AFRL-AFOSR-UK-TR-2019-0018

Self-Consciousness and Theory of Mind for a Robot Developing Trust Relationships

Antonio Chella
UNIVERSITA' DEGLI STUDI DI PALERMO
PIAZZA MARINA 61
PALERMO, 90133
IT

03/29/2019
Final Report

Air Force Research Laboratory
Air Force Office of Scientific Research
European Office of Aerospace Research and Development
Unit 4515 Box 14, APO AE 09421

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
22-03-2019	Final report	01-07-2017-31-01-2019

4. TITLE AND SUBTITLE

Self-Consciousness and Theory of Mind for a Robot Developing Trust Relationships

5a. CONTRACT NUMBER

5b. GRANT NUMBER

AFOSR FA9550-17-1-0232

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Chella, Antonio

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

UNIVERSITA DEGLI STUDI DI PALERMO
PIAZZA MARINA 61
PALERMO 90133
ITA (ITALY)

8. PERFORMING ORGANIZATION REPORT NUMBER

DUNS 513955349
NCAGE CODE AJ391

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
875 NORTH RANDOLPH STREET, RM 3112
ARLINGTON VA 22203-1954

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In this award, we investigated self-consciousness capability as a component of human-robot trust interactions. In particular, we implemented self-consciousness capabilities by allowing the robot to generate a model of its actions and abilities. We exploit the BDI practical reasoning cycle in conjunction with the theoretical model of trust proposed by Castelfranchi and Falcone. We focus on the model in NAO and Pepper robots by means of the BDI agent paradigm in the Jason framework. In the final part of the research we introduce a new concept: the role of inner speech in trustworthy human robot

15. SUBJECT TERMS

Robot self-consciousness; machine consciousness; human robot trust interactions.

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(Include area code)*
	In this award, we				

FA9550-17-1-0232

Self-Consciousness and Theory of Mind for a Robot Developing Trust Relationships

Final Report

Antonio Chella
University of Palermo, Italy
antonio.chella@unipa.it

March 22, 2019

Contents

List of Figures

Abstract

This report proposes a model for trustworthy human-robot interactions by taking into account the self-consciousness capabilities of an agent. The long-term goal of the project is to increment transparent and trustworthy interactions in human-robot teams so that that collaboration may be reliable and efficient. In general, the more a teammate is aware of the limitations and capabilities of the other teammates, the more it may be possible to establish confidence and create productive and trustworthy interactions.

In this scenario, we investigate self-consciousness capability as a component of trust interactions. In particular, we implement self-consciousness capabilities by allowing the robot to generate a model of its actions and abilities.

We exploit the BDI practical reasoning cycle in conjunction with the theoretical model of trust proposed by Castelfranchi and Falcone [20][32]. We focus on the model in NAO and Pepper robots by means of the BDI [64][14] agent paradigm in the Jason framework [13][12].

Starting from the BDI cycle, we extend the deliberation process and the belief base representation to allows the robot to decompose a plan in a set of actions associated with the needed self-knowledge to perform each action. In this way, the robot creates and maintains self-consciousness capabilities able to explain and justify the outcomes of its actions.

In the final part of the research we introduce a new concept: the role of inner speech in trustworthy human robot interactions. We describe the preliminary results obtained. However, new research is needed in order to better analyze the role of inner speech.

Chapter 1 described the theoretical model of trust employed allowing self-consciousness capabilities, along with an early implementation on the NAO robot. Chapter 2 generalizes the approach in the previous chapter by introducing a cognitive architecture to trust human-robot team interactions based on robot self-consciousness. Chapter 3 exploit the implementation issues of the architecture in the BDI paradigm by employing Jason and CArtAgO. Chapter 4 extends the architecture by taking into account the problem of knowledge acquisition at runtime. Chapter 5 further expands the problem of incremental knowledge acquisition at runtime. FInally, chapter 6 describes the preliminary results obtained by introducing the inner speech in the proposed cognitive architecture.

Chapter 1

A Theoretical Model of Trust

1.1 Introduction

Purposeful social interactions in human-robot teams are based on the broad concepts of autonomy, proactivity, and adaptivity. These concepts allow the team to choose the productive activities to be performed to pursue the required goals. From a social point of view, the members of the team have to choose which actions to perform and which ones to delegate to the other components of the team.

Therefore, human-robot interactions involve not only decisions on the action to undertake to reach a goal, but also the actions to delegate to other teammates. These decisions cannot be defined at design time, for reasons ranging from the composition of the environment to the characteristics of the interacting entities. A robot cannot be simply pre-programmed to carry out tasks whose knowledge is acquired during execution. To face with this kind of robot self-adaptation, we need to take into account the state of the robot during execution, its knowledge about itself and the environment and the knowledge about the other teammates.

Interactions with other teammates are based on the knowledge about the capabilities of the other mates, on the interpretation of the actions of the other mates concerning the shared goals and also on the mutual level of trust. Trustworthiness is thus an essential element to choose actions to undertake or to delegate to other members.

According to Castelfranchi and Falcone [20],[32] *trust* is assigned on the basis of specific evaluations regulating the behavior of the agents. Trust is tightly related to delegation, and it refers to a mental state of the trustor towards the trustee. At the origin of trust are *direct experience*, *recommen-*

dations, reputation and *inferences reasoning.*

In a dynamic context where human-robot interactions depend on time, reputation and direct knowledge may not apply and inferences and reasoning become essential elements [34].

The approaches in literature for the development of self-adaptive systems are commonly related to multi-agent systems [79][78]. In the following, we overview the adopted trust theory, the BDI paradigm and we describe the employed elements of Jason and CArtAgO for our proposed implementation.

1.2 Concepts about trust

According to the trust theory proposed by Castelfranchi and Falcone [20][32][33][22], we take into account:

- *trust* as *mental attitude* allowing the prediction and evaluation of other agents' behaviors;
- *trust* as a *decision* to rely on other agent's abilities;
- *trust* as a *behaviour*, i.e., an intentional act of entrusting;

Thus, a set of different figures take part in the model:

- the *trustor* is an intentional entity, i.e., a cognitive agent based on the BDI agent model that pursues a specific goal;
- the *trustee* is an agent operating in the environment;
- the *context* where the trustee performs actions;
- τ - is a "causal process" performed by the trustee and composed by an act α and a result p. The goal g_X is included in p and sometimes coincides with p.
- the *goal* g_X - is defined as $Goal_X(\mathbf{g})$.

The function of trust is *the trust of a trustor in a trustee for a specific context to perform acts to realize the outcome result.* The model is described by a five-part figures relation:

$$TRUST(\text{X Y C } \tau \ g_X) \tag{1.1}$$

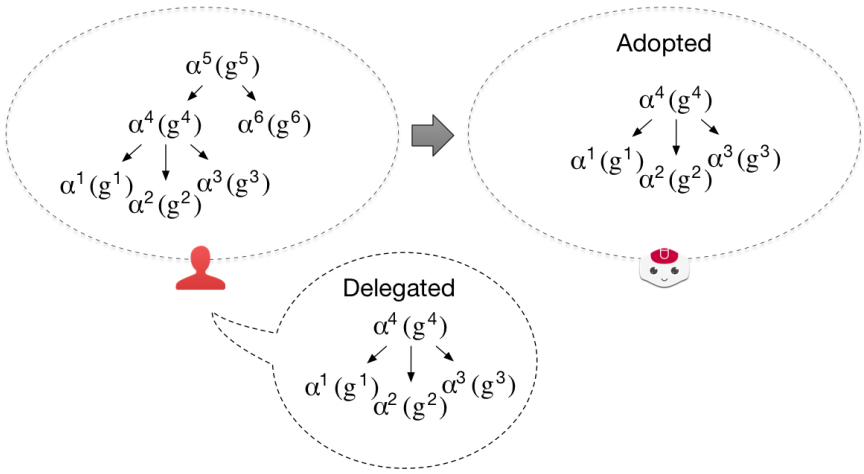


Figure 1.1: Level of Delegation/Adoption, *Literal Help*

where X is the trustor agent, Y is the trustee agent. X's goal, or briefly g_X, is a critical element of this model of trust. In some cases, the outcome can be identified with the goal.

Trust is the mental counterpart of the delegation, in the sense that trust denotes a specific mental state composed of beliefs and goals, but it may be realized only through actions. Delegation is the result of a decision taken by the trustor to achieve a result involving the trustee.

Different levels of delegation are hypothesized [21, 31], ranging from a situation in which the trustor directly delegates the trustee to cases in which the trustee autonomously acts on behalf of the trustor. An interaction is a continuous operation of adoptions and delegations. In particular, we focus on the *literal help*, shown in Figure 1.1.

In the *literal help*, the client (trustor) and the contractor (trustee) act together to solve a problem. The trustor asks the trustee to solve a sub-goal by communicating the trustee the set of actions (plan) and the related result. In the *literal help* the *trustee* tightly adopts all the sub-goals the *trustor* assigns to him [21][31].

The notion of behaving *on behalf of* is one of the key ideas in the multi-agent systems paradigm. Agents' features, such as autonomy, proactivity,

and rationality are useful concepts that make trust-based agents ideal candidates be used in applications such as human-robot interaction. Adopting the multi-agent paradigm, we design and develop a multi-agent system in which many agents are deployed in the robot involved in the application domain.

1.3 BDI Agents

The BDI model approach was proposed as a model of practical reasoning [14], while Jason [12] is an Agent-Oriented Language inspired by models of behavior. The BDI Agent-Oriented Programming is in facts a commonly employed paradigm for the implementation of agents.

According to the BDI model, an agent is characterized by its own beliefs, desires, and intentions:

- beliefs are information about the working area or the world of the agent;
- desires are the possible *states of affairs* of an agent. A desire is not a *must to do* action, but it is a condition influencing other actions;
- intentions are the *states of affairs* an agent decides to perform. Intentions can be considered as operations that can be delegated to other teammates.

An *intentional system* is a system predictable through beliefs, desires and intentions [30].

The decision-making model underpinning BDI systems is the *practical reasoning*, a reasoning process for actions, where agents' desires and beliefs supply the relevant factor [15]. Practical reasoning, in brief, consists of two activities:

- *deliberation and intentions*;
- *means-ends reasoning.*

Each activity can be expressed as the ability to fix behavior related to intentions and deciding how to behave.

```
1.  B ← B0;        /* B0 are initial beliefs */
2.  I ← I0;        /* I0 are initial intentions */
3.  while true do
4.    get next percept ρ via sensors;
5.    B ← brf(B, ρ);
6.    D ← options(B, I);
7.    I ← filter(B, D, I);
8.    π ← plan(B, I, Ac); /* Ac is the set of actions */
9.    while not (empty(π) or succeeded(I, B) or impossible(I, B)) do
10.     α ← first element of π;
11.     execute(α);
12.     π ← tail of π;
13.     observe environment to get next percept ρ;
14.     B ← brf(B, ρ);
15.     if reconsider(I, B) then
16.       D ← options(B, I);
17.       I ← filter(B, D, I);
18.     end-if
19.     if not sound(π, I, B) then
20.       π ← plan(B, I, Ac)
21.     end-if
22.   end-while
23. end-while
```

Figure 1.2: Practical reasoning taken from [13] .

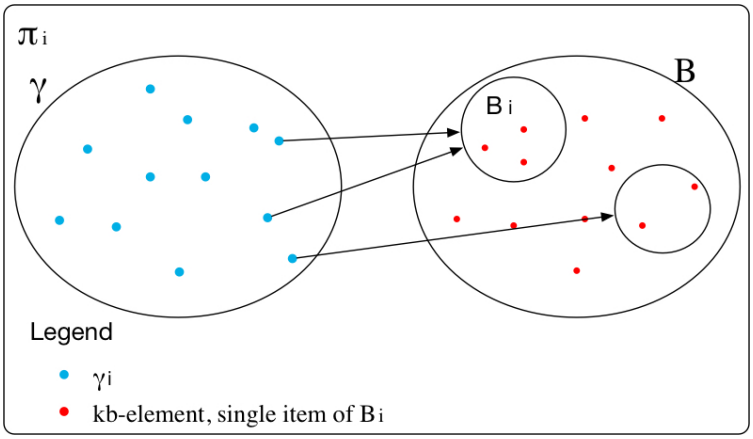


Figure 1.3: Mapping actions onto beliefs

1.4 Jason and CArtAgO

The Jason programming language extends the agent-oriented language AgentSpeak. A BDI agent can sense its environment and update its internal belief base accordingly. Jason implements the BDI paradigm, and thus the components of the language are the beliefs, the desires and the intentions of the model. An agent enters an infinite loop of perception, reasoning, and actions to satisfy its goals [12].

CArtAgO [66] is a general purpose framework based on Agents and Artifacts meta-model. Briefly, it allows the development of virtual environments for BDI systems based on artifacts.

1.5 Self-conscious BDI agents

In this project, we investigate the self-consciousness abilities of the entities as valuable ingredients of a trustworthy relationship.

In our proposed model, the robot has the role of the trustee and the human mate is the trustor. The human mate trusts the robot and delegates goals to it. We assume the level of trust as related to the robot's ability to explain and justify the outcomes of its actions, especially when the robot fails.

As previously stated, our approach is based on the employment of a multi-agent paradigm and the BDI theory to model trust-based interactions in a partially unknown environment. We take inspiration by the theoretical

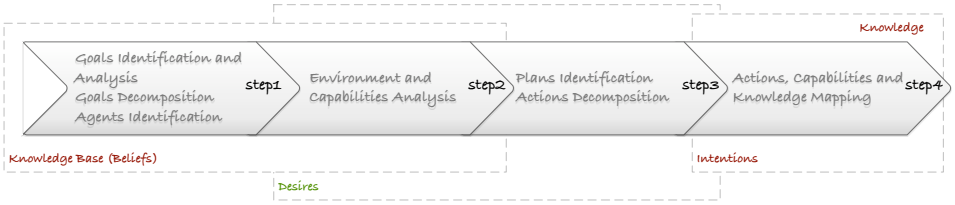


Figure 1.4: The first layer of the architecture: analysis and design

model by Castelfranchi and Falcone and we implement parts of it by the BDI cycle [13] where we include the self-consciousness capabilities at the basis of robot explanation and justification. The module for the robot self-consciousness includes components allowing the robot to reason about its knowledge to perform actions or behaviors.

In the implementation, to establish a transparent and trustworthy interaction, each action is then coupled with the concepts related to itself, that the robot needs to complete that action. Then, the robot may explain and justify at each moment whether and why action is going wrong and, most important, it may motivate faults.

For instance, let us suppose a person sitting on a desk in a room with the goal of going out of the room. The goal may be pursued by some simple actions like standing up, heading to the door, opening the door with the key, going out. For each action, the performer employs the knowledge about the external environment and herself and her capabilities. She has to be able to stand up, to know that a key is necessary for opening the door and she has to own that key and so on. Before and during each action, the person continuously and iteratively checks and monitors the conditions of her actions, and in particular if she already has the knowledge of the conditions allowing the actions to be undertaken and finished.

To this aim, we modify the model from [20]:

$$TRUST(X\ Y\ C\ \tau\ g_X), \quad \text{where} \quad \tau = (\alpha, p) \quad \text{and} \quad g_X \equiv p; \tag{1.2}$$

Here, τ is no longer based on the couples of actions and results, but it combines the trust theory model with a self-consciousness approach: τ is now a couple of a set of possible plans π_i and the related results p_i. This model is implemented in the BDI paradigm by breaking down actions and results in a combination of multiple arrangements of plans and sub-results.

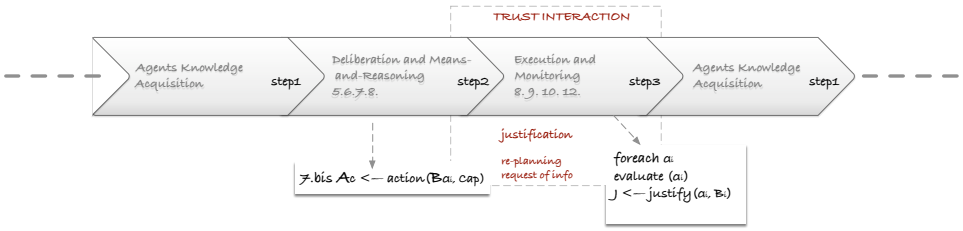


Figure 1.5: The second layer of the architecture: runtime

The model of τ is formalized as:

$$\tau = (\alpha, p) \quad \text{where} \quad \alpha = \bigcup_{i=1}^{n} \pi_i \quad \text{and} \quad p = \bigcup_{i=1}^{n} p_i \tag{1.3}$$

Each atomic plan π_i is the composition of action γ_i and the portion of belief base B_i for pursuing it. It is formalized as:

$$\pi_i = \gamma_i \circ B_i \Rightarrow \alpha = \bigcup_{i=1}^{n} (\gamma_i \circ B_i) \tag{1.4}$$

where B_i is a portion of the initial belief base of overall BDI system. The $\circ$ *operator* represents the composition between each action of a plan with a subset of the belief base (Figure 1.3)

The framework has been implemented in the robotic platform NAO by exploiting Jason [13] and CArtAgO [66]. The environment model has been generated through the perceptual module of the robot NAO. The CArtAgO artifact allows the robot to perform operations in the real world.

The implementation is supported by a two-layered architecture (Figures 1.4 and 1.5).

Before illustrating these two layers, it is worth introducing the employed reference model. Since we adopt a multi-agent paradigm, we take into account the human mate as an agent. Human mate is thus a part of the MAS the robot is part of and with whom it interacts to accomplish a specific task.

The environment is the other crucial element of the model, as we include in the environment the ensemble of intentional and unintentional agents. An intentional agent is an agent (human, robot or software entity) in the environment that interacts intentionally and autonomously with the robot. An intentional agent has objectives, it may change the state of the surrounding environment while interacting and performing the appropriate actions to achieve its objectives. Also, an intentional agent has capabilities (i.e., the

ability to do certain things [29]) that are in correlation with the actions that can be taken within a plan to achieve a goal.

The environment thus evolves by the interactions among its components. An unintentional agent is any resource or object that has a state of its own, and the intentional agents can target it. The environment is dynamic, changing over time as the result of agents' actions.

It is to be noted that the robot is considered as part of the environment: it has representation of the external environment, and also a representation of its internal state. This way to handle the environment by taking into account the robot itself is the critical points of the proposed self-consciousness model of the robot.

The two-layered architecture is based on the MAPE (Monitor, Analyze, Plan and Execute) loop [4]. Each agent continually monitors a portion of the environment is interacting with, it analyzes and chooses the objectives to pursue and the action to undertake.

In Figure 1.4, the first and the second steps of the architecture aim at identifying and analyzing the structural part of the system; the third step is the dynamic part, and the fourth step is the core of the robot self-consciousness capabilities.

During the first step, the goals of the systems are defined at a high level and then decomposed and refined into more fine-grained goals by an AND-OR decomposition. See also [16][59].

The second step aims at analyzing the environment made up of objects and intentional agents with their internal state. For each component of the environment, we analyze its state and the actions (known at design time) that may cause a change in the state. We follow the approach previously proposed in [28][29] that involve concepts, predicates, and actions to describe the entities involved in a domain. Briefly, A *Concept* is a term used in a broad sense to identify "anything about which something is said." A *Predicate* is the expression of a property, a state or a constraint; It serves to clarify or to specify a *Concept* or to infer a restriction on an *Action*. An *Action* represents every actions made on a concept to pursue an objective, and that may change the state of a *Concept* [49]. These two steps build the knowledge base, i.e., the belief base the agents employ at runtime for reasoning about their actions.

In the third step, the functional decomposition of goals is performed. The result is a set of plans and related actions to pursue each goal, which is assigned to the robot in case it possesses the suitable capabilities (known at design time) to reach the objective.

In the fourth step a *Assignment Tree* is created. An example of the

Assignment Tree is provided in the following Section; it is a model of the relationships among actions, the set of capabilities and the knowledge on the environment useful for performing a specific action. This final step also contributes to the creation of the belief base whereas step 2 and 3 contribute to form all the possible *state of affairs* of the human-robot team, what in the BDI logic is called *desire*. The actual objective the robot is assigned (or the one it commits to). The intentions are created during the fourth step as it is shown in the Figure.

The second layer refers to the execution time. The robot system has been analyzed and designed and then put in execution (See Fig. 1.5).

The agents involved in the system acquire knowledge at runtime. Concerning the BDI cycle, they explore the belief base and the initial goals they are responsible for (points 1. 2. 3. 4. - Fig. 1.2). The module implementing deliberation and means-and-reasoning (points 5. 6. 7. - Fig. 1.2) is now extended. At this point, while executing the BDI cycle, the tail of actions for each plan is generally processed to let the agent choose the action to perform. Since we are interested in the knowledge useful for and involved in each action, we add the new function:

$$A_c \leftarrow action(B_{\alpha_i}, Cap) \tag{1.5}$$

where B_{α_i} and Cap are the portion of the belief base related to the action α_i and the set of agent's capability for that action.

The third step of execution and monitoring, implies the points 8. 9. 10. 11. 12 of the BDI cycle that we extended with the capabilities to evaluate the statements *impossible* (I, B) and $\neg$ *succeeded*(I, B) (ref. point 9.)

In this step, when the trust interaction takes place, the robot is endowed with the self-consciousness abilities to re-plan, explain and justify or request supplementary information to the human mate.

The added functions in the case of *explanation*, are shown in the following algorithm:

Algorithm 1

1: *foreach* α_i :
2: *evaluate*(α_i);
3: $J \leftarrow$ *explain*(α_i,B_{α_i});

Figure 1.6 details all the elements of our trust model.

Summarizing, τ is the goal that the trustor delegates to the trustee; then, the BDI agent is assigned the responsibility to perform the actions γ_i

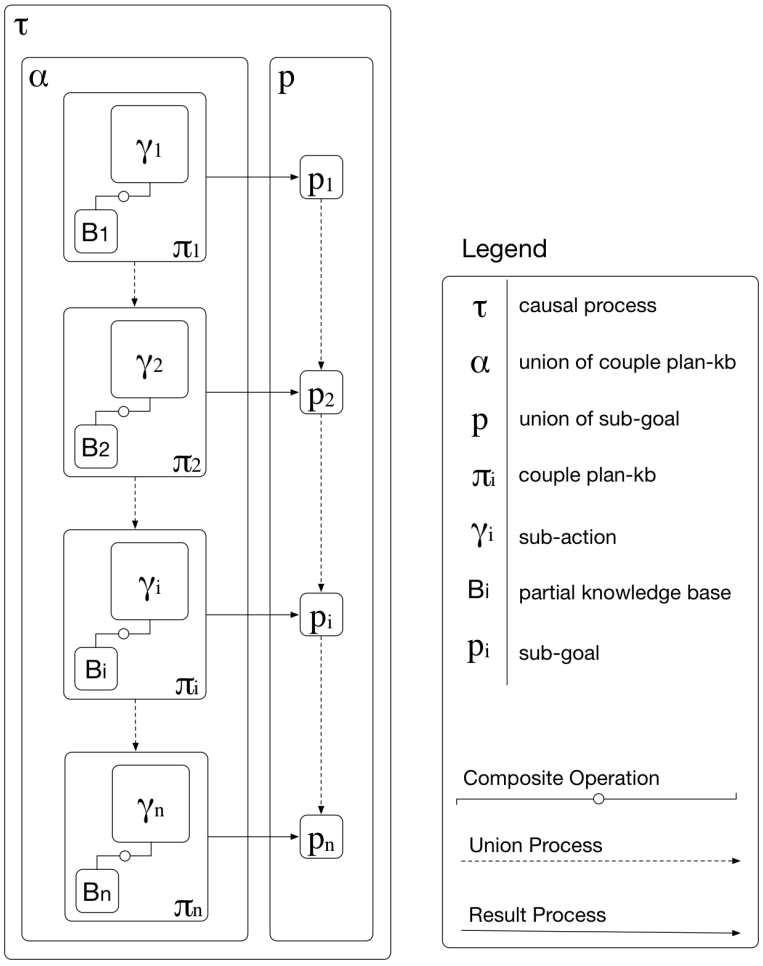


Figure 1.6: All the element of the trust model

included in τ made up of the following elements:

- Jason Agent - the BDI agent that manage the NAO robot through the AgentSpeak formalization [13] with the following:
  - ASL Beliefs - the set of beliefs that include the knowledge about the environment of the agent and its inner capabilities;
  - ASL Rules - the beliefs related to norms, constraints and domain rules;
  - ASL Goals - the list of goals of the application domains, i.e., the list of desires in BDI;
  - ASL Plans - the logic inference needed to perform actions ;
  - ASL Actions - the agent commitments of sequences of actions, hence plans;
- CArtAgO Artifact - it allows the agent to perform a set of actions in the environment. The CArtAgO virtual environment represents the environment through the beliefs acquired by NAO's perception module;
- CArtAgO @Operation - it is employed to perform agent's actions in the environment.

The proposed trust model with the agent with self-consciousness abilities has been implemented through the BDI cycle on a reference model of environment where the critical point is the robot, and by modeling the inner states of the robot as as part of the environment.

1.6 The robot in action using Jason

The case study described in this Chapter concerns a human-robot team whose goal is to carry objects from a position to another in the same room. The work to be done should be exploited in a collaborative and cooperative way. In this setup, we consider the case in which the robot is delegated by the human mate to pursue a specific goal.

The environment is a set of objects marked with the landmarks needed for the NAO to work. The set of capabilities is made up by taking into account the NAO capabilities: for instance, to be able to grasp a small box. The NAO also is endowed with the capability of discriminating the dimensions of the boxes.

In this case, only one agent is managing the robot, with the responsibility of carrying an object to a given position. The human mate, i.e., the other agent of the team, indicates the robot the object and its position.

From the decomposition of goals in sub-goals, and then in plans and actions, and from the mapping of capabilities, actions, and beliefs, we obtain the result shown in Fig. 1.7.

The Figure represents a portion of the *assignment tree* introduced in the previous section. The main goal *BoxInTheRigthPosition* is decomposed in three sub-goals, namely *FoundBox*, *BoxGrasped ReachedPosition*.

Let us consider the sub-goal *ReachedPosition*: two of the actions that allow pursuing this goal are: *goAhead* and *holdBox*[1].

The NAO goes ahead towards the goal and at the same time it holds the box. The beliefs associated with these actions refer to the concepts in the knowledge base affecting these actions. In this case, one of the concepts is related to box, with its attributes as the dimension, color, weight, initial position and so on. The model of the environment (see Sect. 1.1) contains the possible actions to be made on the box, for instance *holdBox*, and the set of predicates representing the beliefs for each object, for instance *hasVisionParameters* or *isDropped*. The beliefs (*visionParameter* and *dropped*) are associated with the action *holdBox* through a relation number (1.4).

In the following, a portion of code related the example:

```
    τ:  +!ReachedPosition:  true ← goAhead; holdBox.
γ1) +!goAhead:  batteryLimit(X) & batteryLevel(Y) & Y < X ← say("My battery
is exhaust.  Please let me charge.").
γ1) +!goAhead:  batteryLimit(X) & batteryLevel(Y) & Y ≥ X ← execActions.
B1:  batteryLimit, batteryLevel

γ2) +!holdBox:  dropped(X) & visionParameters(Y) & X == false ← execAct(Y).
γ2) +!holdBox:  dropped(X) & visionParameters(Y) & X == true ← say("The box
is dropped.").
B2:  dropped, visionParameters
```

Fig. 1.8 reports some pictures showing the execution of the NAO.

[1]For space concerns we only show an excerpt of the AssignmentTree diagram, so only a few explanatory belies for each action are reported.

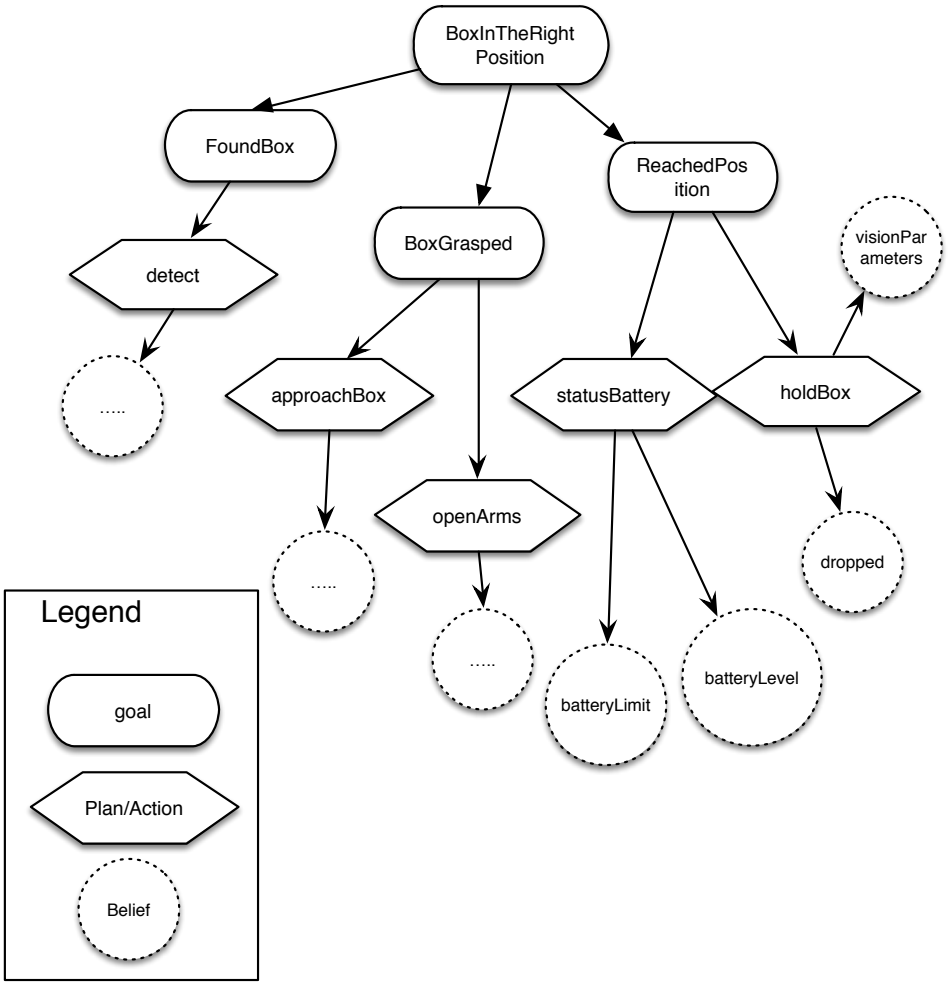


Figure 1.7: A portion of the assignment tree for the case study

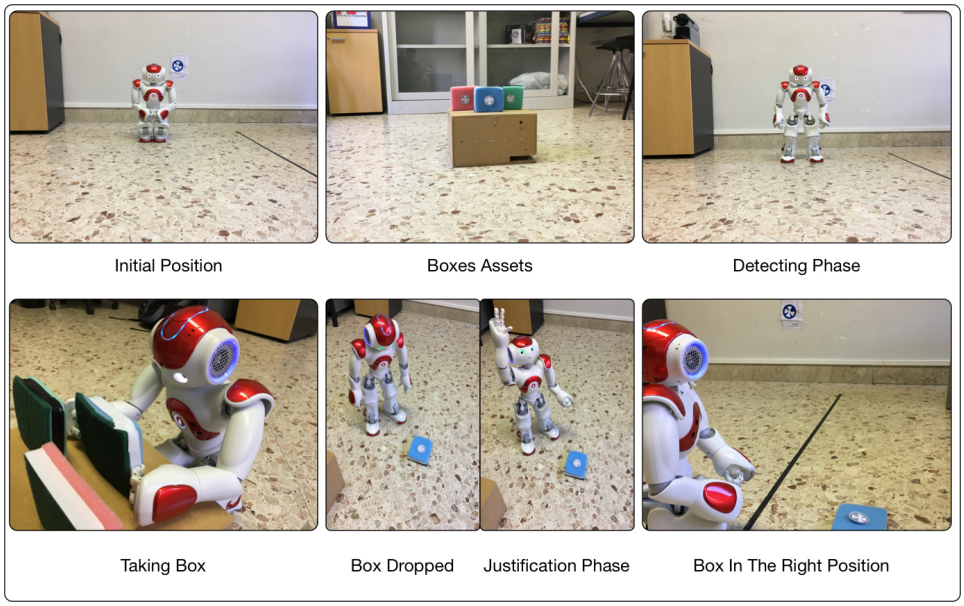


Figure 1.8: The NAO working on the *BoxInTheRightPosition* goal and the justification

1.7 Discussion

The current literature explores the concept of trust, how to implement it and how to employ it generally from an agent society viewpoint in an open and dynamic environment. So, research is mostly focused on organizations in which multiple agents interact with each other and choose which action to take by considering a certain level of trust in each other. Instead, in our case, while sharing the concept of an open and dynamic environment, we focus on the theme of human and robot teammates, and we explore the two-way role of human-robot and robot-human.

In [58], decision making based on trust evaluation is examined through a decision-theoretic model that allows controlling trust activities. The leading point is to make agents able to evaluate trust. Reputation mechanism enables the trustor to make a better evaluation. Our work shares the same objectives but it focusses at a different level of abstraction. We endow the agent with self-consciousness abilities to give the trustor a means for delegating or making the action by himself. We propose self-consciousness as an autonomous form of interaction and cooperation.

In [74] the trust model is applied to virtual organization, and authors

employ a probabilistic theory that considers parameters calculated from past interactions when information lacks or it is inaccurate. In our case, we pose the basis for giving the trustee the ability to ask for help when it does not own the necessary knowledge to perform the delegated action. Thus, we let the possibility of the trustor to evaluate the operations of trustee. It is no longer the trustor concerned about assessing trust to the trustee, but it is the trustee who provides the means to do so.

In [41] a trust model based on reputation is presented, that it allows creating a measure for trust that can be used in different circumstances. This model overcomes the problem of evaluating trust in a dynamic environment where it is difficult to consolidate the knowledge of the environment. The model we propose is constrained by the fact that the trustor establishes a level of trust by observing the other agent. However, endowing the trustee with self-consciousness abilities gives the trustor the possibility to better evaluate the work of the other mate.

A different approach is proposed in [68]: here, the authors use meta-analysis for establishing which features of the robot may affect the trust relationship form, the point of view of the human mate. The robot is a participant to the team but not an active part of it. From this work we may outline the main difference of our proposed trust model against all the others, as we consider the trustee (agent, robot or whatever else) an active autonomous entity in the interaction.

The primary element of our work is to equip the robot with self-consciousness abilities that allow it to be aware of its skills and failures. We have chosen an explicit self-consciousness feature as the ability to explain justify oneself in the case of failure. We may extend the model with the ability to ask for help when the trustor's requests do not fall within the trustee's knowledge and the ability to autonomously re-planning.

Our trust model takes inspiration from the work by Falcone and Castelfranchi and has been integrated with a BDI-based part of the deliberation process to include self-consciousness. The self-consciousness ability is obtained by joining the plan a BDI agent commit to activating with the knowledge base useful for it.

The model is based on a two-level architecture; the two levels allow to maintain distinct the theory developed by its implementation part. In this way, the trust model can be developed on any robotic platform and with any programming language. We have chosen Jason and CArtAgO because they fully support the BDI theory. Besides, it allow us to implement, without significant changes to the agent language paradigm, the elements of the reference model for the environment we previously defined.

Chapter 2

A Cognitive Architecture for Human-Robot Teaming

2.1 Introduction

As stated in the previous Chapter, trustworthy human-robot interactions are rich in research ideas and open problems. Fig. 2.1 represents the typical situation of a human-robot team working in a shared environment to reach a common objective.

Our long-term research goal focuses on analyzing and developing systems where humans and robots collaborate in a human-like fashion. In the following, a list of possible activities of a teammate is reported:

- she knows what she has to do, i.e., she knows the overall goal of the team;
- she knows what she wants to do, hence, she intentionally decides which goal or subgoal to commit;
- she knows what she can do, i.e., she is aware of his capabilities and accordingly she selects the goals she can reach and all the right plans and actions;
- she is aware of the surrounding environment;
- she associates any new element in the environment to what is already in her knowledge base. Generally, she owns a knowledge base that includes a large number of elements, only a few of them are of interest for what she was doing and for the domain she is working in;

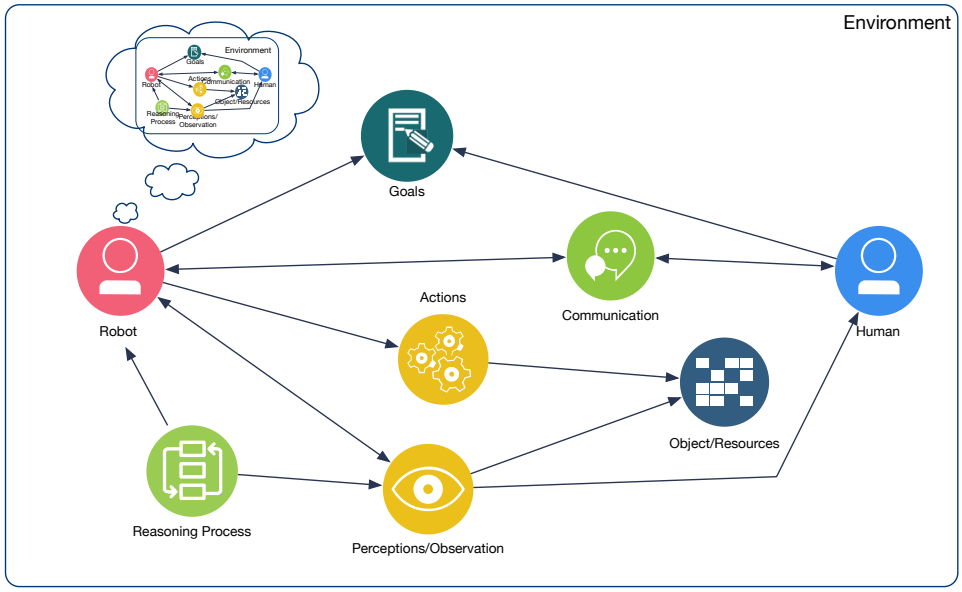


Figure 2.1: Human-Robot teaming scenario in an environment composed of cognitive agents, objects and resources.

- she communicates with the other team members to update her knowledge base about the environment;
- she explicitly or implicitly delegates action to other mates;
- she asks how to do something she is not able to do;
- she observes what other team members are doing and, if the case, she anticipates actions or cooperates with other mates;
- she anticipates whether the other mates can carry out the work and if the case she takes the initiative;
- she observes what the other mates are doing and decides what to do and whether to do something basing on her own emotional or stress state, on the trust level she has on the others and herself and on the possible mental state she may possess;
- she understands if some operative condition for pursuing objective changes. In this case, she can re-plan or create new plan from memory and experience;

- she explains what she is doing and why and, if the case, why she is not able to do something;
- she learns from experiences and stores all the information about the outer and inner continuous changing world.

During each one of the listed activities, the teammate performs different processes that account for a description of the world in time concerning vision, speech understanding, learning, state of mind, decision making.

Taking into account these aspects in a human-robot teaming, a cognitive architecture means to analyze and implement at least these processes: *(i)* knowledge acquisition and representation, including memory management; *(ii)* representation of the external environment; *(iii)* plans selection and creation; *(iv)* learning.

We propose to integrate self-consciousness in a cognitive architecture for human-robot teaming to implement some of the previously listed team features in a robot system.

This work focuses on theoretical and implementation aspects. We aim at identifying an abstract cognitive model and the related implementation counterpart. The contribution of this Chapter lays in how knowledge representation and acquisition dealt with a robot able to generate a simplified self-consciousness.

Two approaches have been considered in the cognitive process area about cognitive architectures, i.e., the *cognitivist* and the *emergent* approaches [72][24][43]. The first approach relies on the perceive-decide-act loop and the symbolic representations to instantiate operations devoted to implementing agents behaviors and decision processes. The second approach considers cognition as a dynamic emergent process implying self-organization: emergent approaches take into account anticipative skills more than knowledge acquisition, and the physical instantiation of the model as a main factor.

ACT-R [3] is based on five specialized modules, where each module processes a different kind of information. ACT-R thus decomposes the cognition process and shows how to integrate the modules to generate a complete cognitive process. ACT-R introduces the chunk as a declarative unit of knowledge. SOAR [47] is based on a cyclic process that includes the production and the decision processes. A decision cycle follows each production cycle; this guarantees that every change in the state of affairs can be accounted, but deadlocks sometimes may occur. EPIC [44] replicates motor and perception systems with several processes running in parallel, multiple rules can fire at the same time. ADAPT [8] is tailored for robotics and it includes adaptive dynamics and concurrent real-time communications. The

learning mechanism follows a sequential process of search and selection that let the decision process be sequential and less prone to changing conditions and self-adaptive requirements. Emergent architectures as AARs [23] and GWCA [69] presents limitations when the system increases.

An interesting hybrid architecture, the *Humanoid Robot Cognitive Architecture*, presents a three-layered architecture where long-term memory, short-term memory, perception and task planning subsystems interact and communicate via an execution manager. Knowledge is globally handled among all the modules, learning and effective low-level implementation of tasks are still a work in progress.

In this Chapter, we present COAHRT, a cognitive architecture endowed with modules implementing the *monitoring, analyzing, planning, action* cycle along with modules devoted to representing memory involved in the decision and learning process at runtime and in the realization of self-consciousness. The architecture is conceived in a highly modular fashion. Each module in COAHRT is mapped to the implementation level by using agent-oriented technology and the BDI paradigm [64][12].

2.2 COAHRT - COgnitive Architecture for Human Robot Teaming

The definition of COAHRT (COgnitive Architecture for Human-Robot Teaming) results from the integration of the features of existing architectures [35][3] with an extended version of the perception-action cycle to add modules for handling decision process and memory. In Fig. 2.2, a cognitive agent employs inputs from the environment perception and from memory for choosing which action to execute. The agent chooses actions to perform after a reasoning process, and it executes and continuously observes the results of its action on the environment. To integrate self-consciousness aspects in the architecture, we added elements in the decision and memory modules.

We represent knowledge by including the objects in the environment, the goals to be pursued and the motivations to execute a specific action. The knowledge representation allows us considering the environment as composed of objects, other cognitive agents and also the agent inner state. All these elements are parts of the agent's self-consciousness that triggers the agent decision process. Continuous observation and perception allows the agent to update knowledge during the execution phase.

The ANTICIPATION module of Fig. 2.2 generates the anticipation of the

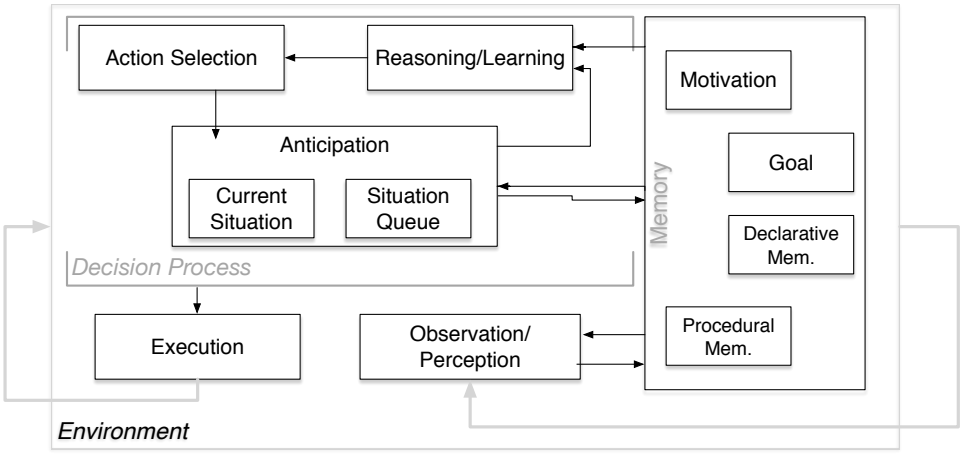


Figure 2.2: The Cognitive Architecture for Human-Robot Teaming

action result, i.e., the post-conditions on the state of affairs at the end of each action. This module allows anticipating the other cognitive agents' behaviors and actions. In so doing, we implement a simplified version of a theory of other agents' minds.

In the ANTICIPATION module we include elements for the generation of the CURRENT SITUATION and a SITUATION QUEUE of possible situations, generated from the knowledge base and gained when the current situation is not applicable.

The MOTIVATION module includes elements related to the inner state to be considered during the decision process. Some motivation elements that lead to a purposeful decision are the emotional state and the level of trust in the others and in itself.

2.3 Knowledge Acquisition and Representation by Self-consciousness

We model and update the agent knowledge base at runtime. Knowledge is necessary for the decision process and for communicating and interacting with other agents. Besides, knowledge representation let the agent to be able to understand what it does not know.

In the subsection, we illustrate multi-agent technological aspects for implementing the reasoning cycle at the core of the decision process. Multi-agent paradigm is employed for developing the system level part of COAHRT.

Then, we discuss the experiment towards self-consciousness abilities, two different ways of representing and handling knowledge at runtime and finally some hints to the motivation module concerning how emotions may trigger the decision process.

2.4 Implementing the decision process

The proposed cognitive architecture includes the modules *knowledge* and *memory*, which in turn is composed of two parts: the *long term memory* and *short term memory*, and other modules as the *perception module*, the *communication system* and the *reasoner* that allows the robot to choose by taking into account the retrieved data.

The cognitive architecture deliberates the robot behavior by the *planner* which interacts with the context in which the agent is plunged. We employed the multi-agent systems paradigm to implement the architecture; each *module* is a *agent* which interacts with all the others for achieving its objectives and at the same time the overall system objective.

As in the previous Chapter, we employ the *Belief-Desire-Intention model* (BDI) [64] to describe the reasoning process of each agent. We employ Jason [13] as a programming language that implements BDI agents.

The decision-making model underpinning BDI systems is known as *practical reasoning*, a reasoning process to do actions, where agents' desires and agents' beliefs supply the relevant factor [15]. Practical reasoning consists of the activities of deliberation and intentions and of means-ends reasoning.

Fig. 2.3 shows this multi-agent model that maps COAHRT. In our approach, each agent is orchestrated, regarding knowledge and memory access, by the *controller* agent *Knowledge Manager*, implementing planning and the reasoning functions. The module ensures the knowledge necessary to allow the collaboration among the agents.

Across the extended reasoning cycle, each agent employs its experience to perform the action and to reason on the situation by analyzing inner states and external perceptions. Once the plan selects an action, it is executed by changing the state of the environment and also the inner state of the robot.

2.5 Implementing self-consciousness abilities

With the aim to endow the robot with self-consciousness abilities, we model tasks as sets of beliefs and intentions. The robot is able to identify failures in

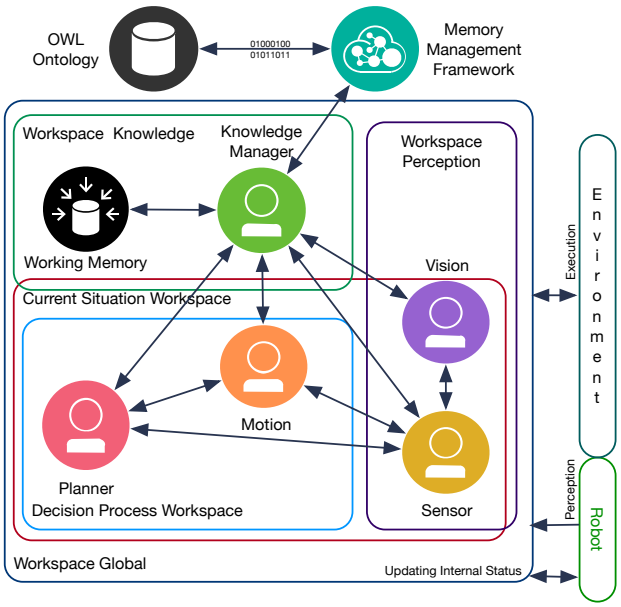


Figure 2.3: Multi-agent view implementation of COAHRT Architecture

executable plans and actions and to explain and justify the incompleteness of its performances.

Perception and external stimuli are modeled in the knowledge ontology of the robot. When a goal is detected, the related beliefs are generated from the ontology by allowing the robot to select the appropriate plan. Each action modifies the state of the environment and the robot inner state.

Our approach involves the knowledge that, based on the selected beliefs, allow the robot to identify the motivations for which a plan could fail. In facts, we keep separate the reasoning component from the environmental managing tools; these two components are implemented respectively by Jason [13] and CArtAgO [66]. In details, Jason implement the BDI agents and it manages the interactions among them, whereas CArtAgO manages the interaction with all the resources and the objects in the environment.

Beyond simple actions, each plan involves *context variables* representing the preconditions to be satisfied to perform the actions of the plan. When one of these variables, instantiated by the perception module, does not satisfy the preconditions (i.e., it has an unexpected value or it is false), then the plan execution fails, and the robot is able to infer the motivations of the failure, thus implementing a simple form of self-consciousness. The motivations of failures are then sent to the other members of the team which may

solve the situation by enforcing collaboration,

2.6 Handling Knowledge in the Cognitive Architecture

The robot knowledge is based on a set of concepts, individuals, and roles with semantic relations between concepts and their properties, modeled by an ontology. The conceptual level is the terminological box which defines the ontological entities concerning the general schema over the facts in the domain; it is the abstract description of concepts, properties, and relations. The low level is the assertional box that includes facts in the actual context: it is the set of individuals corresponding to concrete objects in the environment.

We consider two different methods to acquire knowledge: the first one depends on the interactions with the human mate, which helps the robot to infer the meaning of new perceived objects. In this case, the verbal interaction allows to disambiguate the sense of the percept, for which each feature has to exist as a term of the ontology. The features are in the terminological box, and the interaction allows to disambiguate the perceived entity as a new concept by the features emerging during a conversation. If a feature is not modeled in the ontology, the robot cannot recognize objects.

The second method is based on a probabilistic evaluation of knowledge acquisition that overcomes this problem. In this case, the robot can infer a new term and its correct allocation at the conceptual level.

2.6.1 Knowledge acquisition by interaction.

The scenario we considered for knowledge acquisition by interaction involves the robot performing a specific task and the human mate that provides the information the robot needs to complete the task.

The robot has all the needed knowledge about the objects in the environment, as an object is conceptualized when the robot recognizes all its features. Once it detects an object, the robot queries its knowledge base to retrieve the corresponding features. When all the features are retrieved, then the object is conceptualized, else an interactive linguistic session starts with the human mate. In this case, the robot steers the interaction by making specific queries to the human.

If the information provided by the human mate is exhaustive, then all the features are known, and the object is conceptualized. Otherwise, the

interaction ends, and the conceptualization fails.

The knowledge about the object is not related to its features only, but also to the actions the robot could perform on it and the context variables. For example, if the robot has to move an object from a start position to an end position, then the reasoner generates the suitable plan to execute the task by retrieving the actions the robot may perform on the object. Then, it verifies if the context variables are satisfied and hence if the plan is executable.

2.6.2 Probabilistic evaluation for knowledge acquisition.

We investigate a probabilistic approach for knowledge acquisition to make the robot able to understand and acquire new perceived entities in a dynamic context by updating its general knowledge.

Three different phases are defined: (*i*) an entity is perceived and formalized in a suitable way for the robot; (*ii*) the entity is classified as new or redundant, and finally (*iii*) a new entity is linked in the correct knowledge context, leading to its semantic disambiguation. In this section, we show the final step.

We inspired to the Pitman-Yor Process (PYP) [61] that generates distributions for language modeling and grammar induction. In fact, a PYP produces power-law distributions that resembles those employed when linking concepts to contextual knowledge. The main idea is that the robot ontology is a set of *fragment trees* (the set of ontological nodes and triples) linked to the features of a new entity. A Pitman-Yor Process defines a distribution over sequences of fragment trees to estimate the correct linking.

The model we implemented is build upon the principles of *Tree Substitution Grammar (TSG) induction* [27], that define a power-law distribution over a space of production rules that take the form of *elementary trees*. An elementary tree of the grammar is a tree of height ≥ 1 whose each internal node is labeled as *nonterminal* symbol and each leaf is labeled either a *terminal* or a *nonterminal*. Differently to the Context-Free Grammar (CFG) in which nonterminals can rewrite only immediate children [42], the TSG nonterminal can rewrite entire tree fragments. In this sense, the TSG is an extension of CFG. A *Probabilistic Tree Substitution Grammar (PTSG)* assigns a probability to each production rule, and estimating it requires to learn statistics for linguistic structures from a corpus; parsing involves finding the most probable combination of trees for a given string.

Similarly, we assign a probability to each possible subgraph in the ontology, which can consist of one node only, and then we find the most probable

combination of subgraphs for a given percept to be expressed by a text. Once the subgraphs are inferred, the percept is acquired and new knowledge is discovered. This model allows generating the subgraphs. Different PYP are organized hierarchically, so to allow the combination of subgraphs.

We do not need sufficient statistics for computing the probabilities of production rules as we refer to the *selectional preference strength* [65] that allows us to define a probabilistic measure for establishing how a set of nodes fall in support of the PYP distribution. However, a training step is required for computing the related hyperparameters. Details of the whole method are explained in Chapter 4.

Chapter 3

Implementation Aspects of the Cognitive Architecture

3.1 Introduction

As stated in the previous Chapters, a human-robot team has to cooperate to achieve a goal in a not fully known environment. Robots and humans mates decompose the overall goal into subgoals and they choose the actions needed to reach the goal. They also match their skills with the correct steps to perform, and possibly delegate tasks to the other teammates.

A scenario concerning autonomous cooperation requires a complex software system with runtime adaptation to new situations that may leads to new requirements and constraints. Software injected and evaluated at runtime cannot be defined during design phase, and then the control system of the robot has to be handled as a self-adaptive system.

In brief, the self-adaptive system should be aware of its goals; it should be able to monitor the working environment, to understand how far it is from the goal and if it is deviating from the same goal. It should be also able to adopt alternative plans, and generate new plans when necessary.

From the point of view of software implementation, the challenges in this field concern knowledge representation and updating; selection and creation of plans at runtime; generation of techniques for purposefully and efficiently conveying the (runtime) decision process. These challenges lead to different solutions depending on whether we consider the architectural level or system level.

In the previous Chapter we considered the architectural level, while this Chapter focuses on the system level counterpart. In particular, we take

into account the BDI agents paradigm [37] and Jason the an agent-oriented language [13][11].

Decision processes elaborate data coming from external sources and the environment. In many domains it would be hard to design and implement the decision process merely by employing the monitoring, analyzing, planning, acting (MAPE) cycle. In our system, the decision process must take as input all the internal states of the agents involved in the environment, including human mates. Internal states then embody all the changes occurring at runtime.

The project we discuss aims at considering, as a crucial part of the decision process, the robot capabilities of attributing mental states (beliefs, desires, emotions, knowledge, abilities) to itself and the other mates. In brief, we take into account simplified forms of robot self-consciousness and theory of mind.

Thus, we discuss the steps of the ongoing work aiming at integrating self-consciousness and theory of mind capabilities in an architectural structure implementing adaptive decision process at the system level. The architectural part extends the MAPE cycle [4] with modules allowing the perception of the external world and the inner world as internal states. We structured the architectural part so to fill the gap at the system level. We then present an extended version of the Jason reasoning cycle to map the architectural level into an agent-oriented framework.

3.2 Towards using BDI Agents and Jason for Implementing Human-Agent Interaction

Jason is an implementation of AgentSpeak language [63][13] that allows overcoming the denotation of software, as it is no longer something providing a service by means of coding based on the intervention of the user. In Jason's logic, a computer program has its own know-how and and it is able to choose actions to pursue a goal on behalf of the user, without intervention. Then, a Jason program is an agent. The basic idea behind Jason is the definition of the know-how in the form of a set of plans: the Jason platform allows executing the deliberation process of a BDI agent by choosing the intentions to pursue within a set of possible states of affairs.

Typically, a Jason agent has partial control over the environment as it is populated by other agents having control over their own parts of the environment. The procedures for handling agent-agent interaction is standardized and defined at design time. Human-agent interactions are an open

problem in the context of cooperation between humans and agents, which presupposes delegations and selection of actions to be undertaken.

Human-agent interactions can vary from simple situations where everything is identified and defined at design time (environment, plans, actions and changing situations) to more complex ones where changes occur at any time and where the agent has to decide autonomously and to self-adapt to changing situations.

To gain the case we are facing, let us suppose the following three scenarios: in the first case, a team composed of a human mate and a robot works together to carry out a task known to both. Let us suppose that the working environment is known in advance. At runtime, there are no changes other than those resulting from the actions of the robot or the human mate. However, these changes have been planned in advance. In this situation, the agent acts in complete autonomy, and the goals may be achieved performing the actions in the agent repertory. Here, the collaboration is only apparent in the sense that the agent and the human mate do not need mutual help; then, the BDI logic and its implementation using Jason is efficient and usable.

In a second more complex case, let us suppose the agent needs collaboration by the human mate to perform part of the overall goal. For instance, it may realize not to be able to do an action because of some limitations (e.g., its arms are too short), even though having the correct know-how for completing the action. This situation implies an intervention of the human mate under an explicit request of the agent, and it is then a collaborative work. This case requires a soft self-adaptation: the agent has self-consciousness capability to understand he cannot select an action to achieve a goal. This case can be handled by the Jason interpreter by customizing the methods of some predefined classes (see [13] for more details).

In the third case, the most complicated one, let us suppose that only part of the environment is known beforehand. The common goal, as well as a set of plans to achieve it, is identified at design time, but the interaction of the agent with the environment and with the human mate allow the operating conditions to change unpredictably. This fact happens when the interactions with the environment brings out new terms of operability that must be considered so to choose the action to take.

Generally, when a team is made up of human mates only, they choose actions starting from their experience, the knowledge they have of the other team members, the trust they place in the other team, their emotional state and the anticipation of the actions of other mates. For example, suppose that two people are caring for a disabled patient, where routine care includes

the administration of medicines, cleaning, help during meals, and each of the two people has tasks assigned. If during a meal the patient spills a glass of water, then the operation involves picking up the glass from the ground and cleaning the patient, but neither of the two actions is assigned to a specific person. When a mate takes the initiative and clean the patient, then the other mate chooses to pick up the glass. If there is no procedure to respond to an emergency, the two mates generally do not stay still but choose what to do based on their experience and their internal state. Besides, they will collaborate even delegating to each other what to do.

Replicating this behavior in a human-robot team is a problematic task mainly because we do not have the tools to analyze and identify the possible elements perturbing and changing the environment, so we cannot determine, at design time, a suitable decision-making process to be efficiently implemented at the system level.

In the literature, promising approaches [7][10] solve this problem by shifting the design time to runtime. Also, architectures containing modules for learning and memory have been introduced to pass the decision-making process through the stored and processed sensing data [73][47][35]. However, these approaches do not take into account the robot capabilities of self-consciousness, which is the primary element in our hypothesis to create human-agent interaction systems behaving as human-human systems.

3.3 Extending Jason Interpreter and its Classes

In the first part of our study, we identified an architecture focused on the MAPE cycle. Here, specific modules allow the decision-making process to be triggered from the stimuli coming from the environment, from the internal state of agents and from the observation and interpretation of the actions carried out by the other agents in the environment.

Fig. 6.3 shows the high-level view of the modules of the cognitive architecture introduced in the previous Chapter. The modules centered on the sensing/plan/action cycle are highlighted in red. The core of the decision-making process consists of the reasoning module, the action selection module and the anticipation module.

This module is devoted to generating the current situation. Each time an agent has a goal to reach, the module selects a suitable action and it generates anticipations of the state of the world resulting from that action. The module receives as input the motivations, the goals and the elements in the memory, it processes them and it chooses and executes the corresponding

action.

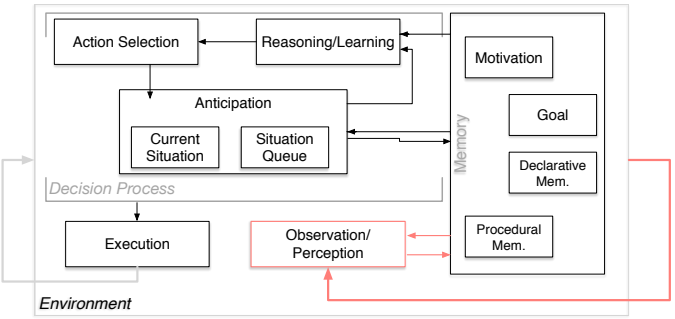


Figure 3.1: The Architecture Level for Human-Agent Interaction Systems.

The motivation module is a trigger for the anticipation and action selection. Here, the information of the robot inner state resides. This module generates decisions about the actions to be conveyed by means of the ability to attribute mental states (belief, desire, intention, knowledge, capabilities) to itself as a simple form of self-consciousness, and others agents as a simple form of a theory of minds.

The architecture has been mapped onto a software system by extending the Jason reasoning cycle. The reasoning cycle of Jason (see Fig. 3.2) is the counterpart of the BDI deliberation and means-ends reasoning process. The rectangles are the components determining the agent state; rounded boxes, diamonds, and circles describe the functions used in the reasoning cycle. In particular, the circles model the application processes, and the diamonds represent the selection functions.

The cycle is divided into ten steps, starting with the perception of the environment to the selection of actions to be taken. The main steps of the reasoning cycle concern the update of the belief base, the management of the events corresponding to the changes in the environment and with respect to the goals, the retrieval of plans from the library, the unification of the events with the plans available to select the most useful plan (the so-called the applicable plan), and the selection of the intentions.

Perception and actions in the environment are implemented by the functions *perceive*, *checkMail*, *act* and *sendMsg* (see [13] for details). The cycle starts by updating the belief base and generating an event through the Belief Update Function (BUF) and Belief Revision Function (BRF); these functions correspond to the *buf* and *brf* methods (see Fig. 3.4). The *brf* takes the agent's current beliefs and percepts and it suitable adds or removes beliefs. An event is then selected by the *event selection function* S_E; events corresponds to the perceived changes in the environment and the agent's

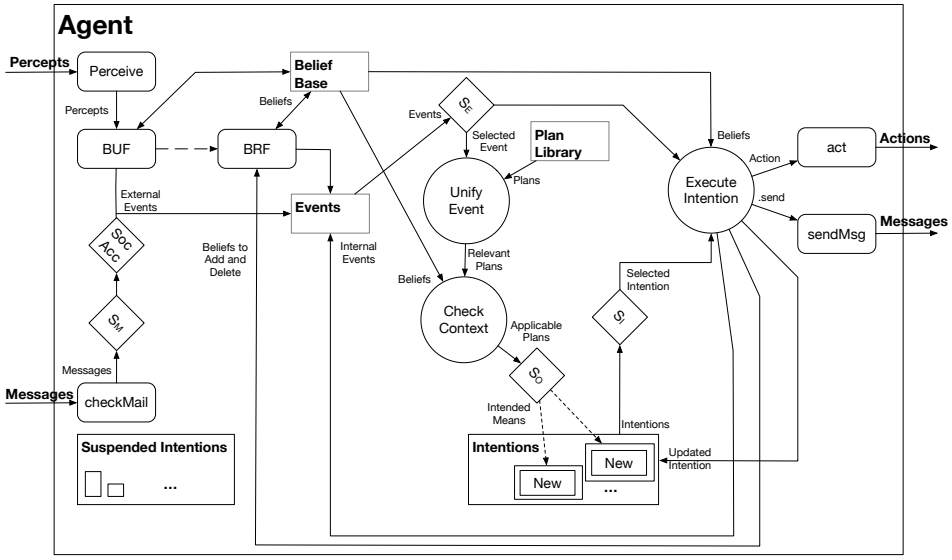


Figure 3.2: Jason agent reasoning cycle. Redrawn from [13]

goals.

The selected event is then unified with the trigger event to individuate the set of plans relevant for the event. Once the relevant plans have been identified, they are checked against the context (a set of belief literals representing the condition for the plan to be activated) to verify whether they are logical consequences of beliefs. The result is a set of applicable plans.

Given the agent's know-how expressed by the plan library and the information about the environment in the belief base, the *option selection* function S_O chooses a plan handling the event and includes it in the set of intentions. The intentions component contains all the intentions ready for the execution. The agent chooses the intention to be executed by the *intention selection* function S_I. The selected intention is then executed.

We exercised the robustness and stability of the Jason interpreter for implementing the BDI agents by extending the reasoning cycle to introduce the modules of the architecture (Fig. 6.3). Figures 3.3 and 3.4 illustrate the added components in the reasoning cycle for the new decision process (Fig. 6.3), and the classes we extended and inserted in the user-defined components.

Mainly, we introduced components and functions (in blue in the Fig.) related to the learning/reasoning module and a process implementing the introduced anticipation module. We added the motivation base including

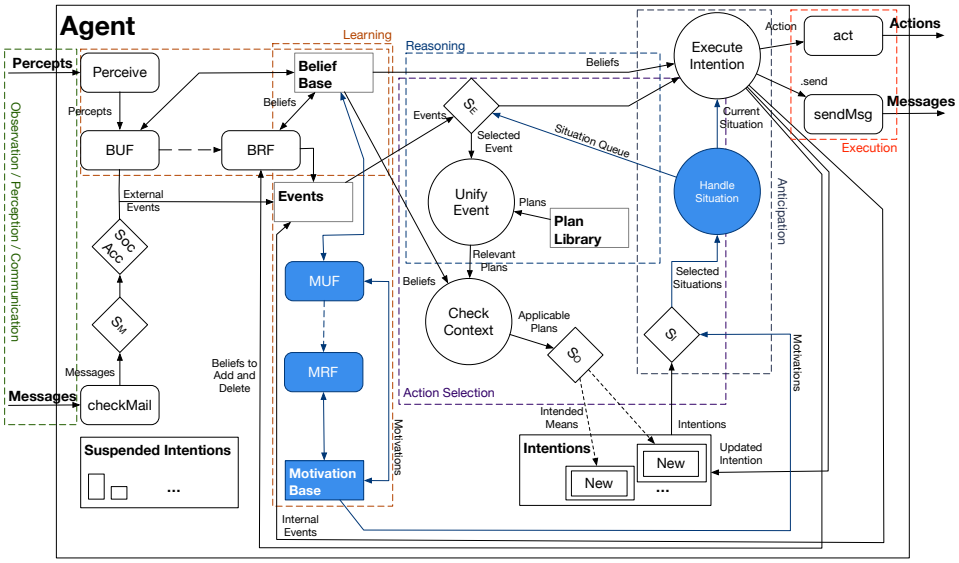


Figure 3.3: Extended Jason reasoning cycle.

all the beliefs related to the mental states and emotions. We consider the motivations as extensions of the belief to include the beliefs in oneself and others. The beliefs are thus related to the external world outside while motivations refer to the inner states, i.e., to the robot self-consciousness.

We added a Motivation Update Function (MUF) and a Motivation Revision Function (MRF). At the beginning of each cycle, the MUF updates and initializes the agent's motivations and the belief base, by taking as input a list of literals with beliefs and motivations (see Figs. 3.3 and 3.4). The input from the belief base is treated as it were from the perception. The motivations are elaborated from the modified S_I function. It generates a list of situations to choose the one to be executed and the queue to be used for the selection of events through the S_E function. A situation is similar to the state of affairs concerning the environment: it represents the overall state of the agent including the agent inner states. In this way, we let agents reason on new events generated from internal states.

Finally, the process *Handle Situation* generates the current situation to be executed and it provides the queue of situations to the S_E function.

Concerning the agent code (Fig. 3.4), we added a class as an extension of the *BeliefBase* class named *Motivation*. The *Motivation* class allows managing resources as the *BeliefBase* and it also queries external services to let the agent be aware of its internal state.

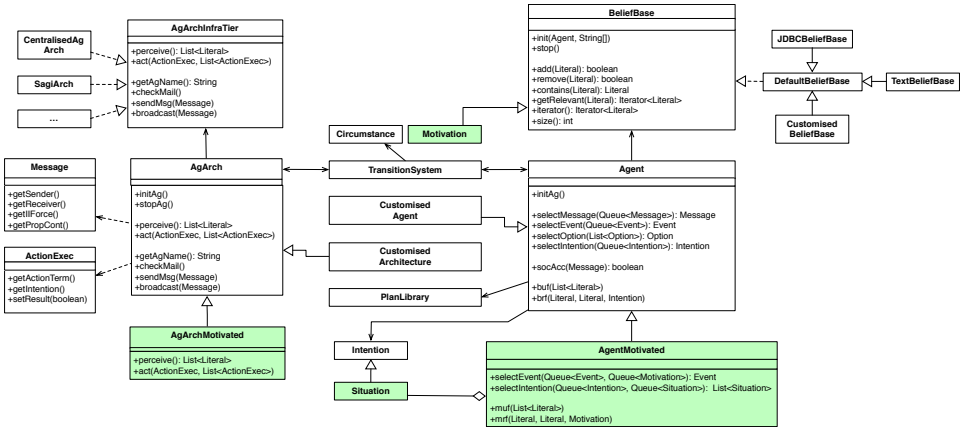


Figure 3.4: Agent and Agent Architecture Class Diagram and the related extension for implementing the reasoning cycle.

The core of the proposed reasoning cycle is the *AgentMotivated* class that extends the *Agent* class. The *selectEvent* and *selectIntention* S_E and S_I functions supports the code related to the MRF and MUF functions (Fig. 3.3) by means of *mrf* and *muf* methods. The agent invokes these methods to modify the Motivation Base. Moreover, the extension of *AgArch* class into *AgArchMotivated* implements the perception and action modules.

Finally, Fig. 3.4 describes the general classes architecture.

3.4 Discussion

In this Chapter, we presented the implementation of the agent's decision making process in a dynamic context. Our proposal is based on the fact that agent's decision-making-process is determined by processing data coming from observation of the external environment and by the knowledge that the agent has about itself and the other agents. The implementation of such a system is a hard task because its features can be considered only at runtime, during the interaction with the environment. Therefore, the system must be treated and implemented by means of self-adaptive characteristics.

We have exploited the BDI agents and the Jason language, which allow creating agents that perform a deliberation and means-ends reasoning process. We modified the Jason reasoning cycle to include modules to manage events, plans, and intentions selection to take into account the motivations in addition to traditional beliefs. To complete the infrastructure the agent

coding level, we modified *user-defined* classes of the Jason component. In particular, we added the classes needed to implement the new reasoning cycle by adding the methods necessary for the agent to be able to choose the plan to pursue using a cognitive process based on motivations that embody the mental states of the agent.

It is worth to note that the proposed cycle extension does not alter the original Jason agent reasoning at a high level, but it extends its capabilities, allowing the development of agents able to manage at the same time the agent self-consciousness and the theory of mind together with the usual decision-making process.

Chapter 4

Knowledge Acquisition in the Cognitive Architecture

4.1 The Cognitive Architecture for Human-Robot Teaming Interaction

Our aim is developing a cognitive architecture that includes the necessary modules for a robot to cooperate in a team to achieve a common goal. The robot has to apply a decision-making process that takes its cue from the objective situation of the environment and also from the knowledge it has of itself and the other members of the team.

The architecture in Fig. 4.1 contains modules for self-consciousness, for representing the surrounding physical world, including the other agents and including the mental states that this involves. In a cognitive agent, it is the mental state that triggers actions. The memory is the support for the inner state processing phase.

To date, architectures base their decision making and learning processes on the concept of stored data or facts and not on the idea of a mental state. Our contribution lies in the creation of memory modules containing the information about the mental state in the world so that the perceive-act cycle becomes what we call the perceive-proact cycle. We identified some main modules: the module devoted to the observation of the environment, the one realizing the decision process (including reasoning, learning and actions anticipation), the execution and the memory. A cognitive agent knows its goal and the state of affairs around and within itself, it perceives objects relevant to the mission in order to trigger a decision about which action to perform. Before performing actions, it produces the anticipation of

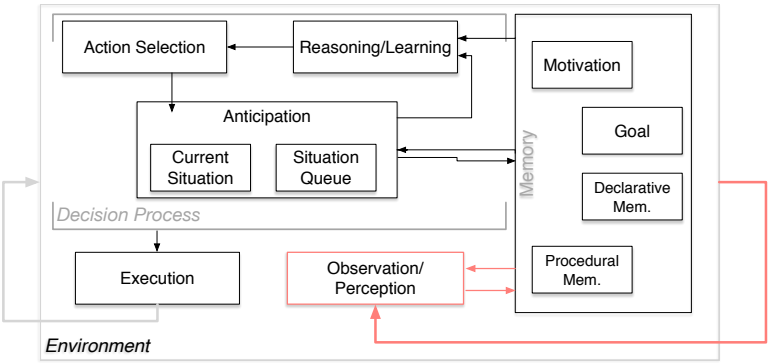


Figure 4.1: The Cognitive Architecture for Human-Robot Teaming Interaction

the action results to check if that action brings to an acceptable situation of the world, and alternatively it generates a queue of situations to be selected if necessary. This process is performed iteratively, each time interacting with the elements in the environment through perception and observation.

In this Chapter, we detail the path highlighted in red that is related to the process of knowledge management and its acquisition through introspection.

4.2 Problem description: the example of a robot working in a partially known environment

The project aims to make a robot *aware* of objects in dynamic environments and *self-aware* of its knowledge by updating it when a new entity is perceived. If the robot does not recognize an object, then it would not be able to use it and refer to it during task execution, thus breaking the collaboration in the team.

A means for knowledge representation is then necessary. Currently, ontology is one of the possible strategies for equipping the knowledge level of cognitive agents. An ontology is more than a simple taxonomy as it allows representing semantics relations beyond the *is-a* subsumption.

Formally, the ontology is a set of concepts, individuals, and roles. The concepts represent abstract entities, and are the symbolic representation of the knowledge; the individuals are instances of concepts and represent concrete entities in the environment. The roles are properties, that can be relational or datatype; the former define abstract relationships among

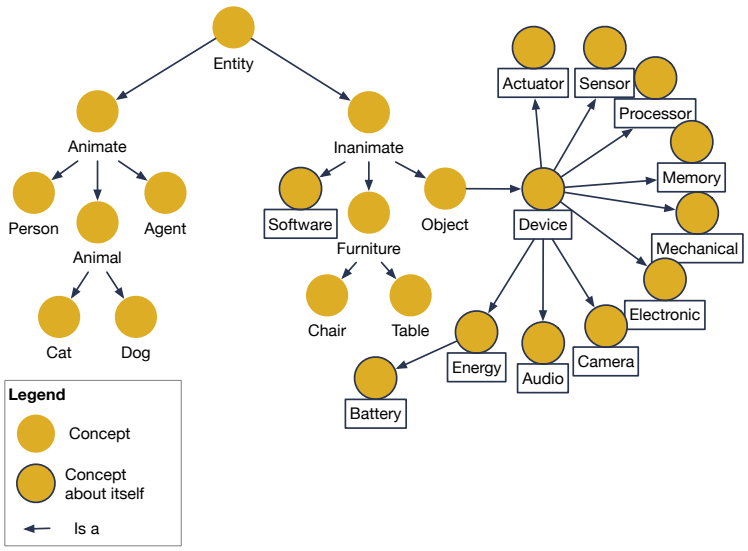


Figure 4.2: The fragment of the ontology including all the concepts about itself. These concepts are framed.

concepts, the latter define properties of a concept with datatype values.

Figure 5.4 is a high-level ontology which includes the concepts (without the correspondent instances) organized in taxonomy; it is the set of concepts the robot possesses. Every time a robot perceives or observes a concept already known, then it creates an instance of that object.

The ontology is generally manually encoded before the robot is deployed. If the robot is situated in a dynamic environment, then a dynamic ontology is expected to grow when it perceives new objects. Thus, the challenge is how to model and represent new knowledge acquired at run-time in an ontology.

Let us consider the simple scenario where a robot is plunged into an environment whose high-level knowledge is represented in Fig. 5.4. Actually, it is an excerpt of the whole ontology, and for the sake of clarity of the example we highlighted the concepts related to elements *about itself* owned by the robot. These elements concern the physical components of the robot that it perceives as integral parts of the environment. In Fig. 4.3, the same ontology is enriched with some concepts the robot has perceived and then instantiated; we may say that it knows these concepts and it recognizes one or more instances of them in the environment.

Let us also consider a self-repair operation as the goal of the robot.

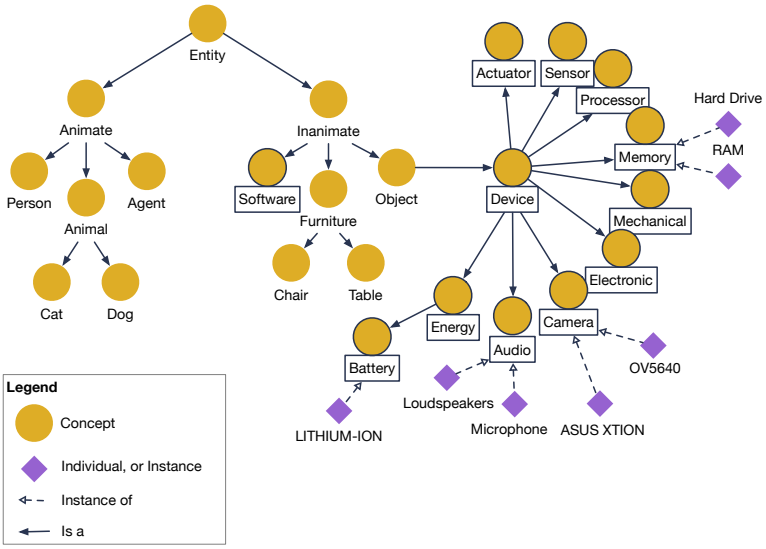


Figure 4.3: The fragment of the ontology including all the concepts about the environment along with some instances.

Physical self-repair operations are critical in applications where no humans are around to assist the robots. Also, it can be useful for cooperating with humans to repair other machines.

Knowing the damaged resource is the first step for self-repairing operations; the robot acquires such a resource and becomes able to understand how to repair it, for example by identifying what object can be used in place to it, or the set of necessary actions for replacing it.

Generally, the robots can self-diagnose and detect if and what component is in trouble. The primary goal of diagnosis is to check every device and its functionalities and to publish whether the device has or has not a fault: until this moment, the robot has no consciousness about this device. It passively communicates to the human the internal state, but it is not able to understand such a state.

To start the self-repair operation, the robot has to conceptualize the device for the timely intervention. In other words, by self-diagnosis, the robot knows that a device is in trouble, but it does not know such a resource, which remains an abstract concept until it is not acquired in its knowledge base.

The robot might not know anything in advance about its own devices, or it could have a partial knowledge about itself.

In our example, we employ a Pepper Robot[1] and we forced it to diagnose a hardware trouble by the self-diagnose library (the `ALDiagnosis`[2]), that returns the variable d as the name of a possible damaged resource. Let suppose that the CPU is this resource, so we declared $d = cpu$.

As it can be seen from Figure 4.3, the CPU is an element not known to the robot. The robot can process the new perceived element, to link it to a known one to produce a new instance (see Figure 4.4). It is worth to note that we do not care if a knowledge element belongs to the external environment or the internal one; in our approach, we use an extensive conceptualization of the environment, including the robot itself.

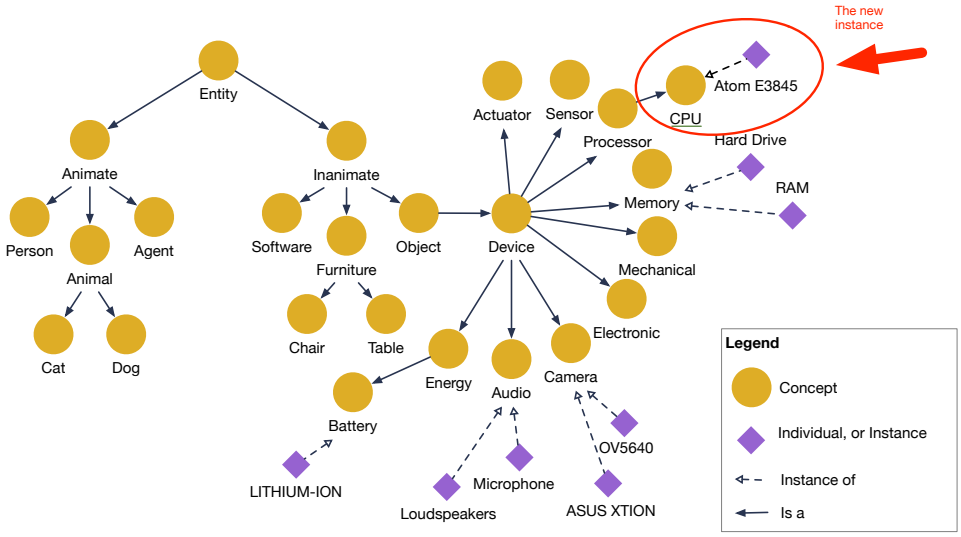


Figure 4.4: The knowledge acquisition related to the *CPU* concept.

When a new entity is perceived, two different cases can be considered:

1. the abstract concept is already modeled in the knowledge base; a new instance for that concept has to be created corresponding to the perceived entity. We will refer to this process as the *mapping process*;

2. the abstract concept is not modeled in the knowledge base; the concept and the instance have to be created. The concept becomes persistent. We will refer to this process as the *merging process*. This is the case of the previous example about the CPU resource.

[1] https://www.softbankrobotics.com/emea/en/robots/pepper/find-out-more-about-pepper

[2] http://doc.aldebaran.com/2-5/naoqi/diagnosis/aldiagnosis.html

In both cases, the problem is related to infer the presence or absence of the concept in the knowledge and then to identify the correct allocation of a new concept and instance in the ontology, leading to the robot self-consciousness.

4.3 Modeling the mapping and merging processes

In this example, the percept is the name of the damaged resource as output of the cited library. In other cases, the surface form of a percept is a descriptive label provided to the robot by the human mate which articulates such a label producing a streaming voice. The `ALSpeechRecognition` library[3] of the Pepper robot allows to transform such a stream to the correspondent string word among those ones in a dictionary defined for testing the method. Future refinements may regard the definition of a set of services for using external speech recognition and visual detection libraries.

Once the string corresponding to a vocal stream is detected, the *mapping process* allows to infer if such an entity is already modeled in the knowledge base, and in this case the percept is instantiated. Otherwise, the concept has to be acquired and correctly allocated in the ontology by the merging process.

For the purpose, the surface form of the ontological label and the surface form of the percept have to be mapped; we refer to the similarity measure defined in [60] that computes the closeness between the ontological labels and an external word. This measure keeps in account a syntactic component (the syntax is a crucial aspect for discriminating the equivalence of two textual elements). Furthermore, a semantic contribution is considered too to disambiguate the word.

The employed measure represents a similarity distance between the words w_1 and w_2; it is the weighted sum of the Jaro-Winkler distance [77] and the Wu-Palmer distance [80]:

$$sim(w_1,\ w_2) = \delta * jaro(w_1,\ w_2) + \gamma * wup(w_1,\ w_2). \tag{4.1}$$

The motivation to consider the Jaro-Winkler distance is the characteristic of the strings to compare, that are typically short words as the labels of the ontology. The Wu-Palmer is considered one of the best semantic measure in literature.

[3] http://doc.aldebaran.com/2-5/naoqi/audio/alspeechrecognition.html

The experiments show that the weight to the syntactic contribution leads to a better identification of the concept in the ontology, and so the parameters are empirical set with the following values: $\delta = 0.7$ and $\gamma = 0.3$.

Given the set of ontological labels O and the percept p, the mapping process is modeled by the *map* function, that is:

$$map(p) = \begin{cases} o & if \quad max_o > \tau \\ mer(p) & otherwise \end{cases}$$

where the *mer* function starts the merging process next defined, while the max_o is the maximum value of the set S_p where $S_p = \{sim(p, o) \mid o \in O\}$.

The *map* function returns the concept o in the ontology if o matches with the percept p according to the similarity value which has to be higher than the threshold value τ empirically set to 0.9; it means that the percept p is similar to o, so it already exists in the knowledge, and p becomes an instance of o.

The *merging process* starts when the max_o value is less than τ; to merge a new concept in the ontology requires to compute its correct allocation. For this purpose, we investigate the probabilistic approach proposed for the *Probabilistic Tree Substitution Grammar (PTSG) induction* [27]; such an approach defines a power-law distribution over a space of production rules that combine the grammatical, linguistic structures.

To estimate the probability of each production rule, the statistics for the linguistic structures they represent has to be learned from text corpora. Parsing a string by using a PTSG means to find the most probable combination of rules for the given string. A Pitman-Yor Process (PYP) [61] is used for this purpose.

Our idea is that the ontology of the robot is a set of ontological structures, that are the nodes and the triples representing properties and relations. We define a PYP for estimating the correct linking between these structures and a given percept in the same way of the previous string parsing.

In facts, we do not need sufficient statistics for computing such probabilities, because we consider the linguistic properties of the text representing the percept in respect to the labels of the ontology.

The PYP process assigns a probability to each ontological structure, and then it finds the most probable combination of these structures for a given percept. Formally, the merging process is modeled by the PYP distribution $mer(p)$ over the ontological structures, that is:

$$mer(p) \propto PYP(\alpha, \beta, G_o) \tag{4.2}$$

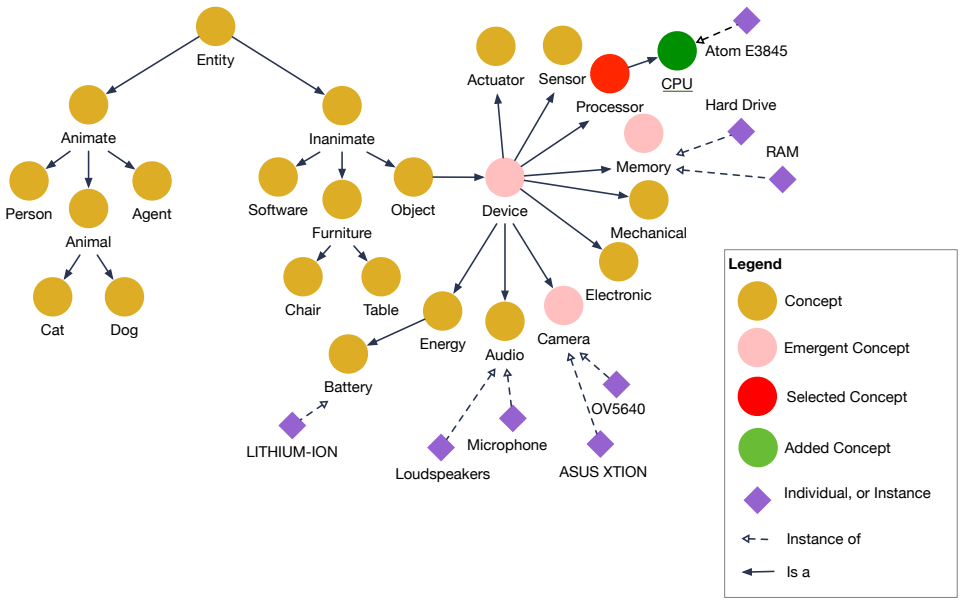


Figure 4.5: The knowledge acquisition related to the *CPU* concept with its instance represented by the diamond shape. The emergent concepts are highlighted. The more probable concept is the candidate parent and it is in red.

where α and β are the hyperparameters of the process that influence the shape of the distribution, while G_o is the base distribution which determines which fragment trees will fall in the support of $mer(p)$.

In particular, G_o is a function that, similarly to $map(p)$, involves the symbolic linguistic properties of p over the space $o \in O$, and allows to choose the more plausibly fragment tree given p. In this sense, the method is hybrid, involving a sub-symbolic process integrated with symbolic properties.

In the self-repair example we illustrated, the mapping process *map* will invoke the *merg* function because the similarity measure computed by 5.9 is under the threshold for each concept in the ontology.

The Table 5.1 shows the results of the merging process. The concept *Processor* is the more probable than the other structures in the ontology; a new concept *CPU* is created as children of *Processor*, with the correspondent instance. Since this moment, the robot has conceptualized the resource, and it can refer to the knowledge about the processor for the self-repairing operation it or for identifying the possible new processor among a set of available spare parts.

Ontological Structure	*Result*
[Processor]	**0.04662**
[Camera]	0.03195
[Memory]	0.01704
[Device, Processor]	0.01251

Table 4.1: PYP results for the CPU resource.

Concept	**Ontological Structure - Result**		
Kiwi	[Fruit] - 0.04250	[Fruit, Pear] - 0.0295	[Fruit, Apple] - 0.02318
Watermelon	[Fruit] - 0.04685	[Fruit, Apple] - 0.02563	[Fruit, Pear] - 0.01664
Voltage	[Energy] - 0.08136	[Device, Energy] - 0.02436	[] - -
Linux	[Software] - 0.13556	[Object, Software] - 0.04744	[] - -
Frog	[Animal] - 0.03827	[Animal, Fish] - 0.01140	[Animal, Cat] - 0.00913
Cat	[Cat] - 1.0	[] - -	[] - -
Sonar	[Sensor] - 0.04739	[Device, Sensor] - 0.03176	[Device, Memory] - 0.02247
Pencil	[Pencil] - 1.0	[] - -	[] - -

Table 4.2: Some of the results obtained during the experiments for testing the method.

New knowledge is discovered, and the ontology is updated as drawn in Figure 4.5. The same figure shows the *emergent concepts* for the resource, which are in the structure which falls in the PYP support; it is to notice that these concepts are semantically similar to the percept. Hence, the base distribution defines a support by excluding concepts that have an entirely different meaning.

To demonstrate the robustness of the proposed approach, we report more experimental results at Table 4.2. These results were validated by domain experts; each row represents the concept to acquire and the ontological structures outputted by the mapping and the merging processes. For each ontological structure, the cell contains the label and the correspondent result. The structures are ordered according to the results. The concepts are from different domains for demonstrating the robustness and the generality of the proposed approach. In the case the concept is already in the knowledge base, only a structure emerges that is the node in the ontology corresponding to such a concept, for which only the instance has to be created (for example, the *Cat* and *Pencil* concepts in the table). The cells with empty square brackets and dashes mean not detected structures.

4.4 Discussion and Conclusions

In this section, we discuss some theoretical implications of the strategy we propose. We demonstrate how some theories are in support of our implemen-

tation choice, and in particular how the actual problems of the transparency and transfer learning are addressed too.

4.4.1 Self-consciousness about knowledge state

The set of facts the robot owns about the world and that are available for reasoning are formally represented by *sentences* [51]. The semantic of these sentences is the knowledge and can be expressed by different formalism, including ontology. The sentences constitute the robot *consciousness* about the environment. So we can claim that the ontology is a form of robotic environmental awareness because it represents a set of sentences available for reasoning.

Facts modeled in the ontology are available for observation which generates other sentences about the whole facts; the facts constitute the *awareness* about the world. When the results of observation are not generated by unconscious processes but by specific mental actions, then they come from a kind of artificial *introspection*, that is, the robot *self-consciousness*.

We argue that to infer if the concept is already in the knowledge base or not (i.e., sentences already express it) is the result of a robot self-consciousness: it produces facts about the state of the knowledge. The robot introspection might look like "Does my knowledge include this perceived entity?" (i.e. "Do I know such entity?"); "Where this unknown concept should be allocated in the ontology?" (i.e. "What is this entity?").

The mapping and the merging processes previously discussed can be considered as the mental actions that make the robot able to introspect about its own knowledge base. An important aspect is that such observation is transparent: the results it produces are explainable and justifiable.

4.4.2 The cognitive semantics for modeling introspection

The cognitive semantics theory [2] claims that the meanings the humans understand are carried by structures in their mind that are of the same nature as those that are created when they *perceive* something (i.e., when they hear, touch, see, manipulate entities in the context).

Briefly, the meanings are located in human heads, and they are not found in the external world; so, when we eat a pizza, we see it as a pizza since the perception we have fits with the cognitive structure in our head that is the *concept of pizza*. In our mental classification, there will be a *schema* about how a concept looks like, and we can infer another kind of information, as

what pizza we are eating (its name, ingredients and so on). This schema is the very meaning of the entities according to cognitive semantics.

A consequence of the cognitivist position is that the semantics for an entity is seen as a mapping from the surface form of the entity to the surface form of the cognitive structure.

In our approach, the ontology constitutes the set of mental schemas for the robot; in particular, the high-level includes the cognitive structures for the semantic mapping. The ontological labels provide the surface forms to link to the surface form of a percept.

When the percept does not fit to any cognitive structures, the robot understands that it does not know such entity. The robot hence learns that new concept and allocates it at the high level.

We consider the emergence of cognitive structures related to a percept the mental actions the robot has to perform to infer if it knows the concept or not, and eventually to conceptualize it. The robot will use the same ontological mental schemas, leading to the robot self-consciousness.

4.4.3 Explanation and transparency

The literature related to self-consciousness is vast, and many approaches were defined as a classification problem by neural networks. Some of this most accurate classifiers do not provide any mechanism to explain how they output each result; their reasoning mechanisms are not transparent, not respecting the actual trend to understand the underlying decision processes.

Understanding the learning model of a neural network has become fundamental; to give transparency to the predictions and decisions of an algorithm is necessary to consider its reliability. According to [57], transparency is the network's ability to explain its reasoning mechanisms; it seems to be the result of a form of introspection that we prosecute considering the described robot scenario.

The training processes at the basis of sub-symbolic methods represent the hidden behaviors. The emergence of latent structures from a dataset allows tuning the parameters of the models. Typically, these techniques lose the *granularity* of the data, that is important when the processes have to be monitored. The works in [56] [57] propose neural networks that attempt to overcome this problem.

The discussed strategy is an alternative approach that integrates symbolic and sub-symbolic methods; in this way, the training process is avoided leading to transparency.

4.4.4 The importance of the incremental knowledge acquisition

The reuse of knowledge learned from the typical training phase of the machine learning models is the objective of the *transfer learning*. With transfer learning, the training phase starts from patterns that have been previously learned for a different task. Instead of starting the training process from an opportunely annotated dataset, it starts from patterns that have been learned by a different machine learning model to solve a different task.

Even if the transfer of knowledge and patterns is ideally possible in a full kind of domains, to realize it remains a challenge: knowledge transfer is possible when it is appropriate, and it involves trust to the way the involved patterns were generated. A validation of patterns and context is required. Moreover, not all advantages and disadvantages are known at this time.

With our method, we have not such a problem. The knowledge is incrementally acquired, and the method does not depend on the training dataset. We apply the strategy in each context in which the knowledge acquisition is required, and it is independent of the domain under investigation. In other words, the ontology can regard different domains without conceptual limitations.

This is a central contribution as we consider that the actual trend is to incentive data scientists to experiment with transfer learning in machine learning projects and to make them aware of the limitations of this method.

Chapter 5

Incremental Knowledge Acquisition

5.1 Introduction

A robot designed to collaborate with human beings in a team has to be aware of the shared space. The structure and the objects of the environment in which the team operates have to be known by the robot for enabling decisional, planning and interactive skills [19]. During collaboration, such a structure changes and the robot has to update its knowledge about the new environmental state. The run-time alignment to the context is fundamental.

The more general principles pointed out by the *Adaptative Resonance Theory* (ART Theory) [38] were adapted for knowledge acquisition [50]: an artificial agent becomes a 'truly' intelligent system when (a) it is able to support incrementally knowledge acquisition in a way it has not to be retrained, (b) it supports the inductive and deductive reasoning for reaching a goal, and, finally (c) it is able to focus on what is relevant knowledge.

We do not claim to address here all these issues, but we focus on the first point of the ART Theory. In this paper, we refer to the term incremental knowledge acquisition as the automated process of abstracting knowledge from facts and other knowledge. It cannot considered a naive accumulation of what is being learned but it should be checked whether new learned knowledge may be acquired upon existing knowledge.

For a robot the knowledge acquisition process regards to link low-level knowledge, as perceptions and actions, to high-level one that is a net of concepts modeling general domain knowledge or common-sense knowledge [48]. Generally, high-level knowledges are modeled by ontologies [5] that are

"a formal and explicit specification of shared conceptualization" [39].

Usually, these ontologies are large and include all the possible concepts the robot could encounter in the world; the structure of the environment is acquired by processing the whole ontology and retrieving those concepts that can be linked to the perceptions. Current systems require manual pre-annotation tasks, leading to efforts for rules definitions on the domain representation [55],[40] .The existence of well-structured knowledge sources and memories is also important [36], [67].

The well-known *stability* and *plasticity* [18] constraints are typical problems for the incremental knowledge acquisition: the plasticity property concerns the capacity to learn new concepts, while the stability property is the capacity not to lose and corrupt previously learned ones. Without plasticity and stability, newly learned knowledge may be redundant, irrelevant or inconsistent concerning the previous ones.

In the human-robot teaming scenario, the relevant information managed by the team during a thread of cooperation are related to the specific task to solve and are limited to a context of the environment. These information are more frequently employed than the others, and the plasticity issue becomes less binding: the concepts not linked to the context of the task are less useful, i.e., not all the concepts in the environment should be acquired. We consider the *conditional plasticity* to take into account the relevant information close to the context. Stability keeps the same sense and regards the correct allocation of a new concept in the ontology, without altering existing facts.

We get stimulus by the Pitman-Yor Process (PYP) [61] defined for *Tree Substitution Grammar (TSG) induction* [27] which computes a power-law distribution over a space of production rules taking the form of a combination of *elementary trees*.

An elementary tree of a TSG is a tree of height ≥ 1 where each internal node is labeled as *nonterminal* and each leaf is labeled a *terminal* or a *nonterminal*. A TSG nonterminal can be rewritten by an entire other elementary tree, and in this case it is a *substitution site*. A *Probabilistic Tree Substitution Grammar (PTSG)* assigns a probability to each production rule and hence a probability to any elementary tree for rewriting a given substitution site. Estimating it requires the learning of statistics for such structures from a corpus. The PYP model embodies the *rich-get-richer* property, in which a few elementary trees will occur with high probability as is typical in natural language where a few grammatical expressions are widespread used only.

We claim that a similar model may also produce exciting implications

in the described scenario. In fact, the power-law distribution resembles those employed when linking concepts from an environmental context to a knowledge model: we assign a probability to each possible subgraph in the ontology, that is considered as an elementary tree of the previous case, and then we estimate the most probable combination of such subgraphs for a given perception expressed purposely by a sequence of words. Once the subgraphs are inferred, the perception is acquired and, if it there was not in the model, new knowledge is discovered. The rich-get-richer dynamic gives to the ontology's subgraphs related to the context an higher probability to be linked to the perception.

Differently than the case of grammar induction, which requires a big corpus for training the underlying model, we do not need statistics to compute the probabilities of each subgraph. This aspect is very important for the human-robot teaming scenario where training data are not available; the ontology of the robot may be small and incrementally grow during cooperation. However, a simulation of training was made to compute the typical hyperparameters of the process: such a simulation is a simple grid search-algorithm that applies the model many times over different domains, i.e. over different ontologies and perceptions, and evaluating the combination of PYP hyperparameters that produces the better results in the largely sets of domains.

We refer to the *selectional preference strength* [65] by Resnik for modeling the PYP base distribution which estimates how a set of subgraphs fall in the PYP support; being a form of linguistic entropy, our base distribution draws the subgraphs that better match to the features of the perception basing on semantic and syntactic similarities, and not on the statistical distribution of the subgraphs in a training dataset. This represents a novelty in the PYP model definition: the estimation of the probability value depends on a deterministic function and not on a probability function.

We compare our method with a set of classifiers for solving the organizational problem which arises when a robot has to organize a set of physical objects in different locations. By our strategy the robot groups the objects basing on their semantic nature, leading to the best classifier of all as the results show.

5.2 Related Works

Over the last decades, there has been an extensive research interest on knowledge acquisition, involving different research areas. It is considered

an important task for object recognition and classification [], task planning [], domain representation [], reasoning [], symbol grounding [] and so on.

The most widespread strategies attempt to discover new knowledge from patterns and rules by machine learning processes; for example, NELL [17] and ELLA [] are two approaches which concern the persistent and cumulative acquisition of knowledge by machine learning. However, in such cases, the basic principles specifying how to analyze the existing knowledge and how to acquire new items are not well formalized. When the system is a robot, the typical training phase of machine learning methods is not desirable because the environment in which the robot acts has very specific demands which are usually not met by the items in the dataset. Moreover, the environment could rapidly evolve making the dataset hold and requiring further training phases. All these aspects could lead to slow reaction time by robot, that severely compromise the performance during collaboration with humans.

Several approaches have investigated for incremental [20] or cumulative [21] knowledge acquisition by logic programming but the possibility to integrate them into robotic artifacts is not considered and represents an open challenge. Other works focus on the importance of the human cognition for improving knowledge acquisition, and among them the work at [] considers the notions of forgetting and memory consolidation as necessary cognitive components for knowledge acquisition. The authors look for a proper foundation for detailed knowledge assessment metrics and criteria for modeling memory and forgetting, not specifying the real improvements.

In the robotic field, the knowledge problem arises in the ontology processing; typical framework as KnowRob [75], OpenRobots [], PEIS K&R [] are aligned to Cyc [] which represents a consensus, and are large and attempt to include all possible concepts of teh world. Generally the problem to acquire knowledge becomes a *symbol grounding* problem, that attempt to anchor knowledge in the physical world. The knowledge does not grow in such a case but is processed by retrieval methods, leading to delay in the task execution. Many other approaches exist, like amodal (in the sense of modality-independent) proxies[54], grounded a-modal representations[55], semantic maps[56–58]or affordance-based planning.

The authors at [**?**], define a framework in which the knowledge available to the robot comes from three sources: a priori knowledge that is stored in an ontology and is loaded at start-up, and implementing the common-sense knowledge. The second part of the knowledge is acquired at run-time from perception, interaction and planning, and finally the third source of symbolic statements comes from the inferences produced by a reasoner. Also in this case, the knowledge acquisition is not meant as expansion but as

Even if in some cases [] this problem is solved by a unified robot knowledge framework that integrates different kinds of knowledge (sensory data, context and domain information, internal states, possible actions and so on), if a low-level data is not linkable to the high-level conceptual representation, then the data could be lost. New concepts at the high level have to be semantically identified and acquired too.

Ontology learning and dynamic instancing of concepts is proposed at [], where the outputted ontology is composed of fixed modules and is not a standard ontology usable in open domain.

The typical behavior-based architecture [] is widely referred as the base for implementing interactive behaviors of robotical systems when operating in a dynamic environment; such a systems have predesigned perception-action pairs, that allow robot to adapt its behavior to external events. If an event is not predefined, a manual intervention on the rules is required and it could be problematic.

We attempt to solve some of these limitations by proposing a probabilistic evaluations for linking concepts not only from high-level to the low one, but considering the emergence of new conceptual entities too. Being probabilistic, the method estimates the most plausibly collocation in the ontology, that can be next validated by interactions with the team. Many contemporary learning algorithms do experience catastrophic forgetting, particularly when they try to learn quickly in response to a changing world. These include the competitive learning, self-organizing map, back propagation, simulated annealing, neocognitron, support vector machine, regularization, and Bayesian models

5.3 Theoretical core idea

We formalize the knowledge representation and the surface form of perception to describe how a typical process of the TSG induction can be applied and adapted to the described scenario.

5.3.1 Formalizing knowledge and perception

Let consider the terminological ontology O and the assertional ontology I both representing the knowledge for the robot. O will include the general concepts of the domain, that are the *classes* with their *properties* and *relations* between them. I will contain the *instances* of the classes, that represent the assertions, and that are the concrete objects in the environment. O is the tuple $O = (C, P_o, P_d)$ where C is the set of classes, P_o is the

set of relations between classes in C, and P_d is the set of properties of each concept in C, that are relations whose domain is C and range is the typical set of datatype values. I contains the instances of C.

According to the general definition of ontology, the elements in C are organized in a taxonomy, i.e., a hierarchical tree where only *is-a* relation links nodes; a node is a class in C and `THING` is the broader concept that subsumes all the others. A child is a kind of parent. An example of a fragment of taxonomy is shown in figure 5.1. In such a representation, not only subsumption relations are represented, but the figure shows an *enriched taxonomy* including some properties definition. In this case, the representation is a graph and not a tree. The internal nodes are classes in C (in the capital text), and they are linked by not-oriented lines, i.e., the is-a relations. The leaves can be instances (represented by not-capital text) or classes.

The relations in P_o and properties in P_d are represented by oriented dashed arrows that link suitable nodes. For example, the *has_color* object property links the concept `APPLE` to the concept `COLOR` and it defines the color of a perceived apple. In the example, the instance `apple` has not a solid color, but it could be defined because such property is defined for its class and it is valid for all the instances of that class. The relations or the properties of classes could not be instantiated for specific class's instances, as in the proposed example. Instead, the *has_shape* property is defined for the specific instance of `APPLE` class.

The *space of subgraphs* S_G from the ontology contains all the possible fragment of the enriched taxonomy, while the *space of fragment trees* S_T contains all the possible fragment of taxonomy with the only is-a relations and without other properties.

A percept $\mathbf{p}$ is the symbolic form of a perceived entity that could be, e.g., a voice stream, an object, one or more features of an object. We suppose that the percept is represented by a suitable text which describes the perceived entity (i.e., the text produced by the speech recognition, the text describing the object or its features, and so on). The percept $\mathbf{p}$ is represented as a sequence of tokens that are the words in the symbolic description without the stop-words (conjunctions, articles, adverbs, and so on). So $\mathbf{p} = \{p_1,\ p_2, ...,\ p_n\}$ being p_i the ith token of the percept. The percept $\mathbf{p}$ is comparable to the string to parse in the grammatical case. Each token of a percept represents a *feature* of the percept; for example, if the percept is related to a juicy red apple, the textual, symbolic description of that percept looks like $\mathbf{p} = \{$*apple, red, juicy*$\}$.

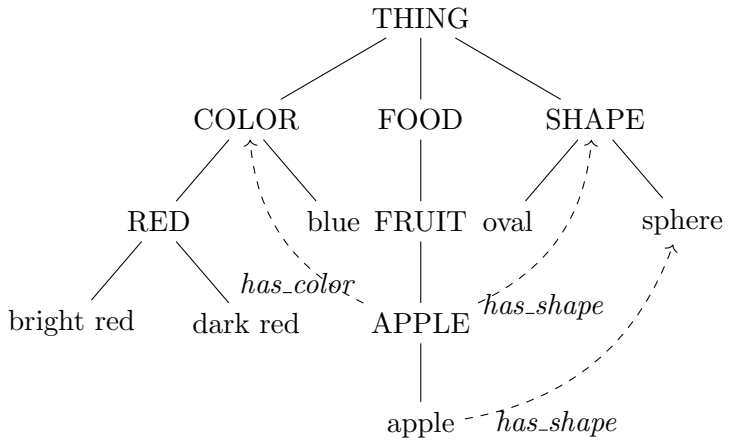


Figure 5.1: An example of enriched taxonomy representation. The classes are internal nodes represented by capital letters. The instances are the leaves represented by not-capital letters. Simple lines are the subsumption relations, and the oriented dashed arrows represent object properties among classes or instances. Datatype properties are not represented for clarity.

5.3.2 Probabilistic tree derivation from ontology

We define the *probabilistic tree derivation* over the space of fragment trees S_T of the O ontology as the process that draws from S_T the more plausibly fragment trees for a percept and links its features. It is formalized by the 3-tuple $A = (T, N, G)$ where T is a set of *terminal symbols*, that are the instances of the ontology, then $T = I$. N is a set of *nonterminal symbols* that contains all classes in C, so that $N = C$; G is a set of fragment trees from the ontology including the fragment trees representing the percept.

The fragment trees take the form of a taxonomy where each node (including the root) is labeled with a nonterminal, and each leaf is a label with terminal (when it is an instance) and nonterminal (when it is a class). Nonterminal nodes are the *frontier nonterminal*, and represent *substitution sites* in which new concepts or instances can be linked.

The fragment trees in G are automatically generated from the ontology by extracting all possible sub-graphs from the pure taxonomy with different depth. The fragment trees for the percept are generated considering each feature and representing it by one-depth tree whose root is a concept and whose leaf is an instance. Examples of fragment trees are shown in figure 5.2b and 5.2d; the one-depth fragment trees `APPLE` − `apple` and `RED` − `red`

are the fragment trees representing each features of $\mathbf{p} = \{apple,\ red\}$.

A *derivation* is a sequence of fragment trees $\mathbf{f}$ involved in the knowledge acquisition process. A final tree t is the resulting tree when combining the involved fragment trees; different derivations can generate the same tree depending on the state of the knowledge (i.e, if a concept already exists or not), as shown in figure 5.2. In this figure, a different set of fragment trees and different states of knowledge are represented; each of this situation, by rewriting the substitution sites, generates the same final tree. The arrows represent the substitution sites, i.e., the nonterminal that can link to others fragment trees. When only instances are acquired, the correspondent class nodes are overlapped, as shown in figure 5.2a and 5.2b. When new concepts are estimated they are linked to the more probable sites leading to the insertion of a new high-level concept too.

The process of building derivations is probabilistic when we compute a probability to build a derivation, and hence the probabilities of the fragment trees that compose them. An higher probability allows to discriminate the ontological nodes that better link to the features of the percept.

Begin $P(\mathbf{f})$ the probability of a derivation $\mathbf{f}$, it is the product of all the probabilities of the fragment trees in $\mathbf{f}$; the probability of a single fragment tree f is denoted by $P(f)$ and it represents the probability that the fragment tree f is drawn from the disribution. As consequence:

$$P(\mathbf{f}) = \prod_{f \in \mathbf{f}} P(f)$$

and the probability of a tree t will be:

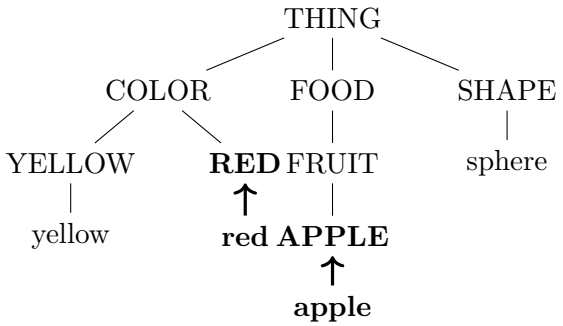
$$P(t) = \sum_{\mathbf{f}:tree(\mathbf{f})=t} P(\mathbf{f})$$

where $tree(\mathbf{f})$ returns the whole tree for the derivation $\mathbf{f}$.

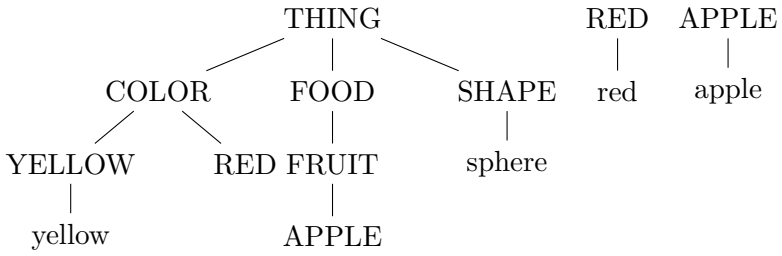
The probability of a percept $\mathbf{p}$ is the probability of the trees that can represent it in the ontology, that is:

$$P(\mathbf{p}) = \sum_{t:instance(t)=\mathbf{p}} P(t),$$

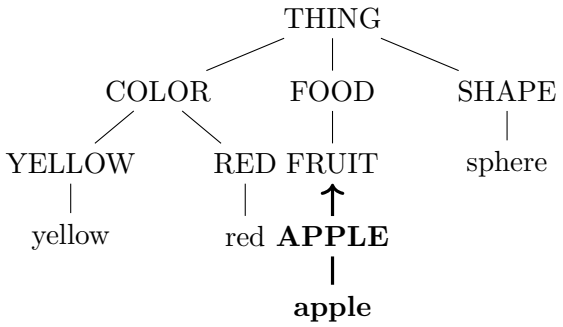
where $instance(t)$ returns the instances of terminal symbols at the leaves of t, that corresponds to the tokens of $\mathbf{p}$.



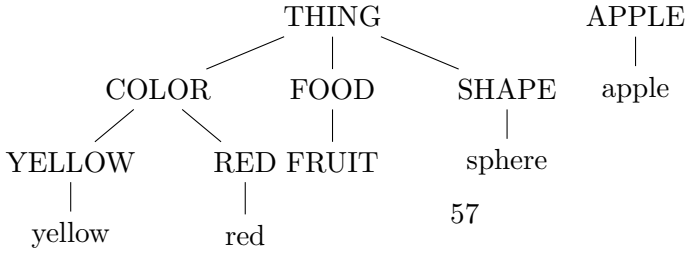
(a) A final tree obtained by linking the fragment trees in 5.2b. Only instances are acquired.



(b) A derivation of three fragment trees



(c) The tree obtained by linking the fragment trees in 5.2d. A concept and an instance are acquired.



(d) A derivation of two fragment trees

Figure 5.2: The same tree t can be obtained from different derivations, de-

5.4 The Probabilistic Model for Knowledge Acquisition

Estimating a probabilistic knowledge acquisition requires to find the most probable tree t in O that links a given percept $\mathbf{p}$, and then to insert its features in the ontology if they do not already exist. The stability property is then satisfied.

The most probable tree t is given by:

$$arg\ max_t P(t|\mathbf{p}).$$

Instead of considering the most probable tree, we find the most probable derivation $\mathbf{f}$ that generates that tree. As consequence, we define the distribution over the space of derivations. Formally, we need to identify the *posterior distribution of* $\boldsymbol{f}$ *given* $\boldsymbol{p}$, that is the Bayesian statistical inference:

$$P(\mathbf{f}|\mathbf{p}) \propto P(\mathbf{p}|\mathbf{f})P(\mathbf{f}) \tag{5.1}$$

Considering that any tree specifies a corresponding set of features, that are those in the leaves (i.e., a percept), we establish that $P(\mathbf{p}|\mathbf{f})$ is:

$$P(\mathbf{p}|\mathbf{f}) = \begin{cases} 1 & \text{if } \mathbf{p} \text{ is consistent with } \mathbf{f} \\ 0 & \text{otherwise} \end{cases}$$

So, it is necessary to compute $P(\mathbf{f})$ to solve (5.1). For this purpose, we use a Pitman-Yor Process (PYP) [**?**]. Given the percept $\mathbf{p}$, we place the Pitman-Yor process prior as the Bayesian prior, and the probability distribution G over the fragment trees becomes:

$$G \propto PYP(\alpha, \beta, G_o) \tag{5.2}$$

where α and β are the hyper parameters of the process that influence the shape of the distribution, while G_o is the *base distribution* which determines which items will fall in the support of G.

A random sample from this process is an infinite probability distribution, consisting of an infinite set of items drawn from G_o. Since it is not possible to represent such kind of distributions, the typical Chinese Restaurant Process (CRP) [1] is often used for inducing dependencies over the items in the space, and it makes the distribution finite.

The metaphor of CRP is simple: let us imagine a restaurant with an infinite number of tables, where the customers enter one at a time. When a

customer goes to the restaurant, he chooses a table according to the following rules:

1. The first customer always chooses an empty table.

2. The i-th customer may chose to sit down either to a previously occupied table or to a new empty table. Being z_i the index of the table chosen by the i-th customer, the probability of the described event is modeled by the following PYP distribution:

$$P(z_i = k|\mathbf{z}_{i-1}) = \begin{cases} \frac{n_k - \alpha}{i-1+\beta} & 1 \leq k \leq K \\ \\ \frac{K\alpha+\beta}{i-1+\beta} & k = K+1 \end{cases} \tag{5.3}$$

   where $\mathbf{z}_{i-1}$ represents the seating arrangement of the previously customers, n_k is the number of customers at table k, K is the number of the occupied tables.

The distribution is known as the Pitman-Yor Chinese Restaurant Process (PYCRP) [62]. It allows to produce a sequence of integers $\mathbf{z}$ (in this sense the process is *generative*. We use it to generate the sequence of fragment trees) and to classify these integers (any seating arrangement creates a partition) according to the rich-get-richer dynamic: as more customers sit at a particular table, this table increases in popularity, so new patrons are more likely to sit down at it not considering empty tables.

The joint probability $P(\mathbf{z})$ of the sequence is the product of the conditional ones at (5.3), so that $P(\mathbf{z}) = \prod_{i=1}^{n} P(z_i|\mathbf{z}_{i-1})$, being n the total number of customers at the restaurant, that is $n = \sum_{k=1}^{K} n_k$.

To produce a sequence of fragment trees, any table represents a fragment tree, so that when a table is generated, the correspondent fragment tree is created too. So, a fragment tree is associated to a table. Keeping the Chinese restaurant metaphor, we imagine such a situation as the association of a single fortune cookie to a table; the cookie is opened only by the first customer, and the message is valid for all following customers at that table. The message is the fragment tree.

The problem is to associate a fortune cookie to a table. Considering that for the PYCRP, like other PYP, the items are drawn from the base distribution, a message μ can be generated by pulling it from the base distribution G_o. A message is drawn from G_o when a new customer chooses a new table, and the message for that table is $\mu(z_i)$.

In a single restaurant all messages at all tables could not tile together, i.e., they cannot overlap to create a connected tree.

We investigate two situations for a single percept:

1. consider not full-connected trees, which correspond to different areas in the ontology;

2. consider a full-connected tree.

In the first case, the single presented PYCRP is enough: the tree t is the set of the fragment trees drawn in correspondence of each occupied table. The probability of a single fragment tree f_i for a table, that is the message of the ith customer that sits down for the first time to that table, will depend on both the probability of the chosen table z_i (that is the PYCRP distribution at (5.3)) and the probability of the message for that table $\mu(z_i)$ drawn from G_o. As these events are independents, then:

$$P(f_i = f|\mathbf{z}_{i-1}, \mu(z_i)) = P(z_i|\mathbf{z}_{i-1}) * G_o(f) \tag{5.4}$$

In the second case, a hierarchical combination of different PYCRPs allows to create dependencies among messages, i.e., a new fragment tree will depend from the previously extracted ones. This dependence has to be managed to enable overlap among trees so that they can be jointed. For this purpose, we define a restaurant for each class in C of the ontology. So we have as many restaurants as many concepts are. The concept $c \in C$ becomes the restaurant sign: all messages at the tables in that restaurant start with the restaurant sign, that means the fragment trees of that restaurant have the same root symbol, that is c. Formally, a separate PYCRP is defined for each concept c, and the base distribution becomes a conditional distribution on c.

The PYP distribution over fragment trees whose root symbol is c is:

$$G_c|\alpha_c, \beta_c, G_o \propto PYP(\alpha_c, \beta_c, G_o(.|c))$$

where $G_o(.|c)$ is a distribution over the fragment trees rooted with c, and α_c and β_c represent the hyperparameters of the process. Finally, to generate a full-connected tree $\mathbf{f}$, the first restaurant with sign `THING` is considered, and the f_1 component is drawn from the correspondent distribution that is $G_{\texttt{THING}}$ with frontiers $l_1,\ l_2,\ ...\ l_m$. Then the others fragments trees f_2, f_3, ..., f_m are drawn in turn from the distribution G_{l_1}, G_{l_2} and $G_{l_{m-1}}$ respectively. The process is iterated until a full-connected tree is generated, and it can start again.

In this case, the probability of a single fragment tree f_i is conditioned by c, so that the equation at (5.4) becomes:

$$P(f_i = f|c, \mathbf{z}_{i-1}, \mu(z_i)) = P(z_i|\mathbf{z}_{i-1}) * G_o(f|c). \tag{5.5}$$

In this paper we discuss only the first case, whose results are already satisfactory. Future works will regard the comparison and the evaluation of the better strategy among the two. Moreover, we will consider the inferences that are intrinsic to the ontological structure and that allow to insert new knowledge in the aftermath. For example, assuming that a red apple is perceived and that in the ontology there is not defined an apple yet, but there is the red color, the proposed method will instantiate the concept `APPLE` with the instance `apple`. Then, the relation *has_color* will be instanced for `apple` and `red` too.

5.4.1 Modeling the base distribution

The definition of the proposed base distribution represents an interesting contribution when used in a PYP. As shown, the base distribution G_o defines the probability that an item (a fragment tree in our case) falls in support of the general PYP distribution G, and it establishes which things are more plausibly than other (and hence are useful for combining the correct final tree). We define a method to make more probable the fragment trees more close to the features of the percept. Then we focus on the definition of a base probability that depends on some similarity measures among the characteristics of the percept and concepts of the ontology. We refer to the *selectional preference* that finds the role of words that can fill a specific argument of a predicate. Resnik [65] proposes a probabilistic model for selectional preference capturing the co-occurrence behavior of predicates and conceptual classes in taxonomy. A prior distribution, depending on frequencies in a corpus, captures the probability of a category of the word occurring as the argument in predicate-argument structure, regardless of the identity of the predicate. For example, given the verb-subject relationship, the prior probability for the concept *fruit* may tend to be higher than the prior probability for the concept *inkwell* (i.e., the word *fruit* may occur more frequently than the word *inkwell*). However, once the instance of the predicate is taken into account, the probabilities can change: if the verb is *write*, then the probability for *inkwell* could become higher than its prior, and *fruit* will be lower. In probabilistic terms, it is the difference between this posterior distribution and the prior distribution that determines selectional preference.

The form of the *relative entropy* by Kullback and Lebier [45] provides an appropriate way to quantify such a difference:

$$E = \sum_{c \in C} P(c|p_i) log \frac{P(c|p_i)}{P(c)} \tag{5.6}$$

where $P(c|p_i)$ is the posterior probability and $P(c)$ is the prior. In our terms, $P(c|p_i)$ represents the probability of the concept c given the feature p_i of the percept, and $P(c)$ is the prior probability of the concept c (i.e. the probability depending on the occurrences in the corpus by Resnik).

Intuitively, E measures how much information the feature p_i provides about the concept c. The better $P(c)$ approximates $P(c|p_i)$, less influence p_i is having on c, and therefore the less strong its selectional preference. An important consideration is that words that fit very well can be expected to have higher posterior probabilities $P(c|p_i)$, compared to their priors $P(c)$.

Given this definition, the natural way to characterize the polarity of a particular concept c to a feature p_i is by its relative contribution to the overall entropy. This contribution is computed by the polarity function $E_r : C \times \mathbf{p} \rightarrow [0,\ 1]$ defined as following:

$$E_r(c,\ p_i) = \frac{1}{E} P(c|p_i) log \frac{P(c|p_i)}{P(c)} \tag{5.7}$$

If p_i has high polarity in respect to the concepts in a fragment tree f, then f has the higher probability to be extracted from the ontology, and hence to fall in support of PYCRP.

The whole polarity of a fragment tree f is then defined as the arithmetic means of the polarities of its nodes; in this way we give more polarity to the fragment trees composed by a single node (whit depth zero) in respect to the fragment trees with higher depth that contain the same node, leading to a more punctual association (i.e. single concepts are preferred in respect to deeper fragment trees). Let $G_o(f)$ be such a probability, then:

$$G_o(f) = \frac{\sum_{c \in f} E_r(c,\ p_i)}{n_c} \tag{5.8}$$

begin n_c the number of nodes in f.

It is obvious that in the PYCRPs hierarchical composition, the $G_o(.|c)$ is the conditioning of E in c, that means $E_o(f|c)$ but given the form of E such a condition has no effects on the base distribution form, so

$$G_o(f|c) = G_o(f) \forall c.$$

To compute the (5.8) equation, the posterior $P(c|p_i)$ and the prior $P(c)$ have to be defined.

The posterior distribution

To define the posterior distributions $P(c|p_i)$, we refer to semantic similarity measure by Wu-Palmer [80] that determines how two concepts are semantically similar basing on their distance on the Wordnet taxonomy [52], according to the score:

$$P(c|p_i) = 2 * \frac{depth(lcs(c, p_i))}{(depth(syn(c)) + depth(syn(p_i)))}$$

where *depth* is a function that returns the depths of the synsets of its argument in the WordNet taxonomies, and *lcs* is the least common subsumer of its arguments in WordNet too.

The prior distribution

The prior, which captures the occurrences of a concept, and hence how it is widespread, is modeled by the *semantic density* of this concept in the Wordnet taxonomy, so that:

$$P(c) = \frac{syn(c)}{all_{syn}}$$

where $syn(c)$ returns the number of synsets of its argument, while all_{syn} is the number of all the synsets in the Wordnet taxonomy, that is 117000 as reported at [1].

5.4.2 Acquisition of new knowledge

The acquisition of the features of a new percept is based on the computation of the maximum probability derivation according to (5.4) if the features have to be linked to different ontological areas, or according to (5.5) if the features have to be linked to a single full-connected tree.

In any cases, if a feature is already in the ontology, it should not to be acquired. For this reason, after the probabilities estimation, a similarity measure between ontological elements in the fragment trees and a feature is computed. This measure keeps in account syntactic similarity because syntax is a strength aspect for discriminating the equivalence of two textual concepts; furthermore, to correct disambiguate the sense of the feature, a semantic contribution is considered too.

[1] https://wordnet.princeton.edu/

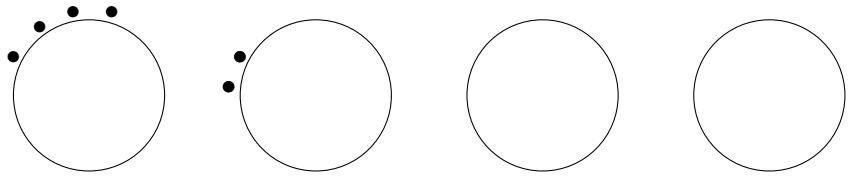


Figure 5.3: See text.

The measure is the weighted sum of the Jaro-Winkler distance [] and the Wu-Palmer distance [], then:

$$sim(w_1,\ w_2) = \delta * jaro(w_1,\ w_2) + \gamma * wup(w_1,\ w_2). \tag{5.9}$$

The main motivation to consider the Jaro-Winkler distance is the characteristic of the strings to compare, that are short words as the labels of the ontology.

The experiments show that to give more weight to the syntactic contribution leads to better identification of the equivalent concept in the ontology, and the parameters are set with the following values: $\delta = 0.7$ and $\gamma = 0.3$. A feature is considered already in the ontology if the similarity measure is above the threshold $\tau > 0.9$, otherwise it is acquired.

5.4.3 A toy example

Let consider the fragment of ontology in figure 5.1 and suppose that the robot is perceiving the color red for the instance `apple`, that is not yet included. Then $\mathbf{p} = \{$*red, apple*$\}$. The similarity measure in respect to all elements in the ontology at (5.9) returns the following values (that can be tested by the demo version of the library at [2]):

- *red*
- *apple*

For the concept `red` the PYCRP starts, and an integer is drawn with probability – and a fragment tree is drawn from the fragment tree set

5.4.4 A case of study for self-repairing

An interesting scenario for which the incremental knowledge acquisition can have a great impact is related to the robotic self-repair. Physical self-repair

[2] xx

could become critical in applications where no humans are around to assist or repair the robots, that has to become able to heal itself. Also, this can be useful for cooperating with humans to repair other machines. To know

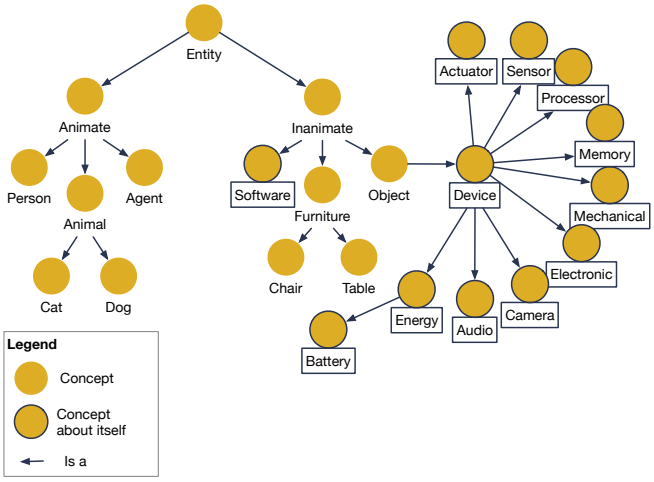


Figure 5.4: The fragment of the ontology including all the concepts about itself. These concepts are framed.

the damaged resource is the first step for self-repairing; the robot acquires awareness of such a resource and becomes able to understand how to repair it, for example by identifying what object can be used in place to it, or the set of necessary actions for replacing. Generally, the robots are able to self diagnose and to detect if and what component is trouble. The main goal of diagnosis is to check every device and its functionalities, and to publish whether the device has or has not an error: until this moment, the robot has not awareness about this device. It passively communicates to the human the internal state, but it is not able to understand such a state. To start the self-repair, the robot has to semantically conceptualize the device for the opportune intervention. In other words, by self-diagnosis the robot knows that a device is trouble, but it does not know such a resource, which remains an abstract concept until it is not acquired in the knowledge. Our experiments involved this first-step.

The robot could not know anything in advance about its devices, or it could have a partially knowledge about itself.

In our experiment, the whole knowledge of the robot includes few concepts about itself; in figure 5.4 a fragment of the ontology is represented with all these concepts.

We force the robot to diagnose an hardware trouble. The self diagnose will return the variable d that is the name of the damaged resource. Let suppose that the CPU is this resource, so $d = cpu$. The mapping process map will invoke the $merg$ function because the similarity measure is under the threshold for each concepts in the ontology.

Fragment Tree	***Result***
[Processor]	**0.04662**
[Camera]	0.03195
[Memory]	0.01704
[Device, Processor]	0.01251

Table 5.1: PYP results for the CPU resource.

The table 5.1 shows the results of the merging process. The concept *Processor* is the more probable than the other fragment trees in the ontology; a new concept *CPU* is created as children of *Processor*, with the correspondent instance. Since this moment, the robot has conceptualized the resource and it can refer to the knowledge about the processor for self repairing it or for identifying the possible new processor among a set of available spare parts.

New knowledge is discovered and the ontology is updated as drawn in figure 5.5. The same figure shows the emergent concepts for the resource; it is important to notice that the merging process allows to select concepts that are semantically similar to the percept, excluding concepts that have a completely different meaning.

5.5 Experiments

We compared our approach to those proposed at [] where the authors faced the *organizational problem* arising when someone has to organize a set of objects in an environment; in the robotic perspective, the goal is to make the robot able to infer where to best place a particular, previously unseen object or where to reasonably search for a particular type of object given past observations about the allocation of the others.

The authors define this kind of problem as a *classification problem* because the robot has to choose the best *location* in the environment for a previously unseen object. A location is defined as a set of products (i.e. a class); to allocate a new object, the robot has to consider the *features* of the objects in a location; among such a features, the semantic similarity is

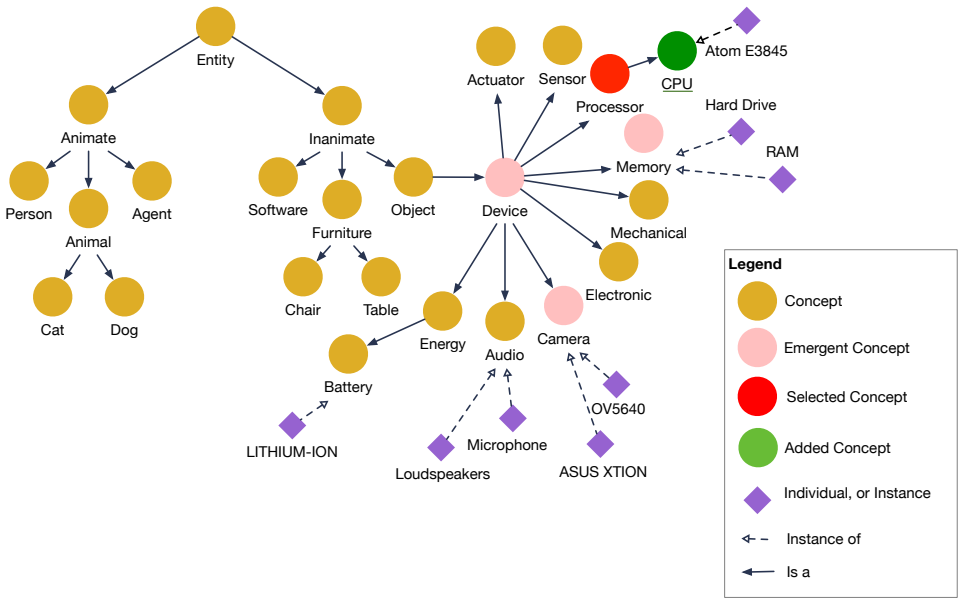


Figure 5.5: The knowledge acquisition related to the *CPU* concept with its instance represented by the diamond shape. The emergent concepts are highlighted, among them the more probable is the candidate parent.

the main one. The semantic similarity is, We can compare with them by considering the same kitchen scenario because according to their position all objects of the same class are allocated into a single location.

Datasets

The datasets the authors propose for their experiments gathered data within twelve different kitchen environments. Ten of these were acquired by simulating the process of placing objects within a fictitious kitchen, two were obtained by carefully annotating the object locations in two real kitchens. They divided each kitchen environment into *locations* representing containers, drawers, fridge, etc .

5.6 Conclusions

A method for cumulative knowledge acquisition by robot is presented. A percept is formalized as a set of textual features that have to be linked to the ontology if they are not already modeled. For this purpose, we think to

define a Pitman-Yor process in the Chinese restaurant variant: it generates a set of fragment trees from the ontology, that are subgraphs linking the percept description and taking the form of elementary trees typical of the Tree Substitution Grammar. The elementary trees are combined to parse a sentence according to the grammatical production rules. The fragment trees are then modeled in the same way to link the features according to a suitable relative entropy among the concepts in the ontology and the features. The result is that the best fragment trees combination corresponds to the percept and the features are acquired so that the ontology grows. The measure allows to discriminate if a feature is already modeled in the ontology, and the acquisition fails to avoid redundancy.

Different strategies may be combined for optimizing the model, such as the definition of the priors for the hyperparameters of the process, or the choice to consider either a full-connected subgraph of the ontology (and in this case a or a set of subgraphs correspondents to a different area in the ontology.

Chapter 6

Inner Speech

6.1 Introduction

Inner speech plays a central role in daily life. A person thinks over her mental states, perspectives, emotions and external events by generating thoughts in the form of linguistics sentences. Talking to herself enables the person to pay attention to internal and external resources, to control and regulate her behavior, to retrieve memorized facts, to learn and store new information and, in general, to simplify otherwise demanding cognitive processes [70].

Moreover, inner speech allows restructuring the perception of the external world and the perception of self by enabling high-level cognition, including self-control, self-attention, and self-regulation.

Even if second-order thoughts may not need language but, for example, images or sensations, Bermudez [9], Jackendoff [26], among others, argue that genuine conscious thoughts need language. In the light of the above considerations, inner speech is an essential ingredient in the design of a self-conscious robot.

We model such a necessary capability within a cognitive architecture for robot self-consciousness by considering the underlying cognitive processes and components of inner speech.

It should be remarked that in the present paper such processes are taken into account independently from the origin of the linguistics abilities which are supposed acquired by the robot.

In Section 6.2 we show a brief overview of the cognitive models underlying the proposed robot architecture, which is detailed in Section 6.3. Conclusions and future works about the proposed robot architecture are discussed in Section 6.4.

6.2 Models of inner speech

Inner speech cannot be directly observed, thus reducing the scope for empirical studies. However, theoretical perspectives were developed during the last decades, and some of them are recognized in different research communities.

Vygotsky [76] conceives inner speech as the outcome of a developmental process during which the linguistics interactions, such as between a child and a caregiver, are *internalized*. The linguistically mediated explanation for solving a task thus becomes an internalized conversation with the self, when the learner is engaged in the same o similar cognitive tasks.

Morin [53][54] claims that inner speech is intrinsically linked to self-awareness. Self-focusing on an internal resource triggers the inner speech, and then it generates self-awareness about such a resource. Typical sources for the self-focus process are social interactions or mirror reflections by physical objects.

Baddeley [6] discussed the roles of rehearsal and working memory, where the different modules in the working memory are responsible for inner speech rehearsal. In particular, the *central executive* oversees the process; the *phonological loop* deals with spoken and written data, and the *visuospatial sketchpad* deals with information in a visual or spatial form. The phonological loop is composed of the *phonological store* for speech perception, which keeps information in a speech-based form for a very short time (1-2 seconds), and of the *articulatory control process* for speech production, that rehearses and stores verbal information from the phonological store.

Inner speech is usually conceived as the back-propagation of produced sentences to an inner ear: thus, a person rehears the internal voice she delivers. Steels [71] argued that the language re-entrance allows refining the syntax emerging during linguistic interactions within a population of agents. The syntax thus becomes more complex and complete by parsing previously produced utterances by the same agent.

In the same line, Clowes [25] discussed an artificial agent implemented by a recurrent neural network whose output nodes are words interpreted as possible actions (for example 'up,' 'left,' 'right,' 'grab'). When such words are re-entrant by back-propagating the output to the input nodes, then the agent achieved the task in far fewer generations than in the control condition where words are not re-entrant.

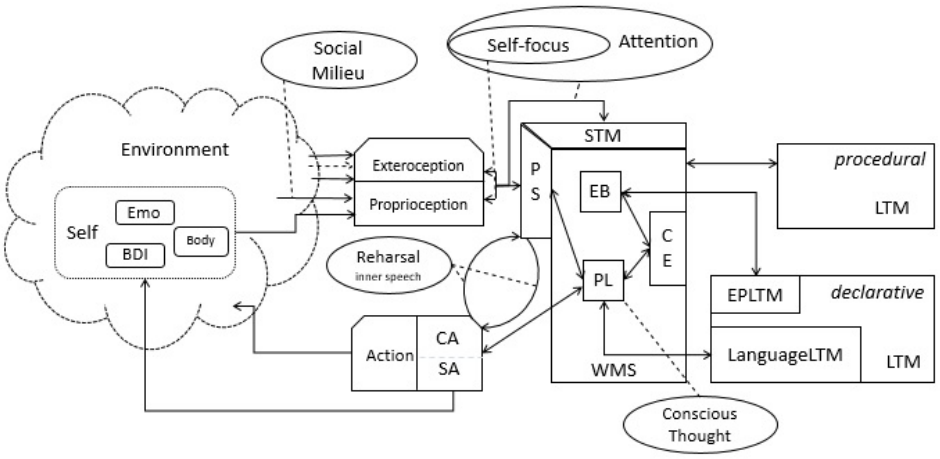


Figure 6.1: The proposed cognitive architecture for inner speech.

6.3 The cognitive architecture for inner speech

Figure 6.1 shows the proposed robot cognitive architecture for inner speech. Such a representation refers to the Standard Model of Mind proposed by Laird et al.[46]. Here, the structure and processing of the Standard Model are decomposed with the aims to integrate the components and the processes defined by the inner speech theories previously discussed.

6.3.1 Perception and Action

The perception of the proposed architecture includes the *proprioception* module related to the self-perception of the emotions (Emo), the belief, desires and intentions (BDI) and the robot body (Body), and the *exteroception* module related to the perception of the outside environment.

The proprioception module, according to Morin [53], is also stimulated by the *social milieu* which, in the considered perspective, includes the social interactions of the robot with the others entities in the environment, as physical objects like the mirrors and the cameras and others robots or humans, by means face-to-face interaction that foster self-world differentiation.

The motor module is decomposed in three sub-components: the *Action* module, the *Covert Articulation* module (CA) and the *Self Action* module (SA). In particular:

- The *Action* module represents the actions the agent performs on the

outside world producing modifications to the external environment (not including the self) and the working memory.

- The *Covert Articulation* (CA) module rehearses information from the *Phonological Store* (PS), i.e., the perceptual buffer for speech-based data considered as a sub-component of the short-term memory (see below). Such a module acts as the inner voice heard by the phonological store by rounding information in a loop. In this way, the inner speech links the covert articulation to the phonological store in a round loop.

- The *Self Action* (SA) module represents the actions that the agent performs on itself, i.e., self-regulation, self-focusing, and self-analysis.

6.3.2 The Memory System

The memory structure, inspired by the Standard Model of the Mind is divided into three types of memories: the short-term memory (STM), the *procedural* and the *declarative* long-term memory (LTM), and the working memory system (WMS).

The short-term memory holds sensory information on the environment in which the robot is plunged that were previously coded and integrated with information coming by perception. As previously mentioned, the short-term memory includes the phonological store.

Information flow from perception to STM allows storing the aforementioned coded signals. In particular, information from perception to the phonological store is related to *conscious* thoughts from exteroception, and to *self-conscious* thoughts from proprioception.

The information flow from the working memory system to perception provides expectations or possible hypotheses that are employed for influencing the *attention* process. In particular, the flow from the phonological store to proprioception enables the *self-focus* modality.

The long-term memory holds learned behaviors, semantic knowledge, and experience. In the considered case, the *declarative* LTM contains the linguistics information in terms of lexicon and grammatical structures, i.e., the *LanguageLTM* memory. The declarative linguistics information is assumed acquired, as specified above, and represent the *grammar* of the robot. Moreover, the *Episodic Long-Term Memory* (EBLTM) is the declarative long-term memory component which communicates to the *Episodic Buffer* (EB) within the working memory system, that acts as a 'backup' store of long-term memory data.

The *procedural* LTM contains the composition rules according to which the linguistic structures are arranged for producing sentences at different levels of completeness and complexity. A procedure does not concern the grammatical plausibility of the structures only. Other rules concerning the regulation, the focusing and the restructuring of resources within the whole environment (including the self) are to be considered.

Finally, the working memory system holds task-specific information 'chunks' and streamlines them to the cognitive processes during the task execution, step by step according to the cognitive cycle of the Standard Model of the Mind. The working memory system deals with cognitive tasks such as mental arithmetic and problem-solving. The *Central Executive* (CE) subcomponent manages and controls the linguistic information of the rehearsal loop by the integrating (i.e., combining) data from the phonological loop and also drawing on data held in the long-term memory.

6.3.3 The Cognitive Cycle

In brief, a cognitive cycle starts with the perception that converts external signals in linguistics data and holds them into the phonological store. The central executive manages the inner thinking process by enabling the working memory system to selectively attend to some stimuli or ignore others, according to the rules stored within the LTMs, and by orchestrating the phonological loop as a slave system.

At this stage, a conscious thought emerges as a result of a single round between the phonological store and the covert articulation triggered by the phonological loop, once the central executive has retrieved the data for the process. The phonological loop enables the covert articulation which acts as a motor for the internal production, and whose output stream is heard to the phonological store. The output stream also affects the self which is then regulated and restructured.

Once the conscious thought is elicited by inner speech, the perception of the new context could take place, repeating the cognitive cycle.

6.4 Conclusions

In this chapter, an initial cognitive architecture for inner speech cognition is presented. It is based on the Standard Model of Mind which was decomposed for including some typical components of the inner speech's models for human beings.

The working memory system of the architecture includes the *phonological loop* considered by Baddeley as the main component for storing spoken and written information and for implementing the cognitive rehearsal process.

The covert dialogue is modeled as a loop in which the *phonological store* hears the inner voice produced by the *covert articulator* process. The *central executive* is the master system which drives the whole system.

By retrieving linguistic information from the long-term memory, the central executive contributes to creating the linguistic thought whose surface form emerges by the phonological loop.

Acknowledgments

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-17-1-0232. Any opinions, finding, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force.

Appendix

The following papers has been submitted, accepted for publications, printed, under acknowledgement of Air Force Office of Scientific Research award FA9550-17-1-0232:

- A. Cangelosi, A. Chella (2018): Lo sviluppo dei concetti nei robot e nelle macchine intelligenti (The development of concepts in robots and intelligent machines), in: F. Gagliardi, M. Cruciani, A. Velardi (eds.), *Concetti e processi di categorizzazione,* pp. 27 - 56, Aracne.

- S. Vinanzi, M. Patacchiola, A. Chella, A. Cangelosi (2018): Would a Robot Trust You? Developmental Robotics Model of Trust and Theory of Mind, *Philosophical Transactions of the Royal Society B* (in press).

- A. Chella, F. Lanza, V. Seidita (2018): Endowing Robots with Self-Modeling for Trustworthy Human-Robot Interactions, *RoboticsLab DIID-Unipa Technical Report.*

- A. Chella, F. Lanza, V. Seidita (2018): A Cognitive Architecture for Human-Robot Teaming Interaction, in: *Proc. of the 6th International Workshop on Artificial Intelligence and Cognition, AIC-2018* (in press).

- A. Chella, F. Lanza, V. Seidita (2018): Representing and Developing Knowledge using Jason, CArtAgO and OWL, in: *Proc. of the 19th Workshop From Objects to Agents WOA-2018.*

- A. Chella, F. Lanza, V. Seidita (2018): Human-Agent Interaction, the System Level Using JASON, in *Proc. of the AAMAS-IJCAI-ECAI 6th International Workshop on Engineering Multi-Agent Systems EMAS-2018.*

- A. Chella, F. Lanza, A. Pipitone, V. Seidita (2018): Knowledge Acquisition through Introspection in Human-Robot Cooperation, *Biologically Inspired Cognitive Architectures*, Vol. 25, August 2018, Pages 1-7.

- A. Chella, F. Lanza, A. Pipitone, V. Seidita (2018): Human-Robot Teaming: Perspective on Analysis and Implementation Issues, *Proc. of AI*IA, Working Group on Artificial Intelligence and Robotics* (in press).

- A. Pipitone, F. Lanza, V. Seidita, A. Chella (2019): Inner Speech for a Self-Conscious Robot. In: A. Chella, D. Gamez, P. Lincoln, R. Manzotti, J. Pfautz (eds.): Papers of the 2019 Towards Conscious AI Systems Symposium (TOCAIS 2019). CEUR-WS vol 2287. `http://CEUR-WS.org/Vol-2287/paper14.pdf`

- A. Chella, A. Pipitone (2019): The inner speech of the IDyOT, *Physics of Life Reviews* (available online) `https://doi.org/10.1016/j.plrev.2019.01.016`.

- A. Chella, F. Lanza, A. Pipitone, V. Seidita (2019): Incremental Knowledge Acquisition in Human-Machine Teaming, *Knowledge-Based Systems Journal* (submitted).

- A. Chella (2019): Inner Speech and Robot Consciousness, *Proc. of The Science of Consciousness*, Interlaken, Switzerland, June 25-28, 2019 (submitted).

Bibliography

[1] D.J. Aldous. Exchangeability and related topics. In *École d'Été St Flour 1983*, pages 1–198. Springer-Verlag, 1985. Lecture Notes in Math. 1117.

[2] J.S. Allwood and P. Gärdenfors. *Cognitive Semantics: Meaning and Cognition*. New series]. J. Benjamins Pub., 1999.

[3] John R Anderson, Michael Matessa, and Christian Lebiere. Act-r: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, 12(4):439–462, 1997.

[4] Jesper Andersson, Luciano Baresi, Nelly Bencomo, Rogério de Lemos, Alessandra Gorla, Paola Inverardi, and Thomas Vogel. Software engineering processes for self-adaptive systems. In *Software Engineering for Self-Adaptive Systems II*, pages 51–75. Springer, 2013.

[5] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, New York, NY, USA, 2003.

[6] A Baddeley. Working memory. *Science*, 255(5044):556–559, 1992.

[7] Luciano Baresi and Carlo Ghezzi. The disappearing boundary between development-time and run-time. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, pages 17–22. ACM, 2010.

[8] D Paul Benjamin, Damian M Lyons, and Deryle W Lonsdale. Adapt: A cognitive architecture for robotics. In *ICCM*, pages 337–338, 2004.

[9] Jose Luis Bermudez. *The Paradox of Self-Consciousness*. MIT Press, 2012.

[10] G. Blair, N. Bencomo, and R. B. France. Models@ run.time. *Computer*, 42(10):22–27, Oct 2009.

[11] Olivier Boissier, Rafael H Bordini, Jomi F Hübner, Alessandro Ricci, and Andrea Santi. Multi-agent oriented programming with jacamo. *Science of Computer Programming*, 78(6):747–761, 2013.

[12] Rafael H Bordini and Jomi F Hübner. Bdi agent programming in agentspeak using jason. In *International Workshop on Computational Logic in Multi-Agent Systems*, pages 143–164. Springer, 2005.

[13] Rafael H Bordini, Jomi Fred Hübner, and Michael Wooldridge. *Programming multi-agent systems in AgentSpeak using Jason*, volume 8. John Wiley & Sons, 2007.

[14] Michael Bratman. Intention, plans, and practical reason. 1987.

[15] Michael E Bratman. What is intention. *Intentions in communication*, pages 15–32, 1990.

[16] Paolo Bresciani, Anna Perini, Paolo Giorgini, Fausto Giunchiglia, and John Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.

[17] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, Estevam R. Hruschka, Jr., and T.M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, pages 1306–1313. AAAI Press, 2010.

[18] Gail A. Carpenter and Stephen Grossberg. The art of adaptive pattern recognition by a self-organizing neural network. *Computer*, 21(3):77–88, March 1988.

[19] Richard Caruana. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann, 1993.

[20] Christiano Castelfranchi and Rino Falcone. *Trust theory: A socio-cognitive and computational model*, volume 18. John Wiley & Sons, 2010.

[21] Cristiano Castelfranchi and Rino Falcone. Delegation conflicts. *Multi-agent rationality*, pages 234–254, 1997.

[22] Cristiano Castelfranchi and Rino Falcone. Towards a theory of delegation for agent-based systems. *Robotics and Autonomous Systems*, 24(3-4):141–157, 1998.

[23] Wayne D Christensen, Cliff A Hooker, et al. Representation and the meaning of life. *Representation in mind: New approaches to mental representation*, pages 41–69, 2004.

[24] Andy Clark. *Mindware: An introduction to the philosophy of cognitive science*. Oxford University Press, 2000.

[25] Robert Clowes. A self-regulation model of inner speech and its role in the organisation of human conscious experience. *Journal of Consciousness Studies*, 14(7):59–71, 2007.

[26] Pragmatics & Cognition. How language helps us think. *Pragmatics & Cognition*, 4(1):1–34, 1996.

[27] Trevor Cohn, Phil Blunsom, and Sharon Goldwater. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096, 2010.

[28] Oscar Corcho and Asunción Gómez-Pérez. A roadmap to ontology specification languages. Springer, 2000.

[29] Massimo Cossentino, Nicolas Gaud, Vincent Hilaire, Stéphane Galland, and Abderrafiâa Koukam. Aspecs: an agent-oriented software process for engineering complex systems. *Autonomous Agents and Multi-Agent Systems*, 20(2):260–304, 2010.

[30] Daniel Clement Dennett. *The intentional stance*. MIT press, 1989.

[31] Rino Falcone and Cristiano Castelfranchi. The human in the loop of a delegated agent: The theory of adjustable social autonomy. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 31(5):406–418, 2001.

[32] Rino Falcone and Cristiano Castelfranchi. Social trust: A cognitive approach. In *Trust and deception in virtual societies*, pages 55–90. Springer, 2001.

[33] Rino Falcone and Cristiano Castelfranchi. Trust dynamics: How trust is influenced by direct experiences and by trust itself. In *Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004. Proceedings of the Third International Joint Conference on*, pages 740–747. IEEE, 2004.

[34] Rino Falcone, Michele Piunti, Matteo Venanzi, and Cristiano Castelfranchi. From manifesta to krypta: The relevance of categories for trusting others. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(2):27, 2013.

[35] Stan Franklin, Tamas Madl, Sidney D'Mello, and Javier Snaider. Lida: A systems-level architecture for cognition, emotion, and learning. *IEEE Transactions on Autonomous Mental Development*, 6(1):19–41, 2014.

[36] French. Pseudo-recurrent connectionist networks: An approach to the 'sensitivity-stability' dilemma. *Connection Science*, 9(4):353–380, 1997.

[37] M Georgeff and A Rao. Rational software agents: from theory to practice. In *Agent technology*, pages 139–160. Springer, 1998.

[38] Stephen Grossberg. Adaptive resonance theory: How a brain learns to consciously attend, learn, and recognize a changing world. *Neural networks : the official journal of the International Neural Network Society*, 37:1–47, 2013.

[39] Nicola Guarino, Daniel Oberle, and Steffen Staab. What is an ontology? In Steffen Staab and Ruder Studer, editors, *Handbook on Ontologies*. Springer, second edition, 2009.

[40] Sumit Gulwani, José Hernández-Orallo, Emanuel Kitzelmann, Stephen H. Muggleton, Ute Schmid, and Benjamin Zorn. Inductive programming meets the real world. *Commun. ACM*, 58(11):90–99, October 2015.

[41] Trung Dong Huynh, Nicholas R Jennings, and Nigel R Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 13(2):119–154, 2006.

[42] Aravind K. Joshi and Yves Schabes. Handbook of formal languages, vol. 3. chapter Tree-adjoining Grammars, pages 69–123. Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[43] JA Scott Kelso. *Dynamic patterns: The self-organization of brain and behavior*. MIT press, 1997.

[44] David E Kieras and David E Meyer. An overview of the epic architecture for cognition and performance with application to human-computer interaction. *Human-computer interaction*, 12(4):391–438, 1997.

[45] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951.

[46] John E. Laird, Christian Lebiere, and Paul S. Rosenbloom. A Standard Model of the Mind: Toward a Common Computational Framework across Artificial Intelligence, Cognitive Science, Neuroscience, and Robotics. *AI Magazine*, 38(4):13, December 2017.

[47] John E Laird, Allen Newell, and Paul S Rosenbloom. Soar: An architecture for general intelligence. *Artificial intelligence*, 33(1):1–64, 1987.

[48] Gi Hyun Lim, Il Hong Suh, and Hyowon Suh. Ontology-based unified robot knowledge for service robots in indoor environments. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41:492–509, 2011.

[49] E Jonathan Lowe. *A survey of metaphysics*, volume 15.

[50] Fernando Martínez-Plumed, Cèsar Ferri, José Hernández-Orallo, and María J Ramírez-Quintana. Knowledge acquisition with forgetting: an incremental and developmental setting. *Adaptive Behavior*, 23(5):283–299, 2015.

[51] John McCarthy. Making robots conscious of their mental states, 1995.

[52] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

[53] Alain Morin. A neurocognitive and socioecological model of self-awareness. *Genetic, Social, and General Psychology Monographs*, 130(3):197–222, 2004.

[54] Alain Morin. Possible links between self-awareness and inner speech. *Journal of Consciousness Studies*, 12(4-5):115–134, 2005.

[55] Stephen Muggleton. Scientific knowledge discovery using inductive logic programming. *Commun. ACM*, 42:42–46, 1999.

[56] Gonzalo Nápoles, Isel Grau, Elpiniki Papageorgiou, Rafael Bello, and Koen Vanhoof. Rough cognitive networks. *Know.-Based Syst.*, 91(C):46–61, January 2016.

[57] Gonzalo Nápoles, Elpiniki Papageorgiou, Rafael Bello, and Koen Vanhoof. Learning and convergence of fuzzy cognitive maps used in pattern recognition. *Neural Process. Lett.*, 45(2):431–444, April 2017.

[58] Chris Burnett Timothy J Norman and Katia Sycara. Trust decision-making in multi-agent systems. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2011.

[59] Loris Penserini, Anna Perini, Angelo Susi, Mirko Morandini, and John Mylopoulos. A design framework for generating bdi-agents from goal models. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, page 149. ACM, 2007.

[60] Arianna Pipitone, Giuseppe Tirone, and Roberto Pirrone. Quasit: A cognitive inspired approach to question answering for the italian language. In *Proceedings of the XV International Conference of the Italian Association for Artificial Intelligence on Advances in Artificial Intelligence - Volume 10037*, AI*IA 2016, pages 464–476, Berlin, Heidelberg, 2016. Springer-Verlag.

[61] J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900, 1997.

[62] Jim Pitman. Combinatorial stochastic processes, 2006.

[63] Anand S Rao. Agentspeak (l): Bdi agents speak out in a logical computable language. In *European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, pages 42–55. Springer, 1996.

[64] Anand S Rao, Michael P Georgeff, et al. Bdi agents: From theory to practice.

[65] Philip Resnik. Selectional preference and sense disambiguation. In *Proc. of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*, Washington, DC, 1997.

[66] Alessandro Ricci, Mirko Viroli, and Andrea Omicini. Cartago: A framework for prototyping artifact-based environments in mas. *E4MAS*, 6:67–86, 2006.

[67] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7:123–146, 1995.

[68] Tracy Sanders, Kristin E Oleson, DR Billings, Jessie YC Chen, and PA Hancock. A model of human-robot trust: Theoretical model development. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 55, pages 1432–1436. SAGE Publications Sage CA: Los Angeles, CA, 2011.

[69] Murray Shanahan and Bernard Baars. Applying global workspace theory to the frame problem. *Cognition*, 98(2):157–176, 2005.

[70] Luc Steels. Inner speech: Development, cognitive functions, phenomenology, and neurobiology. *Journal of Consciousness Studies*, 10(4-5):173–185, 2003.

[71] Luc Steels. Language re-entrance and the 'inner voice'. *Journal of Consciousness Studies*, 10(4-5):173–185, 2003.

[72] Neil A Stillings, Christopher H Chase, and Mark H Feinstein. *Cognitive science: An introduction*. MIT press, 1995.

[73] Ron Sun. The importance of cognitive architectures: An analysis based on clarion. *Journal of Experimental & Theoretical Artificial Intelligence*, 19(2):159–193, 2007.

[74] WT Luke Teacy, Jigar Patel, Nicholas R Jennings, and Michael Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, 2006.

[75] Moritz Tenorth and Michael Beetz. KnowRob – A Knowledge Processing Infrastructure for Cognition-enabled Robots. *Int. Journal of Robotics Research*, 32(5):566 – 590, April 2013.

[76] L.S. Vygotsky. *Thought and Language. Revised and expanded edition*. MIT Press, 1997.

[77] William E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Bureau of the Census, 1999.

[78] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[79] Michael Wooldridge and Nicholas R Jennings. Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(2):115–152, 1995.

[80] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.