:

- No guarantee to find the best possible solution.
- Curse of dimensionality. A larger number of stimuli and actions can make this approach unfeasible depending on how big the space to be explored is.

Both drawbacks are common when dealing with combinatorial optimization problems, and should always be taken into consideration whenever choosing GLAS, or similar mechanisms, to solve a specific problem.

At this point, GLAS is still a stand alone algorithm. In order for it to contribute to the cognitive architecture as a whole it must first be integrated into it. That issue will be addressed in the following chapter.

# Chapter 6

# On-line GLAS in a Biologically Inspired Cognitive Architecture

"For the things we have to learn before we can do them, we learn by doing them."

Aristotle.

## 6.1 Introduction

In the previous chapter we described GLAS, a new algorithm for action selection in the context of intelligent agents capable of learning from sparse and late rewards. This algorithm has the potential to provide our cognitive architecture with the ability to learn from experience, and select actions based not only on the current state of the world, but also on future implications of those actions. However, hitherto GLAS was still a stand alone algorithm. Therefore, in order for it to contribute to the cognitive architecture as a whole it must first be integrated into the triune cognitive architecture.

In this chapter we describe how GLAS was embedded in the triune cognitive architecure as a part of its Central Executive module. To do so we have created supporting codelets around GLAS in order to develop an iterative learning algorithm, able to learn a given task from scratch by interacting with the environment.

Until now GLAS was able to learn in batch mode, in the sense that a fixed sequence of events was given to its learner, and GLAS would then do the best it could to infer a solution tree adequate for describing the task hidden in the given sequence. As stated by Raizer and Gudwin (2014), the problem of this approach is that it assumes that the agent performing the action selection that produced this given sequence was competent enough to explore the space of possible actions. In other words, the agent already have an idea of what it should do in the environment and, even making a few mistakes, it is able to produce a sequence with useful information for GLAS to work with.

This is not necessarily the case, however, if an agent starts from scratch. In this case, the agent might start with random action selection, or some predefined initial rule to deal with incoming stimuli, that could produce inconsistent behavior.

## 6.2 On-line GLAS with a triune cognitive architecture

To solve this problem we have integrated GLAS into our cognitive architecture, and created supporting codelets around GLAS in order to develop an iterative learning algorithm able to learn a given task from scratch by interacting with the environment. A diagram describing this assembly can be seen in Figure 6.1. There are three executive codelets, namely "Sequence Builder Codelet", "Action Selection Codelet" and "Learner Codelet". There are also two codelets in body interface, named "Sensor Codelet" and "Actuator Codelet", which are responsible for interfacing with the environment. Codelets share information through memory objects, denoted here as light-gray boxes.

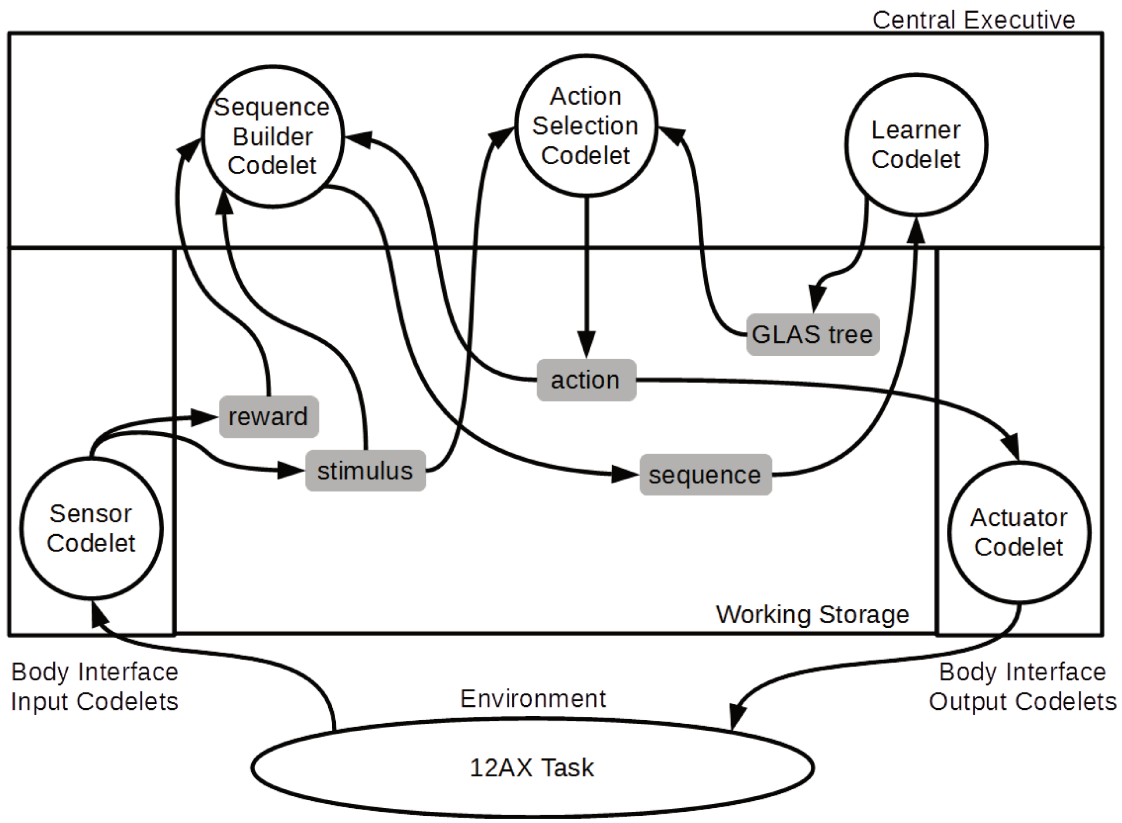



Figure 6.1: Diagram illustrating how GLAS codelets interact with the Cognitive Architecture's Structures.

Memory Objects used here have the following functions:

- stimulus: holds information about which stimulus was last perceived by the agent
- reward: registers rewards for each action selection
- action: the action selected by the agent when presented to a given stimulus
- sequence: stores a sequence of events (i.e., a list of stimuli, actions and rewards)
- GLAStree: stores the best solution tree found until the present moment

### 6.2.1 GLAS Codelets

Here we provide a more detailed description of the developed executive codelets, which where implemented as part of the Central Executive module described in Figure 3.4. The environment is an automatic procedure that produces random stimuli to the agent's sensors, and returns rewards for the response actions. Looking at the central executive as a black box for the moment, the sensor codelet captures the provided stimulus, and updates the respective stimulus memory object. Provided with this new stimulus, the executive produces an action, which is then updated into the action selection memory object. The actuator codelet then sends this chosen action the environment, which in turn produces a value of reward. Finally, this reward is read by the sensor codelet, which updates the value stored inside the reward memory object.

The following algorithms for each executive codelet describe in more detail how this action selection process takes place[1]. The first time each algorithm runs in the system (if it is its "First run") it must, before anything else, instantiate the GLAS methods it needs to perform its function.

[1]Input and Output are lists of Memory Objects, as described in Section 3.5, and proc() is a method running in a loop within the codelet's thread.

**Action Selection Codelet** This codelet has two inputs and one output. It reads new stimuli from "Stimulus Memory Object" and writes a chosen action in "Action Memory Object". In order to decide on which action to choose, it uses the solution tree provided by "Solution Tree Memory Object". Whenever a new solution three is provided in "Solution Tree Memory Object", it promptly updates its action selection mechanism.

**Algorithm 4** ActionSelectionCodelet

**Input:**
1: Solution Tree Memory Object
2: Stimulus Memory Object
**Output:**
3: Action Memory Object
4: **function** PROC()
5: **if** First run **then**
6: Create a new GlasActionSelection based on Initial Solution Tree
7: **if** New Solution Tree detected **then**
8: Update Solution Tree
9: Update stimulus from input memory object
10: Select an action with GlasActionSelection based on input stimulus
11: Update action in output memory object

**Learner Codelet** The learner codelet runs in parallel with action selection. Its function is to update the solution tree in its output "Solution Tree Memory Object", whenever there are enough new events in its input "Sequence of Events Memory Object". For this implementation, a sequence of new events is considered to be long enough if it has at least $N_e$ new events, with $N_e$ being defined by the user [2].

Every time the Learner Codelet produces an updated solution three is called an "iteration" for the learning process. Therefore, each iteration is composed by $N_e$ events.

[2]We are currently working on ways for the algorithm to automatically decide on a proper value for $N_e$, based on the amount of new information present in the recent history of events.

**Algorithm 5** LearnerCodelet

**Input:**
Sequence of Events Memory Object
**Output:**
Solution Tree Memory Object
**function** PROC()
    **if** First run **then**
        Create a new GlasLearner
    **if** New events detected **then**
        Update Sequence of Events
    **if** New sequence is long enough **then**
        Learn new tree with GlasLearner based on new events
        Update Solution Tree Memory Object in Output List

**Sequence Builder Codelet** This codelet has three inputs and a single output. It monitors the contents in "Stimulus Memory Object', "Action Memory Object" and "Reward Memory Object", and bundle then together as a single event every time they are updated, as defined in Chapter 5.

Once a new event is detected, it updates the sequence of events stored in "Sequence of Events Memory Object".

**Algorithm 6** SequenceBuilderCodelet

**Input:**
Stimulus Memory Object
Action Memory Object
Reward Memory Object
**Output:**
Sequence of Events Memory Object
**function** PROC()
    **if** New stimulus, action and reward detected **then**
        Create a new event: $(stimulus, action, reward)$
        Update Sequence of Events Memory Object in Output List with new event

## 6.3 Results

For the experiments, an instance of the learning algorithm was created as described in Section 6.2. We have defined the minimum number of new events for

learning to be $N_e = 100$. This number was chosen arbitrarily, in order to provide enough samples for the learning algorithm to work with. For better comparing solutions from different iterations, we have normalized the fitness of each solution according to the following Equation 6.1.

$$F_i = f_i / S_i \tag{6.1}$$

Where $F_i$ is the normalized fitness for iteration $i$, $f_i$ is the original fitness at iteration $i$ for the best found solution tree, and $S_i$ is the sum of all absolute reward values of the given sequence.

$$S_i = \sum_{j=1}^{N_e} abs(r_j) \tag{6.2}$$

With $abs(r_j)$ being the absolute value of reward at event $j$ in the sequence used for learning. This way, normalized fitness ranges from -1 to +1, as can be seen in Figure 6.2.

At each iteration, the learning algorithm must also learn the number of nodes "N" for the optimized tree. In Figure 6.2, the continuous line shows an increasing fitness value for the best found solutions. It also shows that the initial solution has fewer nodes ( $N = 2$ nodes in this case), but was not able to produce quality action selection. From iteration 2 onward we see that the algorithm has found $N = 7$ to be an interesting size for the solution tree. Finally, the algorithm seems to converge around the $5^{th}$ iteration[3], producing the solution previously presented in Chapter 5.

[3]Even if the solution were to select all the correct actions, the Normalized Fitness $F_i$ would still not reach 1 because there are some penalties being applied at the calculation of $f_i$, such as the penalty for having a higher number of nodes, as described in Raizer and Gudwin (2014).

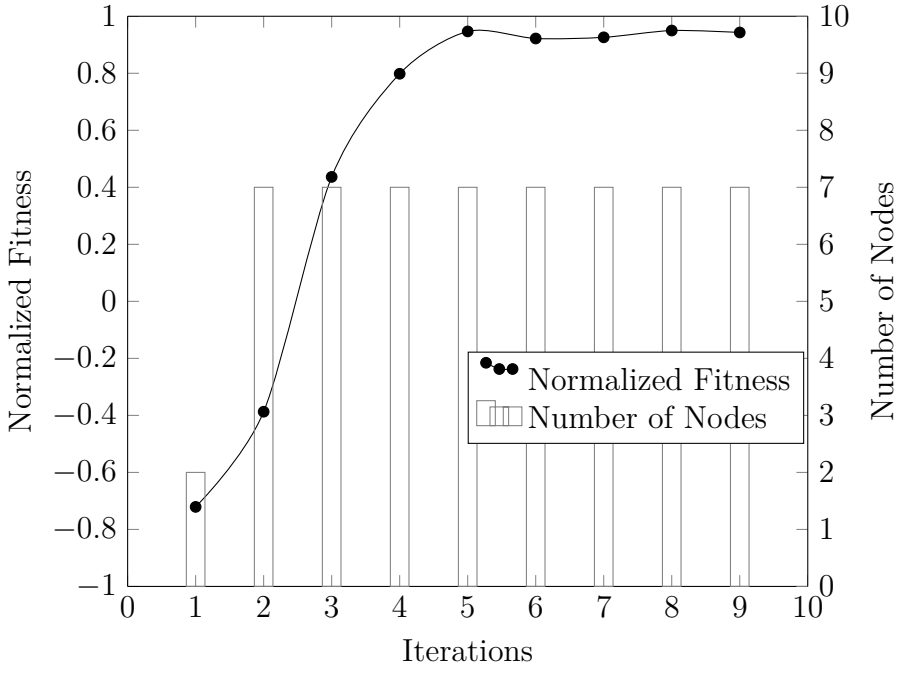



Figure 6.2: Learning the 12AX Working Memory task by interacting with the simulated environment. Each iteration is a new solution tree found by the algorithm after being presented to $N_e$ events.

## 6.4 Discussion

This chapter has presented a method for embedding GLAS - a Gated-Learning Action Selection algorithm - into a triune biologically inspired cognitive architecture. To do so under the constraints imposed by the architecture itself, we have developed specific codelets for dealing with each aspect of the action-selection and learning process.

Results have shown that the developed algorithm is able to learn iteratively, by interacting with the simulated environment, and successfully solve the task at hand.

As it is, one drawback of this approach is scalability. It is still not clear how to scale this system to learn different tasks and integrate this knowledge into a coherent behavioral framework. In order to solve this issue, we are currently investigating its integration with the architecture's modified behavior network.

Future work should, therefore, encompass this algorithm's scalability for more

general agents. We also intend to validate this work with real world problems, such as in the development of intelligence for controlling robotic agents and video game characters.

GLAS adds to the cognitive architecture the following cognitive skills of the paleomammalian level:

**Paleomammalian**

- CS3-7 : Selection of what needs to be stored in memory.

  BP3-7 : Current node represents content gated in (stored) in working memory.

- CS4-1 : Trial and error learning. Re-evaluation of selected objects or events.

  BP4-1 : The tasks for sequences 1AX and 2BY are learned by trial and error, with bad choices represented by negative rewards and good choices by positive rewards.

- CS4-3 : Evaluation of the performance in the achievement of a single goal.

  BP4-3 : Rewards for selecting the correct action at sequence endings are used to get better at it.

# Chapter 7

# Conclusions

In this study, we have embedded a neuroscience inspired cognitive architecture with the executive functions for deliberation, in the form of a modified behavior network, and for learning, in the form of the GLAS algorithm, which is able to learn from late rewards and perform reactive and deliberative action-selection. This chapter summarizes what has been done in terms of research and development, which assumptions were made and indicate future work that could be done.

## 7.1 Publications

This work has made contributions to the state of the art in cognitive architectures and learning mechanisms. These contributions are documented in the present text, and also in a number of international congress articles, journal papers, a submitted patent and registered software, as follows:

- Raizer, K., Paraense, A. L. O. and Gudwin, R. R. (2011). *A cognitive neuroscience inspired codelet-based cognitive architecture for the control of artificial creatures with incremental levels of machine consciousness*, Symposium Proceedings at the AISB11 Convention. Machine Consciousness 2011: Self, Integration and Explanation, UK Society for the Study of Artificial Intelligence and Simulation of Behaviour, York, United Kingdom (Raizer et al., 2011).

- Raizer, K., Paraense, A. and Gudwin, R. (2012). *A cognitive architecture with incremental levels of machine consciousness inspired by cognitive neuroscience*, International Journal of Machine Consciousness (Raizer et al., 2012).

- Raizer, K., Rohmer, E., Paraense, A. and Gudwin, R. (2013). *Effects of behavior network as a suggestion system to assist BCI users*, IEEE 2013 Symposium on Computational Intelligence in Rehabilitation and Assistive Technologies (CIRAT) (Raizer, Rohmer, Paraense and Gudwin, 2013a).

- Cognitive Architecture made available as Open Source under LGPL licence.

- GLAS algorithm made available as Open Source under LGPL licence.

- Raizer, K. and Gudwin, R. (2014). *A Neuroscience Inspired Gated Learning Action Selection Mechanism*, Presented at : The 2014 Annual International Conference on Biologically Inspired Cognitive Architectures, Fifth Annual Meeting of the BICA Society. Held on November 7-9 (Friday-Sunday), at the Massachusetts Institute of Technology, Cambridge, MA 02139, United States (Raizer et al., 2014).

- Raizer, K. and Gudwin, R. (2014). *GLAS: A Neuroscience Inspired Gated Learning Action Selection Mechanism*, Published at : The International Journal of Biologically Inspired Cognitive Architectures, $ISSN$ : $2212 - 683X$ (Raizer and Gudwin, 2014).

- Raizer, K. and Gudwin, R. (2015). *On-line Gated Learning Action Selection in a Biologically Inspired Cognitive Architecture*, Submited to the 2nd BRAINN Congress. CEPID BRAINN - Brazilian Institute of Neuroscience and Neurotechnology

- Raizer, K., Rohmer, E., Paraense, A., Gudwin, R. and Cardozo, E. (2013). Pending patent: *Method for the development of a suggestion agent with behavior network for assistive technology* (Raizer, Rohmer, Paraense, Gudwin and Cardozo, 2013).

- Raizer, K., Rohmer, E., Paraense, A., Gudwin, R. and (2013). Registered Software: *Intelligent Software Agent Applied to Assistive Technology* (Raizer, Rohmer, Paraense and Gudwin, 2013b).

## 7.2 Main Contributions

This research makes contributions in three topics of artificial cognition, in the form of: (i) cognitive architectures and roadmap for artificial agents, (ii) artificial agents for assistive technology, (iii) executive functions in machine learning. In the following sections we discuss the main contributions in each topic, and point out interesting lines of research for future work.

### 7.2.1 Cognitive Architecture for Artificial Agents

For the control of autonomous artificial agents, we started the development of a biologically inspired cognitive architecture. Related research in the field of cognitive architectures can be found in recent works by Franklin et al. (2014) with their LIDA architecture, Anderson et al. (2010) with ACT-R, O'Reilly, Munakata, Frank and Hazy (2012) with Leabra-Emergent, and Laird et al. (2012)'s SOAR. Architectures such as LIDA, ACT-R and Leabra-Emergent are strongly biased towards biological plausibility, and are meant to be used as testbeds for the study of cognitive processes. SOAR is also bioinspired in its origin, but its developers do not necessarily refrain from using mechanisms which are not biologically plausible, if by doing so a better architecture can be achieved (Langley et al., 2009). Motivation for developing our own cognitive architecture were twofold: First, the possibility of developing a modular architecture with a unified mechanism for processing and information storage at its core, and second, the proposal of a roadmap for the development of artificial agents.

In the work of Raizer et al. (2012) we used MacLean (1990)'s triune brain model for the vertebrate brain and Arrabales et al. (2013)'s ConsScale to propose a roadmap for the development of artificial agents and frameworks. The assumption is that this approach should guarantee an embodied and situated agent at every development stage for an artificial agent using the architecture, while providing a quantifiable measure of its cognitive capability.

Major contributions in the field of cognitive architectures and artificial agents were, therefore, the modular bioinspired cognitive architecture for the development of artificial agents with a range of cognitive capabilities, and also a roadmap for the development of artificial agents with a quantified analysis of their cognitive capacity. In this field we identify a number of lines of research that could be investigated in future works:

- Continue the development of this cognitive architecture, by implementing and expanding its other modules.
- Adapt this cognitive architecture to a "cognition as a service" model, to be offered in a cloud environment. This could make it easier for other groups to use our architecture, and apply it to develop agents for web applications.
- Study new mechanisms of machine consciousness for the architecture, and compare them with existing ones such as LIDA's (Franklin et al., 2014).
- Evaluate these mechanisms for machine consciousness using behavioral experiments and also information-based methods such as Tononi (2008)'s information integration index.
- Add to the architecture mechanisms for artificial emotions, such as those described by Avila-Garcia and Canamero (2005) and Samsonovich (2013).
- Validate the architecture in other applications such as controlling mobile and humanoid robots (Puigbo et al., 2013; Schwarz et al., 2014), controlling video-

game agents (Asensio et al., 2014; Goertzel et al., 2014) and solving problems in the field of IoT (internet of things) (Gubbi et al., 2013).

While developing agents with artificial cognition, we noticed a major difficulty when trying to compare their cognitive capacities with those of other agents. For example, in the field of function optimization there is a broad range of standardized benchmarks and problems for evaluating a particular algorithm's performance (Jamil and Yang, 2013) and to compare it with others. To this date, however, there is a lack of analogous benchmarks for the evaluation and comparison of artificial agents regarding their cognitive capacity. We believe that the proposed roadmap, and its association with ConsScale, could help in mitigating this issue.

### 7.2.2 Artificial Agent for Assistive Technology

In the context of assistive technology, we have applied our triune cognitive architecture for developing an assistive agent capable of performing suggestions that might be interesting to a robotic wheelchair user in a simulated scenario. The benefit is a measurable reduction in the number of interface interactions needed for the user to reach his goals. The assumption is that an artificial agent with cognitive functions similar to those present in humans would be able to select the most appropriate action, given the user's goals and current state of the environment.

To solve this problem, we have implemented in our cognitive architecture a modified version of Maes (1989)'s behavior network. The modification consisted on adding a new "soft-preconditions" list to each behavior, which affect activation flow in the same way the original preconditions list does but does not, however, block the behavior from becoming executable if its propositions are not met. The behavior network is built in a "top-down" manner, based on information from the smart environment.

We see, therefore, two major contributions: one to the mechanism of deliberative action selection with behavior networks and another to the development of artificial

assistive agents.

Interesting lines of research to be investigated in the future are the following:

- Provide the system with the ability to learn the user's preferences with experience. Techniques of interest could be traditional reinforcement learning such as Q-Learning (Russell and Norvig, 2010), artificial neural networks (Schmidhuber, 2015; Haykin, 2009), learning classifier systems (Zang et al., 2014; Booker et al., 1989) and other bioinspired learning mechanisms (Herd et al., 2014; Hazy et al., 2007).
- Validate the system with human users in a real world scenario.
- Perform experiments with other HMI (Human Machine Interface) mechanisms, like *eye-tracking* (Majaranta and Bulling, 2014) and EMG (Electromyogram) (da Silva et al., 2015).
- Compare results from using behavior networks with those obtained with action-selection mechanisms which take into account partially observable states, such as POMDPs (Partially Observable Markov Decision Processes) (Hoey et al., 2010).
- Integrate the assistive smart environment in an internet of things context (Gubbi et al., 2013), and provide real time readings about the patient's health.

According to consulted specialists, patients with severe motor disabilities value the feeling of being able to control the automated system. Any assistive agent should therefore find a way to balance how much automation and how much direct control it should provide. Relevant insights could be found in the field of shared-control (Wang and Liu, 2014; Gandhi, 2014) for robotics and BCI (Brain Computer Interface) interaction.

### 7.2.3 Executive Functions in Machine Learning

Executive functions are future directed, based on prediction and goal oriented (Baars and Gage, 2010). In humans and higher mammals these functions are directly linked to the PFC (prefrontal cortex) which is responsible, among other things, for making it possible to organize sequences of actions in order to reach particular goals (Chersi et al., 2011). It is the PFC, and how it interacts with the rest of brain, that makes it possible for us to solve complex problems, to build high level plans and learn from experiences far apart in time.

In this context, we have developed GLAS, a Gated Learning Action Selection algorithm (Raizer and Gudwin, 2014), which is able to learn from late and sparse rewards. We drew inspiration from Leabra, a computational neuroscience model for the interaction between the PFC and BG (Basal Ganglia) structures in the human brain, which was developed by O'Reilly et al. (2007). However, Leabra consists of a group of interconnected neural networks, which makes it difficult to interpret the knowledge it contains. GLAS, on the other hand, employs a tree based representation, which can be directly read and interpreted by humans.

We see, therefore, two major contributions in this work: the development of a novel algorithm for action selection with an interpretable tree representation, and its integration with a biologically inspired cognitive architecture. Here follows a non-exhaustive list of interesting topics for future research:

- Formal analysis of GLAS in terms of scalability (Luna et al., 2014) and capacity for generalization.
- Compare GLAS's performance with different optimization algorithms for its learning mechanism, such as PSO (Particle Swarm Optimization) (Kiranyaz et al., 2014), simulated annealing (Zhu et al., 2014) or other algorithms specific for combinatory optimization.

- Compare GLAS with other gating based action selection algorithms, such as LSTM (Long Short Term Memory) recurrent neural networks (Sundermeyer et al., 2015).
- Validate GLAS with experiments in robotics and computer games.
- Evaluate GLAS's potential for solving multi-objective problems (Deb, 2014).
- Study new heuristics and fitness functions that make GLAS more efficient for specific applications.
- Development of hybrid solutions of GLAS with other algorithms such as Learning Classifier Systems (Debie et al., 2013) or the modified behavior network described by Raizer, Rohmer, Paraense and Gudwin (2013a).
- Investigate the possibility of describing GLAS as a particular case of genetic programming (Poli and Koza, 2014).

The field of computational neuroscience can provide inspiration for the development of novel algorithms with applications to engineering problems. This line of work has the potential to strengthen cooperations between diverse fields such as biomedical engineering, neuroscience, behavioral psychology, cognitive robotics, assistive technology and machine learning in general.

# Bibliography


Alvarez, J. A. and Emory, E. (2006). Executive function and the frontal lobes: A meta-analytic review, *Neuropsychology Review* **16**(1): 17–42.

Anderson, J., Lebiere, C., O'Reilly, R. and Stocco, A. (2010). Integrated cognitive architectures for robust decision making, *Technical report*, DTIC Document.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C. and Qin, Y. (2004). An integrated theory of the mind., *Psychological Review* **111**(4): 1036.

Arkin, R. C. (1998). *Behavior-based robotics*, MIT press.

Arrabales, R., Ledezma, A. and Sanchis, A. (2009a). A cognitive approach to multimodal attention, *Journal of Physical Agents* **3**(1): 53–63.

Arrabales, R., Ledezma, A. and Sanchis, A. (2009b). Assessing and Characterizing the Cognitive Power of Machine Consciousness Implementations, *AAAI Fall Symposium Series*.

Arrabales, R., Ledezma, A. and Sanchis, A. (2009c). Cera-cranium: A test bed for machine consciousness research, *Proceedings of the International Workshop on Machine Consciousness*.

Arrabales, R., Ledezma, A. and Sanchis, A. (2010a). Consscale: A pragmatic scale for measuring the level of consciousness in artificial agents, *Journal of Consciousness Studies* **17**(3-4): 131–164.

Arrabales, R., Ledezma, A. and Sanchis, A. (2010b). The Cognitive Development of Machine Consciousness Implementations, *International Journal of Machine Consciousness* **02**(02): 213.

Arrabales, R., Ledezma, A. and Sanchis, A. (2013). Characterizing and assessing human-like behavior in cognitive architectures, *in* A. Chella, R. Pirrone, R. Sorbello and K. R. Johannsdottir (eds), *Biologically Inspired Cognitive Architectures 2012*, Vol. 196 of *Advances in Intelligent Systems and Computing*, Springer Berlin Heidelberg, pp. 7–15.
**URL:** *http://dx.doi.org/10.1007/978-3-642-34274-5_2*

Arrabales, R., Muñoz, J., Ledezma, A., Gutierrez, G. and Sanchis, A. (2012). A machine consciousness approach to the design of human-like bots, *Believable Bots: Can Computers Play Like People?* p. 171.

Asensio, J. M. L., Peralta, J., Arrabales, R., Bedia, M. G., Cortez, P. and Peña, A. L. (2014). Artificial intelligence approaches for the generation and assessment of believable human-like behaviour in virtual characters, *Expert Systems with Applications* **41**(16): 7281 – 7290.
**URL:** *http://www.sciencedirect.com/science/article/pii/S0957417414002759*

Avila-Garcia, O. and Cañamero, L. (2004). Using hormonal feedback to modulate action selection in a competitive scenario, *In From Animals to Animats: Proceedings of the 8th International Conference of Adaptive Behavior (SAB04)*, MIT Press, pp. 243–252.

Avila-Garcia, O. and Canamero, L. (2005). Hormonal modulation of perception in motivation-based action selection architectures, *Procs of the Symposium on Agents that Want and Like*, SSAISB.

Baars, B. J. (1997). Some essential differences between consciousness and attention, perception, and working memory., *Consciousness and cognition* **6**(2-3): 363–71. **URL:** *http://www.ncbi.nlm.nih.gov/pubmed/9262417*

Baars, B. J. and Franklin, S. (2003). How conscious experience and working memory interact, *Trends in Cognitive Sciences* **7**(4): 166 – 172.

Baars, B. J. and Franklin, S. (2009). Consciousness is computational: the lida model of global workspace theory, *International Journal of Machine Consciousness* **01**(0101): 23.

Baars, B. J. and Gage, N. M. (2010). *Cognition, brain, and consciousness: Introduction to cognitive neuroscience*, Vol. 1, 2 edn, Academic Press, Imprint of Elsevier: 30 Corporate Drive, Suite 400, Burlington, MA 01803, USA; 525 B Street, Suite 1900, San Diego, California 92101-4495, USA; Elsevier, The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, UK.

Baddeley, A. (2003). Working memory and language: an overview, *Journal of Communication Disorders* **36**(3): 189 – 208.

Baddeley, A. D. (2002). Fractioning the Central Executive, *in* D. T. Stuss and R. Knight (eds), *Principles of frontal lobe function*, 1 edn, Oxford University Press, USA, New York, chapter 16, pp. 246–260.

Baddeley, A. D. and Hitch, G. J. (1975). Working memory, *The psychology of learning and motivation* **8**: 47–89.

Bagnell, J. A. and Schneider, J. G. (2001). Autonomous helicopter control using reinforcement learning policy search methods, *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, Vol. 2, pp. 1615–1620 vol.2.

Bakker, B. (2002). Reinforcement learning with long short-term memory, *Advances in neural information processing systems* **14**: 1475–1482.

Bakker, B., Zhumatiy, V., Gruener, G. and Schmidhuber, J. (2003). A robot that reinforcement-learns to identify and memorize important previous observations, *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, Vol. 1, IEEE, pp. 430–435.

Bargh, J. A. and Morsella, E. (2008). The unconscious mind, *Perspectives on psychological science* **3**(1): 73–79.

Bell, C. J., Shenoy, P., Chalodhorn, R. and Rao, R. P. N. (2008). Control of a humanoid robot by a noninvasive brain–computer interface in humans, *Journal of Neural Engineering* **5**(2): 214.

Bogner, M. B. (1999). *Realizing "consciousness" in software agents*, PhD thesis, The University of Memphis. AAI9949957.

Booker, L. B., Goldberg, D. E. and Holland, J. H. (1989). Classifier systems and genetic algorithms, *Artificial Intelligence* **40**(1-3): 235 – 282.
**URL:** *http://www.sciencedirect.com/science/article/pii/0004370289900507*

Braitenberg, V. (1986). *Vehicles: Experiments in synthetic psychology*, MIT press.

Chakravarthy, V. S., Joseph, D. and Bapi, R. S. (2010). What do the basal ganglia do? a modeling perspective, *Biological Cybernetics* **103**(3): 237–253.
**URL:** *http://dx.doi.org/10.1007/s00422-010-0401-y*

Chan, R. C. K., Shum, D., Toulopoulou, T. and Chen, E. Y. H. (2008). Assessment of executive functions: Review of instruments and identification of critical issues, *Archives of Clinical Neuropsychology* **23**(2): 201 – 216.
**URL:** *http://www.sciencedirect.com/science/article/pii/S0887617707001928*

Chersi, F., Ferrari, P. F. and Fogassi, L. (2011). Neuronal chains for actions in the parietal lobe: a computational model, *PloS one* **6**(11): e27652.

Cooper, R. P. (2010). Cognitive Control: Componential or Emergent?, *Topics in Cognitive Science* **2**(4): 598–613.

Cowan, R. E., Fregly, B. J., Boninger, M. L., Chan, L., Rodgers, M. M. and Reinkensmeyer, D. J. (2012). Recent trends in assistive technology for mobility, *Journal of NeuroEngineering and Rehabilitation* **9**(1): 20.

da Silva, H. P., Fairclough, S., Holzinger, A., Jacob, R. and Tan, D. (2015). Introduction to the special issue on physiological computing for human-computer interaction, *ACM Trans. Comput.-Hum. Interact.* **21**(6): 29:1–29:4.
**URL:** *http://doi.acm.org/10.1145/2688203*

da Silva, R. C. M. and Gudwin, R. R. (2010). An introductory experiment with a conscious-based autonomous vehicle, *4th Workshop in Applied Robotics and Automation*.

Deb, K. (2014). Multi-objective optimization, *in* E. K. Burke and G. Kendall (eds), *Search Methodologies*, Springer US, pp. 403–449.
**URL:** *http://dx.doi.org/10.1007/978-1-4614-6940-7_15*

Debie, E., Shafi, K., Lokan, C. and Merrick, K. (2013). Investigating differential evolution based rule discovery in learning classifier systems, *Differential Evolution (SDE), 2013 IEEE Symposium on*, pp. 77–84.

Dehaene, S. and Naccache, L. (2001). Towards a cognitive neuroscience of consciousness: basic evidence and a workspace framework, *Cognition* **79**(1-2): 1–37.
**URL:** *http://www.ncbi.nlm.nih.gov/pubmed/11164022*

Dennett, D. C. (1991). *Consciousness explained*, Vol. 1, 1 edn, Back Bay Books, Hachette Book Group USA; 237 Park Avenue, New York, NY 10017.

Dorer, K. (2010). Modeling human decision making using extended behavior networks, *in* J. Baltes, M. Lagoudakis, T. Naruse and S. Ghidary (eds), *RoboCup 2009: Robot Soccer World Cup XIII*, Vol. 5949 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 81–91.

Dorigo, M. and Bersini, H. (1994). A comparison of q-learning and classifier systems, *From Animals to Animats: Proceedings of the Third International Conference on the Simulation of Adaptive Behavior*, Vol. 3, pp. 248–255.

Dubois, D. (2007). *Constructing an agent equipped with an artificial consciousness: application to an intelligent tutoring system*, PhD thesis, PhD thesis, Université du Québec à Montréal.

Edelman, D. B., Baars, B. J. and Seth, A. K. (2005). Identifying hallmarks of consciousness in non-mammalian species, *Consciousness and Cognition* **14**(1): 169 – 187. Neurobiology of Animal Consciousness.

Edelman, G. M. (2004). *Wider than the sky: The phenomenal gift of consciousness*, Vol. 1, 1 edn, Yale University Press, 302 Temple Street; New Haven, CT; 06511-8909.

Elliott, R. (2003). Executive functions and their disorders imaging in clinical neuroscience, *British Medical Bulletin* **65**(1): 49–59.

Escolano, C., Antelis, J. M. and Minguez, J. (2012). A telepresence mobile robot controlled with a noninvasive brain–computer interface, *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* **42**(3): 793–804.

Faghihi, U. and Franklin, S. (2012). The lida model as a foundational architecture for agi, *Theoretical Foundations of Artificial General Intelligence* pp. 103–121.

Ferreira, A., Cavalieri, D. C., Filgueira, P. N. S., Silva, R. L., Filho, T. F. B. and Celeste, W. C. (2012). Sistema assistivo de interface homem-máquina.

Ferreira, A., Silva, R. L., Celeste, W. C., Bastos Filho, T. F. B. and Sarcinelli Filho, M. (2007). Human–machine interface based on muscular and brain signals applied to a robotic wheelchair, *Journal of Physics: Conference Series*, Vol. 90, IOP Publishing, p. 012094.

Frank, M. J. and Badre, D. (2012). Mechanisms of hierarchical reinforcement learning in corticostriatal circuits 1: computational analysis, *Cerebral cortex* **22**(3): 509–526.

Franklin, S. (1995). *Artificial minds*, Vol. 1, 1 edn, The MIT Press.

Franklin, S., Madl, T., D'mello, S. and Snaider, J. (2014). Lida: A systems-level architecture for cognition, emotion, and learning, *Autonomous Mental Development, IEEE Transactions on* **6**(1): 19–41.

Fuster, J. M. (2008). *The prefrontal cortex*, Vol. 1, 4 edn, Academic Press, Imprint of Elsevier: 32 Jamestown Road, London NW1 7BY, UK; 30 Corporate Drive, Suite 400, Burlington, MA 01803, USA; 525 B Street, Suite 1900, San Diego, CA 92101-4495, USA.

Gandhi, V. (2014). *Brain-computer Interfacing for Assistive Robotics: Electroencephalograms, Recurrent Quantum Neural Networks, and User-centric Graphical Interfaces*, Academic Press.

Goertzel, B., Pennachin, C. and Geisweiller, N. (2014). A brief overview of cogprime, *Engineering General Intelligence, Part 1*, Vol. 5 of *Atlantis Thinking Machines*, Atlantis Press, pp. 21–40.
**URL:** *http://dx.doi.org/10.2991/978-94-6239-027-0_2*

Goleman, D. (2006). *Emotional intelligence*, Vol. 1, 10 edn, Bantam Dell Pub Group, 1745 Broadway; New York, NY 10019.

Gubbi, J., Buyya, R., Marusic, S. and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and future directions, *Future Generation Computer Systems* **29**(7): 1645 – 1660.
**URL:** *http://www.sciencedirect.com/science/article/pii/S0167739X13000241*

Hawkins, J. and Blakeslee, S. (2005). *On intelligence*, St. Martin's Griffin.

Haykin, S. O. (2009). *Neural networks and learning machines*, Vol. 3, Pearson Education Upper Saddle River.

Hazy, T. E., Frank, M. J. and O'Reilly, R. C. (2007). Towards an executive without a homunculus: computational models of the prefrontal cortex/basal ganglia system, *Philosophical Transactions of the Royal Society B: Biological Sciences* **362**(1485): 1601–1613.

Herd, S., Szabados, A., Vinokurov, Y., Lebiere, C., Cline, A. and O'Reilly, R. C. (2014). Integrating theories of motor sequencing in the {SAL} hybrid architecture, *Biologically Inspired Cognitive Architectures* **8**(0): 100 – 108.
**URL:** *http://www.sciencedirect.com/science/article/pii/S2212683X14000139*

Hoey, J., Poupart, P., von Bertoldi, A., Craig, T., Boutilier, C. and Mihailidis, A. (2010). Automated handwashing assistance for persons with dementia using video and a partially observable markov decision process, *Computer Vision and Image Understanding* **114**(5): 503 – 519. Special issue on Intelligent Vision Systems.
**URL:** *http://www.sciencedirect.com/science/article/pii/S1077314210000354*

Hofstadter, D. and Mitchell, M. (1994). *The Copycat Project: A Model of Mental Fluidity and Analogy-making*, BasicBooks, 10, East 53rd Street, New York, NY 10022-5299, pp. 205–268.

Holland, J. H., Booker, L. B., Colombetti, M., Dorigo, M., Goldberg, D. E., Forrest, S., Riolo, R. L., Smith, R. E., Lanzi, P., Stolzmann, W. and Wilson, S. W. (2000). What is a learning classifier system?, *in* P. Lanzi, W. Stolzmann and S. W. Wilson (eds), *Learning Classifier Systems*, Vol. 1813 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 3–32.
**URL:** *http://dx.doi.org/10.1007/3-540-45027-0-1*

Holzner, C., Guger, C., Grönegress, C., Edlinger, G. and Slater, M. (2009). Using a p300 brain computer interface for smart home control, *World Congress on Medical Physics and Biomedical Engineering, September 7-12, 2009, Munich, Germany*, Springer, pp. 174–177.

Ingvar, D. H. (1984). "memory of the future": an essay on the temporal organization of conscious awareness., *Human neurobiology* **4**(3): 127–136.

Itti, L., Koch, C. and Niebur, E. (2000). A Model of Saliency-Based Visual Attention for Rapid Scene Analysis, *Journal of Socio-Economics* **29**(6): 579–586.

Jamil, M. and Yang, X. (2013). A literature survey of benchmark functions for global optimisation problems, *International Journal of Mathematical Modelling and Numerical Optimisation* **4**(2): 150–194.

Kiranyaz, S., Ince, T. and Gabbouj, M. (2014). Particle swarm optimization, *Multidimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition*, Vol. 15 of *Adaptation, Learning, and Optimization*, Springer Berlin Heidelberg, pp. 45–82.
**URL:** *http://dx.doi.org/10.1007/978-3-642-37846-1_3*

Koch, C. and Tsuchiya, N. (2007). Attention and consciousness: two distinct brain processes, *Trends in cognitive sciences* **11**(1).
**URL:** *http://www.sciencedirect.com/science/article/pii/S1364661306003032*

Kolter, J. Z. and Ng, A. Y. (2011). The stanford littledog: A learning and rapid replanning approach to quadruped locomotion, *The International Journal of Robotics Research* **30**(2): 150–174.
**URL:** *http://ijr.sagepub.com/content/30/2/150.abstract*

Kuffner Jr., J. J. and Steven M. LaValle, S. M. (2000). Rrt-connect: An efficient approach to single-query path planning, *Proceedings of the 2000 IEEE International Conference on Robotics and Automation, ICRA 2000, April 24-28, 2000, San Francisco, CA, USA*, pp. 995–1001.

Laird, J. E. (2012). *The Soar Cognitive Architecture*, MIT Press (MA).

Laird, J. E., Kinkade, K. R., Mohan, S. and Xu, J. Z. (2012). Cognitive robotics using the soar cognitive architecture, *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Langley, P., Laird, J. E. and Rogers, S. (2009). Cognitive architectures: Research issues and challenges, *Cognitive Systems Research* **10**(2): 141–160.

LeDoux, J. (1998). *The emotional brain: The mysterious underpinnings of emotional life*, Simon and Schuster.

Levitt, J. B., Lewis, D. A., Yoshioka, T. and Lund, J. S. (1993). Topography of pyramidal neuron intrinsic connections in macaque monkey prefrontal cortex (areas 9 and 46), *The Journal of Comparative Neurology* **338**(3): 360–376.
**URL:** *http://dx.doi.org/10.1002/cne.903380304*

Lukasiewycz, M., Glaß, M., Reimann, F. and Teich, J. (2011). Opt4J - A Modular Framework for Meta-heuristic Optimization, *Proceedings of the Genetic and Evolutionary Computing Conference (GECCO 2011)*, Dublin, Ireland, pp. 1723–1730.

Luna, F., González-Álvarez, D. L., Chicano, F. and Vega-Rodríguez, M. A. (2014). The software project scheduling problem: A scalability analysis of multi-objective metaheuristics, *Applied Soft Computing Journal* **15**: 136–148.
**URL:** *http://dx.doi.org/10.1016/j.asoc.2013.10.015*

MacLean, P. D. (1990). *The triune brain in evolution: Role in paleocerebral functions*, Vol. 1, 1 edn, Plenum Press, 233 Spring Street, New York, N.Y. 10013.

Maes, P. (1989). How to do the right thing, *Connection Science* **1**(3): 291–323.

Mahadevan, S. and Connell, J. (1992). Automatic programming of behavior-based robots using reinforcement learning, *Artificial Intelligence* **55**(2–3): 311 – 365.
**URL:** *http://www.sciencedirect.com/science/article/pii/0004370292900586*

Majaranta, P. and Bulling, A. (2014). Eye tracking and eye-based human-computer interaction, *in* S. H. Fairclough and K. Gilleade (eds), *Advances in Physiological Computing*, Human-Computer Interaction Series, Springer London, pp. 39–65.
**URL:** *http://dx.doi.org/10.1007/978-1-4471-6392-3_3*

Metta, G., Sandini, G., Vernon, D., Natale, L. and Nori, F. (2008). The iCub humanoid robot: an open platform for research in embodied cognition, *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, ACM, pp. 50–56.

Miyake, A., Friedman, N. P., Emerson, M. J., Witzki, A. H., Howerter, A. and Wager, T. D. (2000). The unity and diversity of executive functions and their contributions to complex "Frontal Lobe" tasks: a latent variable analysis., *Cognitive psychology* **41**(1): 49–100.
**URL:** *http://www.ncbi.nlm.nih.gov/pubmed/10945922*

Monsell, S. (2003). Task switching, *Trends in Cognitive Sciences* **7**(3): 134 – 140.
**URL:** *http://www.sciencedirect.com/science/article/pii/S1364661303000287*

Moore, D. S., McCabe, G. P. and Craig, B. A. (2009). *Introduction to the Practice of Statistics*, 6 edn, W. H. Freeman and Company, New York, New York, USA.

Negatu, A. and Franklin, S. (2002). An action selection mechanism for "conscious" software agents, *Cognitive Science Quarterly* **2**: 363–386.

Ng, G. W., Tan, Y. S., Teow, L. N., Ng, K. H., Tan, K. H. and Chan, R. Z. (2011). A cognitive architecture for knowledge exploitation, *International Journal of Machine Consciousness* **3**(2): 237–253.

O'Reilly, R. C., Frank, M. J., Hazy, T. E. and Watz, B. (2007). Pvlv: the primary value and learned value pavlovian learning algorithm., *Behavioral neuroscience* **121**(1): 31.

O'Reilly, R. C., Hazy, T. E. and Herd, S. A. (2012). The leabra cognitive architecture: How to play 20 principles with nature and win!, *Manuscript* .

O'Reilly, R. C., Munakata, Y., Frank, M. J. and Hazy, T. E. (2012). *Computational Cognitive Neuroscience*, Wiki Book, 1st Edition, URL: http://ccnbook.colorado.edu.
**URL:** *http://ccnbook.colorado.edu*

Pérez-Ortiz, J. A., Gers, F. A., Eck, D. and Schmidhuber, J. (2003). Kalman filters improve lstm network performance in problems unsolvable by traditional recurrent nets, *Neural Networks* **16**(2): 241–250.

Poli, R. and Koza, J. (2014). Genetic programming, *in* E. K. Burke and G. Kendall (eds), *Search Methodologies*, Springer US, pp. 143–185.
**URL:** *http://dx.doi.org/10.1007/978-1-4614-6940-7_6*

Puigbo, J., Pumarola, A. and Tellez, R. (2013). Controlling a general purpose service robot by means of a cognitive architecture., *AIC@ AI* IA*, Citeseer, pp. 45–55.

Raizer, K. and Gudwin, R. R. (2014). A neuroscience inspired gated learning action selection mechanism, *International Journal of Biologically Inspired Cognitive Architectures* .
**URL:** *http://www.sciencedirect.com/science/article/pii/S2212683X14000796*

Raizer, K., Paraense, A. L. O. and Gudwin, R. R. (2011). A cognitive neuroscience-inspired codelet-based cognitive architecture for the control of artificial creatures with incremental levels of machine consciousness, *Proceedings of a symposium at the AISB11 Convention. Machine Consciousness 2011: Self, Integration and Explanation*, UK Society for the Study of Artificial Intelligence and Simulation of Behaviour, York, United Kingdom, pp. 43–50.
**URL:** *http://www.aisb.org.uk/publications/proceedings/aisb2011.zip*

Raizer, K., Paraense, A. L. O. and Gudwin, R. R. (2012). A cognitive architecture with incremental levels of machine consciousness inspired by cognitive neuroscience, *International Journal of Machine Consciousness* **04**(02): 335–352.
**URL:** *http://www.worldscientific.com/doi/abs/10.1142/S1793843012400197*

Raizer, K., Paraense, A. L. O. and Gudwin, R. R. (2014). A neuroscience inspired gated learning action selection mechanism, *International Conference on Biologically Inspired Cognitive Architectures*.
**URL:** *http://bicasociety.org/meetings/2014/abstracts/*

Raizer, K., Rohmer, E., Paraense, A. L. O. and Gudwin, R. R. (2013a). Effects of behavior network as a suggestion system to assist bci users, *Computational Intelligence in Rehabilitation and Assistive Technologies (CIRAT), 2013 IEEE Symposium on*, pp. 40–47.

Raizer, K., Rohmer, E., Paraense, A. L. O. and Gudwin, R. R. (2013b). Registered software ag as: Agente de software inteligente aplicado a tecnologia assistiva. Registration Office: INPI - Instituto Nacional da Propriedade Industrial. Instituição(ões) financiadora(s): CAPES; CNPQ; FAPESP.

Raizer, K., Rohmer, E., Paraense, A. L. O., Gudwin, R. R. and Cardozo, E. (2013). Pending patent: Método de sugestões baseado em uma rede de comportamentos para tecnologia assistiva. Number: BR10201302310; Registration Office: INPI - Instituto Nacional da Propriedade Industrial; Financing Institutions: CAPES; CNPQ; FAPESP.

Reggia, J. A. (2013). The rise of machine consciousness: Studying consciousness with computational models, *Neural Networks* **44**(0): 112 – 131.
**URL:** *http://www.sciencedirect.com/science/article/pii/S0893608013000968*

Riedmiller, M., Gabel, T., Hafner, R. and Lange, S. (2009). Reinforcement learning for robot soccer, *Autonomous Robots* **27**(1): 55–73.
**URL:** *http://dx.doi.org/10.1007/s10514-009-9120-4*

Rodrıguez, F., Galvan, F., Ramos, F., Castellanos, E., Garcıa, G. and Covarrubias, P. (2010). A Cognitive Architecture Based on Neuroscience for the Control of Virtual 3D Human Creatures, *Brain Informatics: International Conference, BI 2010, Toronto, Canada, August 28-30, 2010, Proceedings*, p. 328.

Rohmer, E., Reina, G. and Yoshida, K. (2010). Dynamic simulation-based action planner for a reconfigurable hybrid leg-wheel planetary exploration rover, *Advanced Robotics* **24**(8-9): 1219–1238.
**URL:** *http://www.tandfonline.com/doi/abs/10.1163/016918610X501499*

Russell, S. and Norvig, P. (2010). *Artificial Intelligence A Modern Approach*, 3 edn, Prentice Hall.
**URL:** *http://aima.cs.berkeley.edu/*

Samsonovich, A. (2013). Emotional biologically inspired cognitive architecture, *Biologically Inspired Cognitive Architectures* **6**(0): 109 – 125. {BICA} 2013: Papers from the Fourth Annual Meeting of the {BICA} Society.
**URL:** *http://www.sciencedirect.com/science/article/pii/S2212683X13000765*

Schmidhuber, J. (2015). Deep learning in neural networks: An overview, *Neural Networks* **61**(0): 85 – 117.
**URL:** *http://www.sciencedirect.com/science/article/pii/S0893608014002135*

Schwarz, M., Stückler, J. and Behnke, S. (2014). Mobile teleoperation interfaces with adjustable autonomy for personal service robots, *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction - HRI '14*, ACM, pp. 288–289.
**URL:** *http://dl.acm.org/citation.cfm?doid=2559636.2563716*

Seth, A. K., Baars, B. J. and Edelman, D. B. (2005). Criteria for consciousness in humans and other mammals, *Consciousness and Cognition* **14**(1): 119 – 139. Neurobiology of Animal Consciousness.
**URL:** *http://www.sciencedirect.com/science/article/pii/S1053810004000893*

Smith, C. (2010). The triune brain in antiquity: Plato, aristotle, erasistratus, *Journal of the History of the Neurosciences* **19**(1): 1–14.
**URL:** *http://dx.doi.org/10.1080/09647040802601605*

Smith, R. E., Dike, B. A., Ravichandran, B., El-Fallah, A. and Mehra, R. K. (2000). The fighter aircraft lcs: A case of different lcs goals and techniques, *in* P. Lanzi, W. Stolzmann and S. W. Wilson (eds), *Learning Classifier Systems*, Vol. 1813 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 283–300.
**URL:** *http://dx.doi.org/10.1007/3-540-45027-0-15*

Stewart, T. C., Choo, X. and Eliasmith, C. (2010). Dynamic behaviour of a spiking model of action selection in the basal ganglia, *Proceedings of the 10th International Conference on Cognitive Modeling*.
**URL:** *http://compneuro.uwaterloo.ca/files/publications/stewart.2010.pdf*

Stone, P., Sutton, R. S. and Kuhlmann, G. (2005). Reinforcement learning for robocup soccer keepaway, *Adaptive Behavior* **13**(3): 165–188.
**URL:** *http://adb.sagepub.com/content/13/3/165.abstract*

Sundermeyer, M., Ney, H. and Schluter, R. (2015). From feedforward to recurrent lstm neural networks for language modeling, *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* **23**(3): 517–529.

Tallon-Baudry, C. (2011). On the neural mechanisms subserving consciousness and attention., *Frontiers in psychology* **2**(January): 397.

Tononi, G. (2008). Consciousness as integrated information: A provisional manifesto., *The Biological Bulletin* **215**(3): 216–242.

Tyrrell, T. (1994). An evaluation of Maes's bottom-up mechanism for behavior selection, *Adaptive Behavior* **2**(4): 307–348.

Vernon, D., Metta, G. and Sandini, G. (2007). The iCub cognitive architecture: Interactive development in a humanoid robot, *2007 IEEE 6th International Conference on Development and Learning* pp. 122–127.

Wang, H. and Liu, X. P. (2014). Adaptive shared control for a novel mobile assistive robot, *Mechatronics, IEEE/ASME Transactions on* **19**(6): 1725–1736.

Wilson, S. W. (1995). Classifier fitness based on accuracy, *Evol. Comput.* **3**(2): 149–175.
**URL:** *http://dx.doi.org/10.1162/evco.1995.3.2.149*

Zang, Z., Li, D. and Wang, J. (2014). Learning classifier systems with memory condition to solve non-markov problems, *Soft Computing* pp. 1–21.
**URL:** *http://dx.doi.org/10.1007/s00500-014-1357-y*

Zhu, C., Tang, L. and Zhang, W. (2014). Multi-vehicle coordination and flexible scheduling based on simulated annealing algorithm, *Control and Decision Conference (2014 CCDC), The 26th Chinese*, pp. 2686–2691.