ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 8 N. 4 (2019), 83-96
eISSN: 2255-2863
DOI: http://dx.doi.org/10.14201/ADCAIJ2019848396

Towards a model-theoretic framework for describing the semantic aspects of cognitive processes

Sergio Miguel-Tomé[a]

[a] Grupo de Investigación en Minería de Datos (MiDa), Universidad de Salamanca, Salamanca, Spain
sergiom@usal.es

KEYWORD

Heuristics; cognitive architectures; model-theoretic semantics; episodic memory

ABSTRACT

Semantics is one of the most challenging aspects of cognitive architectures. Mathematical logic, or linguistics, highlights that semantics is essential to human cognition. The Cognitive Theory of True Conditions (CTTC) is a proposal to implement cognitive abilities and to describe the semantics of symbolic cognitive architectures based on model-theoretic semantics. This article focuses on the concepts supporting the mathematical formulation of the CTTC, its relationship to other proposals, and how it can be used as a framework for designing cognitive abilities in agents.

1. Introduction

Artificial intelligence (AI) attempts to reproduce the cognitive abilities of humans, and cognitive psychology (CP) explains these abilities. This complementarity is exhibited by the development and implementation of computer models, called cognitive architectures, which explain the structure and functioning of the human mind. Cognitive architectures have emerged from the programme proposed by Allan Newelll to build unified theories of cognition (Newell, 1990; Newell, 1973). Currently, there are many cognitive architectures, and each has its own features (Kotseruba and Tsotsos, 2020; Samsonovich, 2010; Langley *et al*., 2009). One classification method divides the different cognitive architectures into the following groups: symbolic, connectionist, and hybrid. However, designers of symbolic and connectionist architectures disagree on classification methods because the former assume computational functionalism and the latter generally reject it. Until now, these discussions have been in the philosophical field. From the computational point of view, the difference between these two approaches lies in the fact that the specification and resolution of some computational tasks are easier in one approach than in the other. For that reason, the point of view assumed in this paper is that they can be understood as different levels of description, and they can be translated to another level (Mira-Mira and García-Delgado, 2007).

Since Newell's proposal, significant research has been done to develop cognitive architectures. However, while many steps have been taken to understand the syntactic aspects of cognition, few have been taken to understand semantics in cognition (Targon, 2016; Wang, 2009; Wang, 2005). Several disciplines, such as mathematical logic and linguistics, have highlighted the need to incorporate semantics because of the limitations of the syntactic approach. Therefore, semantic aspects should be researched and incorporated into the research of cognitive architectures. Specifically, developing computational grounded cognition has been proposed as

an objective. (Pezzulo *et al*., 2013). With regard to the study of natural language, Donald Davidson proposed the use of true conditions as a mechanism used by the mind to generate meanings (Davidson, 1967; Davidson, 2005).

This article is focused on presenting a proposal to research these semantic aspects, called the Cognitive Theory of True Conditions (CTTC) (Miguel-Tomé, 2006). The essence of the CTTC is that true conditions are not only a tool for giving meaning to language expression, but also play a fundamental role in all cognitive features that the human brain possesses. The CTTC can play two roles: 1) as a framework for studying how model-theoretic semantics is involved in cognitive architectures at a symbolic level and 2) as a theory for designing agents with cognitive abilities. In this paper, we address the CTTC as a theory to provide agents with cognitive abilities. We also highlight the concepts underlying the CTTC and the mathematical notation.

The structure is described here. The second section briefly reviews the main elements of the CTTC: true conditions, the PPFMM structure and the hierarchy of languages. The third section deals with how the CTTC addresses describing decision-making processes and designing mechanisms to make decisions. The fourth section shows how the CTTC has been used to implement episodic memory in agents. The fifth section discusses the differences between the CTTC and other symbolic approaches that already exist in AI. The last section summarizes the main issues reviewed and discusses future research plans.

2. The Cognitive Theory of True Conditions

In the early 20th century, David Hilbert proposed doing away with the notion of truth to create a foundation for mathematics. His programme was based only in syntactic notions, but Kurt Gödel showed that this was not powerful enough. Alfred Tarski reintroduced the notion of truth in mathematics when given a formal definition of truth for formal languages (Manzano, 1999). From this definition of truth emerged a new branch of mathematics, called model theory, which studies how formal languages describe classes of mathematical structures. In model theory, a formal language and a mathematical structure are connected by a type of function, called interpretation, which determines what part of the structure is designed by a formula. Curiously, although Tarski was against the notion of truth in natural language, his definition was the key to Davidson's proposal explaining meaning in natural language. He proposed that a finite theory of meaning can be given for a natural language through true conditions (Davidson, 1967). The true conditions are formulated using the T-scheme of Tarski, e.g., "snow is white" is true if and only if snow is white. "Snow is white" is a sentence, and snow is white is a physical fact. The T-scheme combines these two kinds of objects, creating a true condition to the sentence. Davidson argued that because humans learn their languages empirically, natural language must be stateable in a finite form, even if it is capable of a theoretically infinite number of expressions. Thus, the meanings of an infinite number of sentences should be based on a finite system of axioms.

The concept of model-theoretic semantics is also important in CP. The seminal work of Philip Johnson-Laird (Johnson-Laird, 1980) gave rise to the field now known as mental models. Currently, grounded cognition is a major topic in CP (Barsalou, 2010; Barsalou, 2008). This is because little empirical evidence supports the existence of amodal symbols, cognitive traditional theories fail to address how cognition interfaces with perception and action, and it is unknown how the brain handles amodal symbols with neural computation (Barsalou, 2008). Despite of the importance model-theoretic semantics in the mentioned fields, it has not been replicated in the research of cognitive architectures. There has been little scientific research at the symbolic level of how model-theoretic semantics is relevant or can help in developing autonomous agents or robots. This has led to the proposal of computational grounded cognition (Pezzulo *et al*., 2013). The CTTC is a mathematical framework to deal with this gap. Because the mathematical formulation of the CTTC is too extensive to provide here, this article includes only a gentle mathematical introduction with graphical representations to avoid giving long mathematical definitions. For the formal definitions, the reader is referred to another paper (Miguel-Tomé, 2018b).

The main idea behind the CTTC is that the perceptual space is a set of formal languages that denote elements of a model embedded in a quotient space of the physical space. In other words, the CTTC assumes that the perceptual space is a language, and the relation between the perceptual space and the physical space has

two features. One is the omission of elements of the physical space within the perceptual space (e.g., humans cannot see infrared and ultraviolet radiation). The other feature is that different physical states result in the same perceptual state in the perceptual process. (e.g., humans cannot perceive atomic variations when perceiving a macroscopic object). It is important to note that the CTTC does not propose that the mathematical structure connected to the languages by the true conditions is isomorphic to the physical universe. Nature has not led to the emergence of the human brain architecture because it describes completely the physical structure of the universe. Human brain architecture has emerged because there is a differential reproduction in human niches, and the human brain architecture allows longer and more frequent survival than other brain architectures. Also, evolution punishes brain architectures that waste energy and time describing elements and features of the physical space that are not relevant to the differential reproduction.

The CTTC uses the following methodology to explain cognitive abilities:

- Determines a class of mathematical structure from the perceptual human space.
- Defines formal languages to describe the structures of the previous class of mathematical structures.
- Defines algorithms that manipulate the previous formal languages to explain cognitive abilities.

It must be noted that the CTTC relies on the proposal that we can obtain the class of the mathematical structures the brain is trying to describe by studying the perceptual human space. One could be surprised by the proposal that a class of mathematical structures can be determined from the perceptual space, because the CTTC considers the perceptual space to be a language to describe the physical space. However, in model theory, one classical technique for building a mathematical structure that can be a model for a set of formulas is employing atomic formulas.

According to the described approach, the CTTC proposes to identify mental representations with formulas of a hierarchy of formal languages that describe a specific class of mathematical structures model whose signature is the same as the perceptual human space. The CTTC also proposes that an agent can be endowed with cognitive abilities by implementing algorithms that can handle the formulas of the hierarchy of formal languages for the class of mathematical structures that the brain is trying to describe. Therefore, the first element of the CTTC is a class of mathematical structures, and the second one is a set of formal languages that must allow for describing the previous class of mathematical structures. The meaning of *describing* is that there must be true conditions that, by means of denoting elements of the structure, determine whether a well-formed formula of the language is true or false in the structure. The true conditions connect the physical space with the perceptual space.

2.1. Multi-optional Many-sorted Past Present Future structures

As was mentioned above, the first element of the CTTC is a class of mathematical structures. The CTTC determines that the signature of the class of structures is generated from the perceptual human space. The signature is a list of the kind of functions and relations that the structure has. Therefore, the CTTC states that the relationship between the physical and perceptual spaces is the same that exists between models and languages in model theory. To clarify, although the current section is discussing structures based on the physical space, the previous section discussed perceptual space because there has been an evolutive process that has conformed to the perceptual space. Among other features, the structure must model the perceptual arrow of time. Currently, the CTTC proposes a class of structures, called *Multi-optional Many-sorted Past Present Future* (MMPPF), as the class of models to be described (Miguel-Tomé, 2006). This class was obtained from the subjective perceptual human space. An MMPPF structure is a nested structure of possible worlds. In other words, each world of the structure also contains another structure of possible worlds. Thus, they are more complex than the classical structures of possible worlds used in temporal logics. The Kripke structures used to interpret temporal logics do not allow a flow of time, since the temporal logics are designed for reasoning in a static temporal view that is relative to a temporal point, the present. The mathematical structures of temporal logics are like a temporal picture of the universe. However, this is not sufficient to implement cognitive abilities because many are connected to how the mind deals with how the environment changes with the flow of time. If there is no mathematical

structure that describes the flow of time, there cannot be languages with a model-theoretic semantics that address the perceptual flow of time. For example, complex learning involves comparing an expected future with what actually occurs. Therefore, complex learning is impossible without considering the flow of time. Now that the necessity of a complex class of structures such as the MMPPF has been explained, the types of elements and their relations that the structure contains will be explained.

An MMPPF structure has worlds, named *temporal perspectives*. The relation between temporal perspectives is a strict total order. A temporal perspective is also a structure of possible worlds. The worlds that have a temporal perspective are called *moments of time*. The number of temporal perspectives is equal to the number of moments of time. Each moment of time is a set that contains elements that are tuples. Each of these tuples is called *reality*. Each reality is a triple, and two positions of the triple determine two different features: a *temporal location* and a *condition of reality*. The temporal location has three possible values: *past, present*, and *future*. All the realities of a moment of time have the same value of temporal location. The condition of reality has two possible values: *existing* and *hypothetical*. These two values exist because humans have the ability to imagine hypothetical episodes that could have turned out differently in the past (Jacques *et al*., 2018). Each temporal perspective has only one moment of time whose realities are present and existing. An important point of the MMPPF structures is that when one passes from one temporal perspective to the next, the features of the worlds change consistently with the psychological arrow of time, e.g., the moment of time that is present in a temporal perspective changes to the past in the next temporal perspective. In a temporal perspective, one of the moments of time that is future changes to present in the next temporal perspective.

Alongside the relation between temporal perspectives and moments of time, there is a binary relation between realities. Figure 1 shows a graphical example of the aforementioned concepts and relations of an MMPPF structure. It is important to note that although we use the same notation, moments of time of different temporal perspectives are different objects. We should use an index denoting the temporal perspective to which the moment of time belongs. However, the reality notation is general to all temporal perspectives. If in any two different temporal perspectives there is the same name for a reality in each temporal perspective, they reference the same reality.

It was mentioned above that a reality is a triple, yet only two elements were explained. The third element of the triple consists of a many-sorted structure, which has sorts of three kinds. The sorts of the first kind are those that contain identifiers for the objects or theirs parts. The many-sorted structure has two sorts of this first kind: the sort for the identifiers of objects, $\mathcal{O}$, and the sort for the identifiers of parts of the objects, $\mathbf{H}$. $\mathbf{H}$ is called the *essence set*. These sorts are the same in each reality of an MMPPF structure.

The sorts of the second kind are the properties that an object can have, and the elements of each of these sorts are the values that each of the parts of an object can possess. Thus, each of these sorts is called *property*, and denoted by $\boldsymbol{V}_p$. Each $\boldsymbol{V}_p$ can be not only a set but also a mathematical structure, i.e., a vector space and it has dimensions. The existence of dimensions in a $\boldsymbol{V}_p$ is important because we need to consider changes in the dimensions, as we will see below, in addition to changes of the property.

The sorts of the last kind contain functions, called *actions*. Each action represents how an object can alter the values of one of its properties. For each property and each object, there is a sort that contains actions. Each object has an assigned set of actions.

Together with sorts, each many-sorted structure has functions and relations. The functions connect the elements of the different sorts. There is a function that assigns parts to each object, and another for each property that assigns values to each part. Also, there is a function that assigns actions to each object. Each object always has an action assigned for each property, which does not alter the value of the property. These actions are named *neutral actions*. If the set of actions assigned to an object has more actions than the neutral actions, the object is named an *agent*.

Regarding the relations of the many-sorted structure, they determine when one object influences the values of another in a specific property. This means that if an object $\boldsymbol{o}_i$ is related to another object $\boldsymbol{o}_i$ in the property p, then an action of the former in the property p affects the values of the corresponding property of the second object. That is, when one grabs a bottle with her hand, the bottle also moves if she moves her hand. These relations are called *momentary superset relations*.

It was mentioned above that there is a relation between realities. This relation is directly connected to the actions associated with each object, and it connects a reality of the moment of time ***t*** with a reality of another moment of time ***t′*** if the objects of ***t*** have actions that change the values in ***t*** to what they have in ***t′***. The actions that connect two realities are represented with a tuple of actions called *interaction*. An interaction contains one action for each object and feature. The realities can be considered nodes of a directed graph, and the edges are determined by the interactions. The interactions are always from a reality of a moment of time ***t*** to a reality of the next moment of time to ***t***. These concepts are represented graphically in Figure 2.

When considering a temporal perspective as a graph where the realities are its nodes, a path where each node belongs to a different moment is called a *temporal path*. The CTTC considers them a key to learning and reasoning.

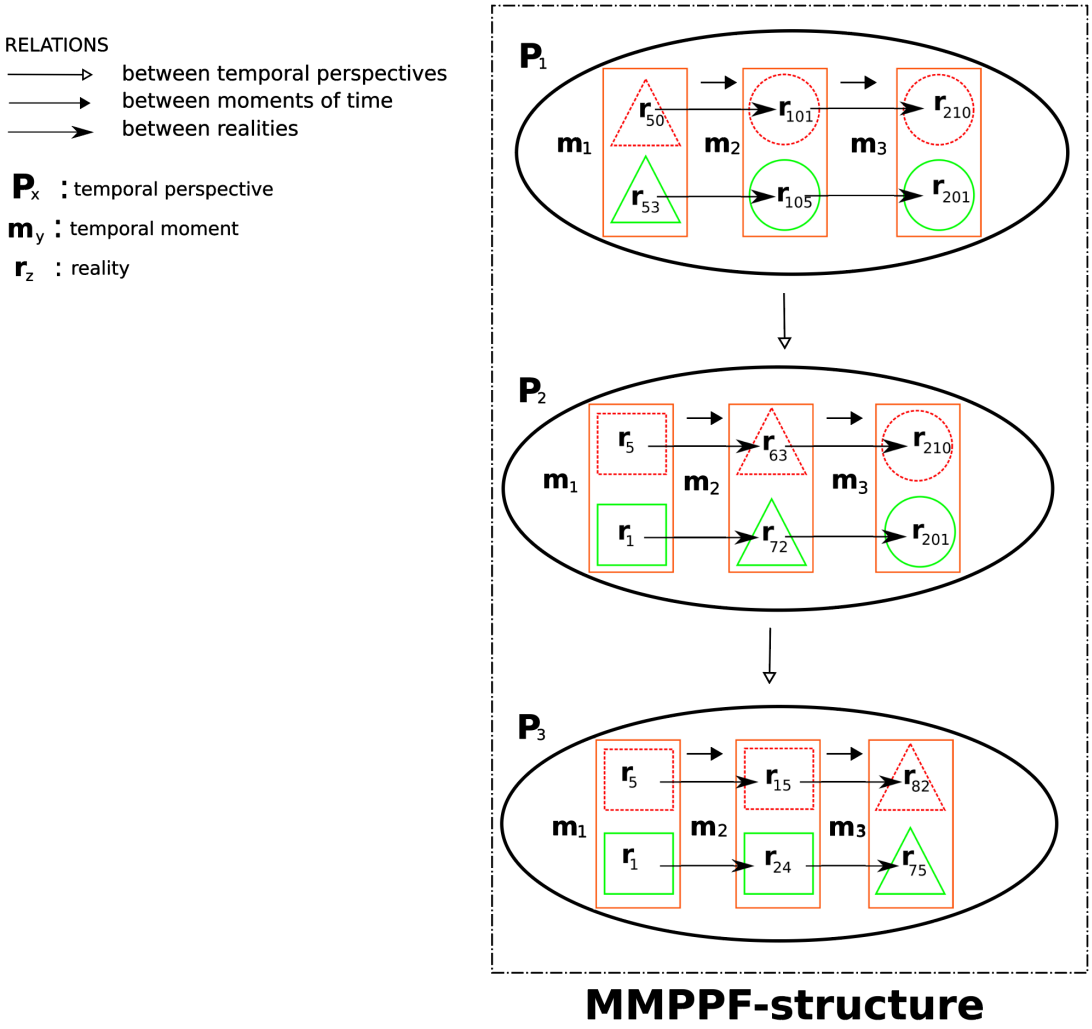


Figure 1: Graphical representation of an MMPPF structure. The MMPFF structure shown here has three temporal perspectives: P_1, P_2, P_3. Each temporal perspective has three temporal moments: m_1, m_2, m_3. Each temporal moment has two realities. The ellipses are temporal perspectives. The rectangles are temporal moments. Square = Past reality, Triangle = Present reality, Circle = Future reality. Green/Continued Line = Existing reality, Red/Discontinued Line = Hypothetical reality.

The binary relation between realities can be described as a function. That function assigns a reality, named *producing function*, and denoted by &, to each reality and interaction.

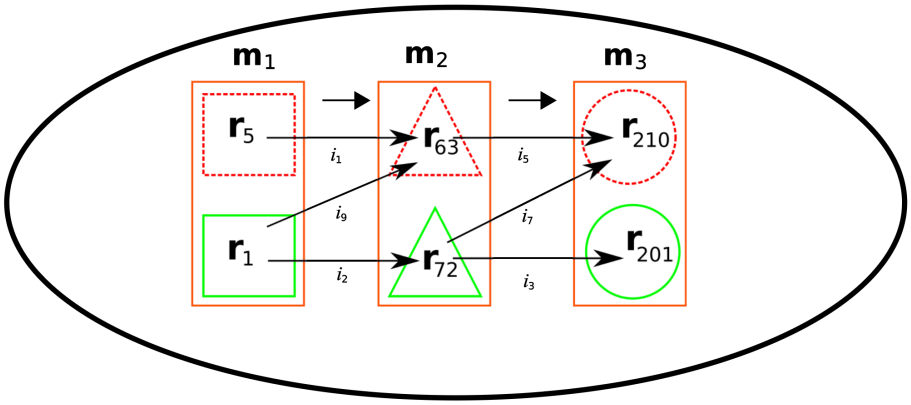


Figure 2: Graphical representation of a temporal perspective. The temporal perspective shown has three temporal moments, each of which has two realities. The rectangles are temporal moments. Square = Past reality, Triangle = Actual reality, Circle = Future reality. Green/Continued Line = Existing reality, Red/Discontinued Line = Hypothetical reality.

$$\begin{array}{rccc} \&: & \boldsymbol{R} \times \boldsymbol{I} & \rightarrow & \boldsymbol{R} \\ & (\boldsymbol{r}, \vec{\boldsymbol{i}}) & \mapsto & \boldsymbol{r} \end{array}$$

Another important function that can be drawn from an MMPPF structure is the interaction function, denoted by ⓘ. It returns the interaction that happens in each moment of time. It is defined as follows:

$$\begin{array}{rccc} \textbf{ⓘ}: & \boldsymbol{T} & \rightarrow & \boldsymbol{I} \\ & \boldsymbol{t} & \mapsto & \boldsymbol{i} \end{array}$$

where T is an index set isomorphic to the set of moments of time.

Usually, we are interested in the actions an object contributes to the interaction, such as when there is only one agent among all the objects of the environment. To denote the specific actions of one object, we add a sub-index to indicate the agent, ⓘ_u.

Using the functions & and ⓘ, a function called *successor function* is built, and it is defined as:

$$\begin{array}{rccc} \boldsymbol{Succ}: & (\boldsymbol{T} \times \boldsymbol{R}) \times \boldsymbol{I} & \rightarrow & \boldsymbol{T} \times \boldsymbol{R} \\ & [(\boldsymbol{t}, \boldsymbol{r}_x), \boldsymbol{i}] & \mapsto & (\boldsymbol{t} + 1, \boldsymbol{r}_y) \end{array}$$

Although we have obtained the functions & and ⓘ from the MMPPF structures, we could go in the other direction and define an MMPPF structure from these functions.

The last element about the MMPPF structure to discuss is a specific sort that contains values that reference reward and aversion states. This sort, named *system of rewards and aversions*, is denoted by ***SRA***. The elements of the sort ***SRA*** are a tuple made of pairs.

$$\boldsymbol{SRA} = \{[(\boldsymbol{s}_1, \boldsymbol{c}_1), \ldots, (\boldsymbol{s}_m, \boldsymbol{c}_m)], \ldots, [(\boldsymbol{c}_1, \boldsymbol{c}_1), \ldots, (\boldsymbol{c}_m, \boldsymbol{c}_m)]\} \cup \{\emptyset\}$$

Each pair determines the level of reward for a sensation, e.g., hunger and thirst. The first element of the pair determines the sensation of reward or aversion, and the second element is the maximum sensation of reward. The elements are ordered so that the farther the distance is from the first element of the pair to the second, the greater the aversion to the situation. Reward and aversion do not exist in the external physical world. Although one could think that reward and aversion should not be included in the MMPPF structure definition because

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 8 N. 4 (2019), 83-96
eISSN: 2255-2863 - http://adcaij.usal.es
Ediciones Universidad de Salamanca - CC BY NC DC

they do not exist in the external physical world, they exist in the brain through neuronal signals that organisms integrate into their representations of reality (Sescousse *et al*., 2013) and therefore are part of the model that organisms describe with perceptual space.

Primary rewards are fundamental to driving behavior in animals. The MMPPF structure captures that fact with the function $\odot$, which assigns a state of reward and aversion to each object. It is defined in the following way:

$$\begin{array}{rccc} \odot: & \mathcal{O} & \rightarrow & \boldsymbol{SRA} \\ & o & \mapsto & \vec{s} \end{array}$$

If $\vec{s} \neq \emptyset$ the object is denominated *subject*.

It is interesting to note that if function $\odot$ contradicts the situations that allow an agent to survive, it leads the agent to its death. Therefore, natural selection causes the correct definition of function $\odot$ in nature.

2.2. Describing the MMPPF structure: The hierarchy of languages

In model theory, a formal language is defined to describe a given mathematical structure. This mathematical approach of model theory is used by the CTTC. It proposes that the physical space is described by the perceptual space. The MMPPF structure definition provided in the previous subsection was obtained from the subjective perceptual space. The CTTC defines formal languages to describe the MMPPF structure (Miguel-Tomé, 2018b; Miguel-Tomé, 2006); these formal languages serve as the basis for an agent to possess cognitive abilities. The CTTC proposes an agent use a hierarchy of three formal languages to describe an MMPPF structure. Each language is built from the previous one in the hierarchy. The language at the bottom only quantifies, and the successive languages qualify different elements that were quantified in the previous one. This kind of language building is based on neuroscientific facts. In a nervous system, an organism registers all information through receptor cells, which are each characterized by a modality to which it is particularly sensitive and responsive. They register the value of a physical phenomenon in that specific moment, which involves generating any other information that the nervous system has from that information registered by the organism. Thus, the idea of the hierarchy of languages is a replication of the process of generating information to describe the environment at the symbolic level made by nervous systems.

The formal languages of the hierarchy are named as follows: perceptive language, extended perceptive language and categorical language. They are denoted PL_{MMPPF}, PL^{*}_{MMPPF} and CL_{MMPPF} respectively.

All the formulas of these languages only reference one temporal perspective of the MMPPF structure. PL_{MMPPF} is at the base of the hierarchy. It has a constant for each element of each sort of the MMPPF structure but it does not have any variable. Because PL_{MMPPF} has neither quantifiers nor variables, it describes the MMPPF structure exactly and cannot express any abstract property about the structure or its elements.

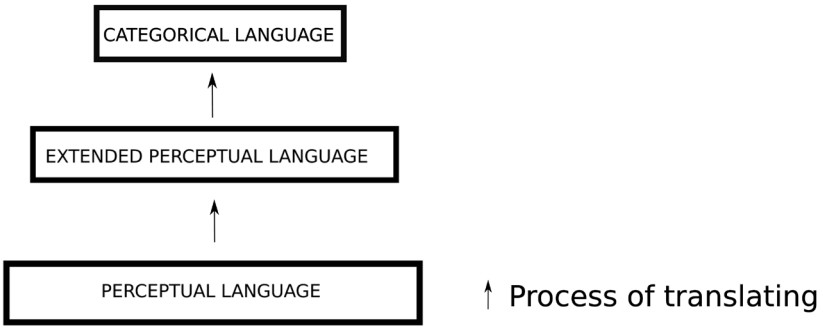


Figure 3: Graphical representation of the hierarchy of languages.

An agent can generate a well-formed formula of PL_{MMPPF} that describes the structure directly by using the information registered by its sensors and concatenating and processing the atomic formulas continuously. The

formula of PL_{MMPPF} that the agent generates using from the information from its sensors is denominated *log formula*. From the log formula, the agent can generate formulas of PL^{*}_{MMPPF} because atomic formulas of PL^{*}_{MMPPF} describe qualitative states about the same object from consecutive moments of time. The agent only needs a process to substitute the value of a property with an element from a metainformation alphabet using the true conditions. However, building a formula of CL_{MMPPF} from PL^{*}_{MMPPF} is more complex because it requires temporal qualification. The CTTC uses a mechanism denominated *recognizer grammar of true conditions* (RGTC) to achieve it (Miguel-Tomé, 2018b). The CTTC proposes.

It should be noted that how an agent builds its log formula is an issue closely related to how the agent determines if a formula is true or false. Given a formula of CL_{MMPPF}, an agent must check if the CL_{MMPPF} formula can be generated from its log formula to determine if it is true. This is because a wff of CL_{MMPPF}, ϕ, is true in the structure if a wff of PL_{MMPPF}, φ, is true in the structure and ϕ is the translation of φ.

3. Describing decision-making processes in the CTTC

Until now, we have shown how the CTTC determines how an environment can be described by an agent. However, we have not discussed how decision-making processes are described in the CTTC. The first issue is that the CTTC establishes a relation that connects the MMPPF structures with the hierarchy of languages. The general equation is the following:

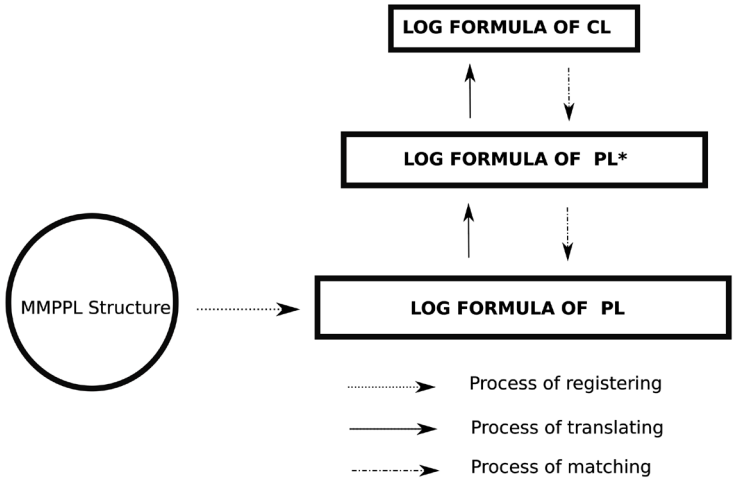


Figure 4: Processes proposed by the CTTC as the basis for implementing cognitive abilities.

$$ⓘ_u(\boldsymbol{t}) = I^r_p(\pi^1(\pi^1(F_u([\phi_1, ..., \phi_n]_t, [\psi_1, ..., \psi_{n'}]_{t-1})))) \quad (1)$$

where, F_u is a function that generates a pair of elements (the first element is a sequence of constants that denote interactions of the agent u, and the second is a tuple of formulas of the formal languages of the hierarchy), $[\phi_1, ..., \phi_n]t$ is a tuple of formulas of the formal languages of the hierarchy that is input, $[\psi_1, ..., \psi_{n'}]_{t-1}$ is a tuple of formulas of the formal languages of the hierarchy that was generated by the system, π^n is the projection function that projects the n-element of a tuple, and $\boldsymbol{r}_p = (\boldsymbol{t}, \square, ||, \boldsymbol{e}_x)$ is the existing reality of the moment of time $\boldsymbol{t}$, I^r_p an interpretation function, and $ⓘ_u$ is the interaction function of the agent u.

The relation formulated can be seen as a functional equation if F_u is considered an unknown variable. The problem is that we do not know a method to determine F_u. From Equation 1 different particular equations can be derived. For more details, we refer to the reader to (Miguel-Tomé, 2018a).

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 8 N. 4 (2019), 83-96
eISSN: 2255-2863 - http://adcaij.usal.es
Ediciones Universidad de Salamanca - CC BY NC DC

4. Implementing episodic visual memory in agents

Episodic cognition is an important skill that comprises episodic memory and foresight. This section explains how the CTTC has been used as a framework for implementing episodic memory in agents and mechanisms of attention. This research is aimed at providing agents of a microworld with episodic visual memory through the hierarchy of languages of the CTTC. Specifically, the target consists in an agent having the ability to remember whether it sees other specific agents to assist its navigation through the microworld. The starting point was a microworld and an architecture of agents, the Topological Qualitative Navigation Architecture (TQNA) 1.0., designed and implemented in previous research (Miguel-Tomé, 2018c). The research has extended the TQNA 1.0. with data structures to store and retrieve visual information.

Using the CTTC as a framework, episodic memory can be identified with the log formula. By defining new data types and classes, the log formula has been implemented. New data types define the following: the condition of reality, temporal location values, atomic formulas of PL_{MMPPF} and connectors, respectively. The two defined classes are to define a moment of time and the log formula. The class defined to implement the log formula allows storing registers of two fields. One field is an object of the class defined to implement the moment of time. The other field is of the connectors type, which determines which connector is between the moment of that register and the next register. The implementations in Object Pascal are in Appendix A.

Using the types and structures defined, the algorithm to store information to create episodic memory implemented in the agents is Algorithm 1.

Algorithm 1 Episodic memory. The loop builds a moment formula with all current information registered by the agent.

Require: n; a counter variable

formulaL0; This variable stores the information of an atomic formula of PL_{MMPPF} actualMoment; This object stores all atomic formulas with the current registered information logFL0; This object

Ensure: Stores all visual information registered by the agent.

1: momentoActual:= CMomentL0.create; // It creates an instance of the class that implements a moment of time

2: IF Lobjvistos.count > 0)THEN; // It checks that the visual space is not empty FOR n:= 0 TO Lobjvistos. count-1 DO // It builds the moment

BEGIN

actualMoment.insert(Lobjvistos.item(n),e,present); // It builds an atomic formula and adds it to the moment END

logFL0.insertMoment(actualMoment); // It builds the log formula adding the actual moment

One issue to take into account is that when the moment of time created is added to the log formula, the last moment stored must change the temporal location of the atomic formulas of which it is composed of the past.

Regarding determining if a specific agent is seen or not, a method is defined that uses a *for* loop that iterates throughout the whole array and checks if there is any moment that has an atomic formula about the specific agent. This method allows an agent to retrieve a memory. During the implementation, checking the moments is done in a sequential manner, but it could also be performed in a parallel manner.

The previous implementation does not consider a limit in the computing resources, but we know that the human brain has a limit of the number of objects it can pay attention to (Miller, 1956). Therefore, an option has also been added that simulates a process of attention that affects episodic memory. The process limits the number of objects that the log formula stores in each moment. Thus, the process selects at most the seven nearest objects from the agent to store them. The rest of the objects registered in the agent's visual field are ignored in the process of episodic memory.

The software development used to write the implementation was Delphi XE7. Figure 5, Figure (a), and Figure (b) show screenshots of the interface of the program created.

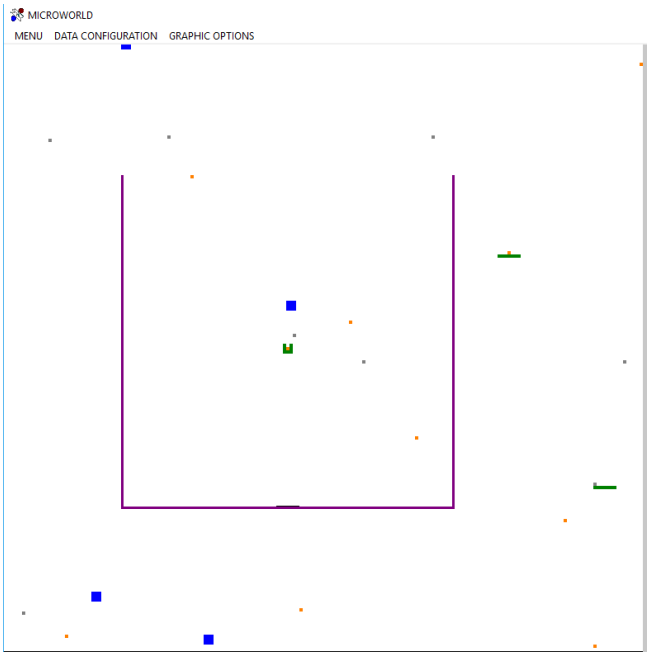
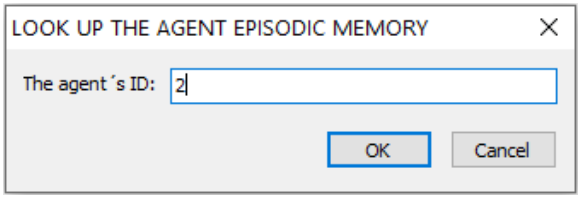


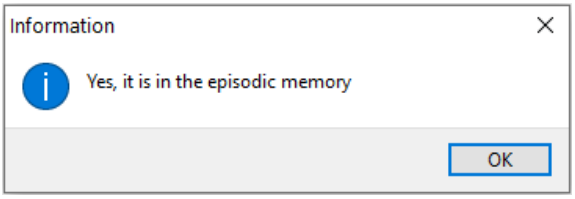
Figure 5: This is a screenshot of the program while the agent is navigating through the environment. The purple lines determine the limits of the agent's visual field. The agent with episodic memory is black. The agents without episodic memory are green. Obstacles are blue. Edible objects are yellow. Inedible objects are grey.

5. Discussion

The logical approach is one of the oldest in AI. It was proposed by McCarthy in 1958 in the Symposium on Mechanization of Thought Processes at the National Physical Laboratory in Teddington. In AI, there are several symbolic formalisms, such as situation calculus (Brachman and Levesque, 2004), event calculus (Mueller, 2015) or fluent calculus (Thielscher, 2005). The last two resolve the Frame Problem. To see the difference between the CTTC and other proposals, we must know the difference between the approaches in logic. In mathematical logic, there are two approaches: proof theory and model theory. Proof theory is syntactic in nature, whereas



(a) The user can write the ID of an agent to know whether the agent remembers it saw it.



(b) The program answers whether the ID of the agent is, or is not, in the episodic memory.

Figure 6: Screenshots of the interface to retrieve information from the episodic memory of the agent.

model theory is semantic in nature. Proof theory is focused on rules, and the validity of the rules is proven with a proof that does not produce formulas that disagree with the axioms. In model theory, on the other hand, the rules are valid if they are semantically correct in models. Gödel proved that a syntactic approach cannot provide a foundation for mathematics (Gödel, 1931) but deductive systems of first order logic are consistent and

semantically complete. In terms of computation, all that is effectively calculable can be done with first order logic. This is why there are general-purpose logic programming languages.

Situation calculus, event calculus, and fluent calculus are formalisms that use the proof theory approach. In those formalisms, deductive systems are defined using sets of axioms, and plans of actions are generated using a syntactic approach. They apply resolution to achieve the goal. Resolution is a rule where semantics does not play any role. Iteratively applying resolution allows for knowing whether a propositional formula is satisfiable and for proving that a first-order formula is unsatisfiable. Thus, these formalisms only consider syntactic aspects. However, the CTTC involves the use of semantics because the mechanism for making decisions is based on a heuristic of qualitative semantic, and its rules for calculation are always contrasted with the MMPPF structure.

This leads to the fact that situation calculus, event calculus, and fluent calculus need external mechanisms to solve the symbol grounding problem that is not contemplated in their own formalisms. The predicates describe the properties and states of the objects, but the previous formalism never addresses how a robot or an agent can create the predicates from sensors to describe its environment. However, the CTTC contains its own mechanism, the hierarchy of languages, to address the symbol grounding problem, e.g., from the position of an object, the CTTC's hierarchy of languages determines that the object is moving or unmoving.

There are many proposals to describe how the human mind works in the field of cognitive architectures (Langley *et al*., 2009; Samsonovich, 2010), but what makes the CTTC attractive is that a framework, rather than a cognitive architecture, tries to incorporate model-theoretic semantics. This has not received much attention from AI researchers at the symbolic level, even though it is highlighted as an important topic in CP. To the author's best knowledge, the CTTC attempts to make a formal definition of a mathematical structure with realistic complexity to describe the perceptual human space. This opens a new line of research for describing semantic aspects of cognition through model-theoretic semantics. Previously, model-theoretic semantics had been criticized because truth values cannot be threatened by the acquisition of new knowledge (Wang, 2005). However, this is not a problem for the model-theoretic approach; the problem is caused by simple mathematical structures that do not contemplate the passing of time in a complex way. The realistic complexity of MMPPF structures avoids that problem. Also, model-theoretic semantics has been criticized, stating that "the model tells the truth value of statements to us, but not to the system"(Wang, 2005, p.286). The CTTC avoids that problem on the basis of the physical space is a model and the perception space and thought are languages that describe it. Thus, the CTTC proposes that sensors generate basic formulas. The agents build the log formulas combining the basic formulas, and building on the log formula, the hierarchy of languages explains how an agent can generate formulas that give more abstract descriptions of the environment. Therefore, an agent can know the truth value of a statement, but it needs sensors and to carry out parsing and translating processes. Obviously, implementing the CTTC requires much more work than other AI techniques, but it provides solutions for some problematic aspects of semantic and cognition.

6. Conclusions and future work

This article introduces the main concepts of the CTTC (Miguel-Tomé, 2006) and provides an example of how it can be used as a framework for designing cognitive abilities for agents. The article began by explaining the ambit of the CTTC and its two underlying principles: its model-theoretic approach and the need for an agent to have mechanisms that generate qualitative descriptions from the quantitative information it registers from sensors. After introducing the CTTC, we addressed how these principles are developed in the theory. The model-theoretic approach requires a class of mathematical structures able to describe the psychological arrow of time. To fulfill that requirement, MMPPF structures are employed as the model. Thus, we provided an overview of the MMPPF structures' conceptual elements, with an emphasis on how they describe the flow of time, and then addressed the formal languages used for descriptions in the CTTC for descriptions in the CTTC. The formulas for those formal languages form an agent's mental representations, so we explained how an agent generates those formulas. Once these theoretical concepts were presented, we addressed how to create agents with episodic memory, assuming the two principles underlying the CTTC. Because episodic memory can be

interpreted as the log formula contemplated in the CTTC, new code was implemented, from a previous implementation of agents able to navigate in dynamic and unknown environments, that allows for creating a log formula in an agent. Also, we added a simple interface to check whether an object is registered for the agent's navigation. The interface calls a function that checks whether there is any subformula in the log formula relative to the object. In this way, the implementation shows how the CTTC can be a mathematical framework to describe cognitive processes.

There are many possible lines of future research to expand the CTTC, but regardless if elements are modified or new elements incorporated, all must agree with the two principles that underlie the CTTC. In addition, the implementation of episodic memory in agents was discussed in this paper. The implementation performed serves as an example of how cognitive abilities and processes can be interpreted in the CTTC.

At this moment, the mathematical formulation of the CTTC is neither closed nor definitive. There are many issues that need to be developed and others have not even begun to be researched. For example, the hierarchy of languages is only about an internal description of the environment, but an external description has not been mentioned in this paper, and it has only begun to be addressed (Miguel-Tomé, 2017). An external description of the environment refers to one that can be exchanged among cognitive systems. That line of research can help create systems that generate and process natural language. One example of an issue that has not been considered yet in CTTC is learning. However, learning and value systems (Merrick, 2017) can be researched because the MMPPF structure contains states of rewards and aversions. Another issue is that $PL_{MMP\,P\,F}$, $PL^{*}_{MMP\,P\,F}$, and $CL_{MMP\,P\,F}$ only reference a temporal perspectives, but humans can consider knowledge from different times in complex reasoning. Thus, a future line of research should be a formal language to describe different temporal perspectives.

To conclude, the CTTC shows that understanding cognitive abilities and addressing the challenges of computational grounded cognition from a model-theoretic point of view is a legitimate line of research.

7. References

Barsalou, L., 2008. Grounded Cognition. *Annual Review of Psychology*, 59(1):617–645.

Barsalou, L., 2010. Grounded Cognition: Past, Present, and Future. *Topics in Cognitive Science*, 2(4):716–724. Brachman, R. and Levesque, H., 2004. *Knowledge Representation and Reasoning*. Morgan Kaufmann.

Davidson, D., 1967. Truth and Meaning. *Synthese*, 17(3):304–323. Davidson, D., 2005. *Truth and Predication*. Harvard University Press.

Gödel, K., 1931. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik*, 38(1):173–198.

Jacques, P. *et al*., 2018. Remembering and imagining alternative versions of the personal past. *Neuropsychologia*, 110:170–179.

Johnson-Laird, P., 1980. Mental Models in Cognitive Science. *Cognitive Science*, 4(1):71–115.

Kotseruba, I. and Tsotsos, J., 2020. 40 years of cognitive architectures: core cognitive abilities and practical applications. *Artificial Intelligence Review*, (53):17–94.

Langley, P. *et al*., 2009. Cognitive architectures: Research issues and challenges. *Cognitive Systems Research*, 10(2):141–160.

Manzano, M., 1999. *Model Theory*. Oxford University Press.

Merrick, K., 2017. Value systems for develpmental cognitive robotics: A survey. *Cognitive Systems Research*, 41:38 – 55.

Miguel-Tomé, S., 2006. *Principios matemáticos del pensamiento natural: Teoría cognitiva de condiciones de verdad.* Gráficas Quintanilla.

Miguel-Tomé, S., 2017. *Principios matemáticos del comportamiento natural*. Ph.D. thesis, Universidad de Salamanca.

Miguel-Tomé, S., 2018a. Decision-making processes in the Cognitive Theory of True Conditions. ArXiv:1803.02476.

Miguel-Tomé, S., 2018b. Multi-optional Many-sorted Past Present Future structures and its description. ArXiv:1801.08212.
Miguel-Tomé, S., 2018c. Navigation through unknown and dynamic open spaces using topological notions. *Connection Science*, 30(2):160–185.
Miller, G., 1956. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *Psychological Review*, 63(2):81–97.
Mira-Mira, J. and García-Delgado, A., 2007. On how the computational paradigm can help us to model and interpret the neural function. *Nat. Comput.*, 6(3):211–240.
Mueller, E., 2015. *Commonsense Reasoning: An Event Calculus Based Approach*. Morgan Kaufmann, 2nd edition edition.
Newell, A., 1973. You can't play 20 questions with nature and win. In *Visual Information Processing*, pages 283–308. Academic Press.
Newell, A., 1990. *Unified theories of cognition*. Harvard University Press.
Pezzulo, G. *et al*., 2013. Computational Grounded Cognition: a new alliance between grounded cognition and computational modeling. *Frontiers in Psychology*, 3:612.
Samsonovich, A., 2010. Toward a unified catalog of implemented cognitive architectures. In *Proceeding of the 2010 conference on biologically inspired cognitive architectures*, pages 195–244.
Sescousse, G. *et al*., 2013. Processing of primary and secondary rewards: A quantitative meta-analysis and review of human functional neuroimaging studies. *Neuroscience & Biobehavioral Reviews*, 37(4):681 – 696.
Targon, V., 2016. Learning the Semantics of Notational Systems with a Semiotic Cognitive Automaton. *Cognitive Computation*, 8(4):555–576.
Thielscher, M., 2005. *Reasoning Robots:The Art and Science of Programming Robotic Agents*. Springer Netherlands.
Wang, P., 2005. Experience-grounded semantics: a theory for intelligent systems. *Cognitive Systems Research*, 6(4):282–302.
Wang, P., 2009. Analogy in a general-purpose reasoning system. *Cognitive Systems Research*, 10(3):286–296.

Appendix A

This appendix shows Object Pascal code for the definition of the types and the classes to the implementation of episodic memory. The code for the implementation of the methods of the classes is not included because space limitations.

The code for the types to the reality indicators, the temporal situators, and the connectors:

```
TReallity = (e,h);
TTemporal = (past,present,future);
TConnector = (plus, next, later ,alternatively, empty );
```

The code to implement a type to the atomic formulas of L_{MMPPF} is the following:

```
TAtomicFL0 = record

    id_objeto:integer;
    rpositioner: TReallity ;
    tsituator: TTemporal;
    objeto:TobjVistoTopo;

    end;
```

The structure of the formulas of the set of moments was implemented as a class. Thus, an object of this class is a formula of the set of moments. The class has an array of atomic formulas. This implementation avoids the use of the connector *plus* between atomic formulas. The code for the defined class is the following:

```
CMomentL0 = Class

Moment: array of    TAtomicFL0; private

public

constructor Create ; //
procedure insert (objeto:TobjVistoTopo; rp: TReallity; ts: TTemporal);
procedure transformarTemporal ( ts: TTemporal);
procedure copy (var copia: CMomentL0 ) ;
function size:integer;
function isObject (id:integer): boolean;

end;
```

To create the Log formula, it was necessary to define a new type of data first. This type is a register with two fields. One is a formula of the set of moments of time. The other is a connector.

```
TParFL0 = record

formulaM: CMomentL0;
connector: TConnector;

end;
```

The structure of the log formula has been implemented as a class. The class has an array of the previous type. The value of the field connector of a element of the array stores the connector between the formula stored in the same register and the formula of the next register. The last element of the array has always the value empty in the field connector.

```
CLogFL0 = Class

logL0:    array of    TParFL0 ;
private
public
constructor Create;
procedure insertMoment (m:CMomentL0);
function retrieve(id:integer): Tmemory;

end;
```

An object of this last class has been declared in the class of the agent. Thus, when an instance of the agent is created, it has a log formula. The method retrieve allows to determine if in the log formula exists an atomic formula about a specific agent. The method of retrieving allows for determining if an atomic formula about a specific agent exists in the log formula. If it exists, the method returns true.