Ontological Foundations for Contrastive Explanatory Narration of Robot Plans

Alberto Olivares-Alarcos[a,], Sergi Foix[a], Júlia Borràs[a], Gerard Canal[b], Guillem Alenyà[a]

[a]*Institut de Robòtica i Informàtica Industrial, Consejo Superior de Investigaciones Científicas (CSIC) - Universitat Politècnica de Catalunya (UPC), Llorens i Artigas 4-6, Barcelona, 08028, Spain*
[b]*Department of Informatics, King's College London, London, United Kingdom*

Abstract

Mutual understanding of artificial agents' decisions is key to ensuring a trustworthy and successful human-robot interaction. Hence, robots are expected to make reasonable decisions and communicate them to humans when needed. In this article, the focus is on an approach to modeling and reasoning about the comparison of two competing plans, so that robots can later explain the divergent result. First, a novel ontological model is proposed to formalize and reason about the differences between competing plans, enabling the classification of the most appropriate one (e.g., the shortest, the safest, the closest to human preferences, etc.). This work also investigates the limitations of a baseline algorithm for ontology-based explanatory narration. To address these limitations, a novel algorithm is presented, leveraging divergent knowledge between plans and facilitating the construction of contrastive narratives. Through empirical evaluation, it is observed that the explanations excel beyond the baseline method.

Keywords: Applied ontology, Reasoning for robots, Robotics, Contrastive explainable robots, Explanatory narratives construction

1. Introduction

Autonomous artificial decision-making in environments with different agents (e.g., robots collaborating with or assisting humans) is complex to model.

URL: aolivares@iri.upc.edu (Alberto Olivares-Alarcos)

Accepted in Information Sciences. DOI: https://doi.org/10.1016/j.ins.2026.123280.

This is often due to the high degree of uncertainty and potential lack of communication among agents. For instance, robots might need to choose between competing plans (i.e. sequences of actions that would allow the robot to achieve goals), comparing their properties and deciding which one is better. Note that this decision-making problem is different from finding a single plan through automated planning, as here the idea is that there are already two valid plans to execute and the robot shall compare them and identify the best one. This might happen when a human gives an ambiguous command (e.g. 'can you bring me a drink?'), thus the robot may decompose the abstract command into different concrete goals [1], and find a plan to achieve each of the goals (such as bringing any of the available drinks). Then it would be needed to compare and disambiguate the plans. In these cases, mutual understanding of the ongoing decisions and communication between agents become crucial [2]. Hence, trustworthy robots shall be able to model their plans' properties to make sound decisions when contrasting them. Furthermore, they shall also be capable of narrating (explaining) the knowledge acquired from the comparison. Note that robots add the possibility of physically executing the plan, which may affect the human, strongly motivating the need for explanations. This may serve two purposes: justifying the robot's selection of a plan, or asking the human to help in the disambiguation (i.e. the human may prefer the plan that the robot inferred as worse). Reflecting on these thoughts, this work addresses the following research questions:

- **RQ1 -** How could robots model and reason about what differentiates plans, making one better?
- **RQ2 -** How could robots leverage the proposed ontological model to explain (narrate) what differentiates plans?

First, an ontological analysis is conducted and a new ontological model is obtained, augmenting the scope of an ontology from the literature (OCRA [3]), which answers RQ1. Specifically, a new theory for plan comparison is formalized, focusing on the properties and relationships that allow comparing plans. The robot's knowledge about the plans to compare is stored, and together with some logical rules, it is used to infer which plan is better. Second, RQ2 is addressed by introducing a novel Algorithm for Contrastive eXplanatory Ontology-based Narratives (ACXON), extending an existing literature

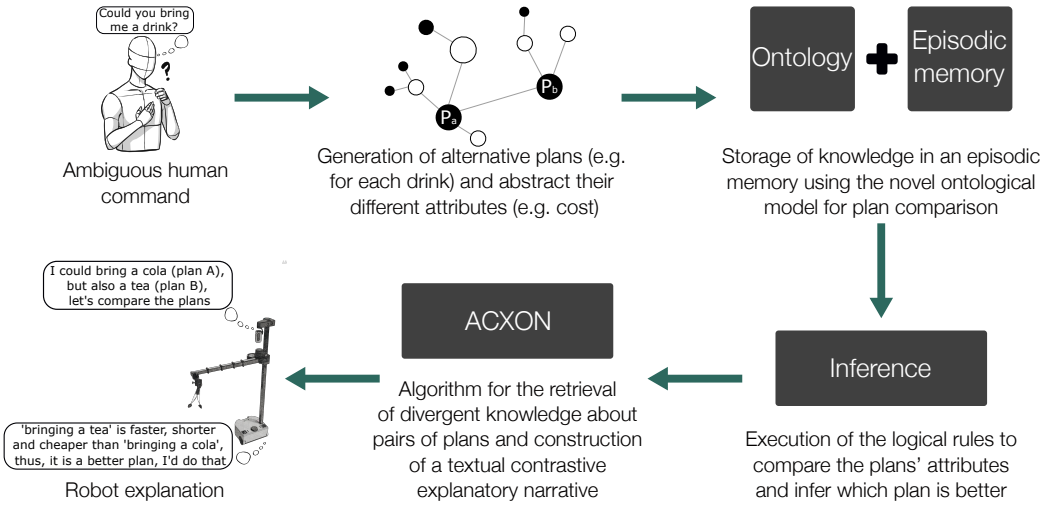

Figure 1: Proposed approach overview and its application to a prototypical human-robot interactive scenario that requires contrastive explanations. A human gives an ambiguous command, a robot generates different plans, stores knowledge about them, and contrasts them and reasons about which plan is better to execute. Finally, the inferred knowledge is used to explain the rationale.

methodology (XONCRA [4]) to contrastive cases. From the robot knowledge, ACXON retrieves the divergent information about the plans, and then it constructs the final textual contrastive narrative. The proposed algorithm produces different types of narratives based on the chosen amount of detail (specificity), addressing different users' preferences. Based on objective evaluation metrics and using several planning domains, the algorithm is evaluated with respect to the original algorithm proposed in XONCRA, which is used as a baseline. The proposed algorithm outperforms the baseline, using less knowledge to build the narratives (skipping repetitive knowledge), which shortens the time to communicate the narratives. Figure 1 provides an overview of the complete approach. At the end of the article, it is briefly discussed how the proposed algorithm can be slightly modified to enhance and restrict the knowledge selection, which helps to shorten the constructed narratives.[1]

[1] An extended abstract of this paper appears in the Proc. of AAMAS'24 [5].

2. Related work

Concerning the modeling of domain knowledge for reasoning, a common approach is to use sound formalisms such as formal ontologies. The literature shows that multiple ontologies have recently been developed [6] for different robotic applications and domains. The literature also encompasses various works that have concentrated on ontologically modeling concepts related to plans and their properties. In this regard, Bermejo-Alonso [7] conducted a survey of that domain, reviewing existing task and planning vocabularies, taxonomies, and ontologies, while also discussing their potential integration. The surveyed works mostly defined general concepts regarding plans, and only in some cases plan properties were discussed (e.g. cost and constraints). In another work, Moulouel et al. [8], proposed an ontology to model contextual knowledge in partially observable environments, which is used for probabilistic commonsense reasoning and probabilistic planning. They defined concepts to model the relationship between the context in which robot actions and observations take place and other elements used during the planning (e.g. reward function). These works are relevant in the domain, but if one wants to contrast plans during robots' decision making, there is still a need for an ontological theory that formalizes the properties of plans.

A significant source of inspiration was discovered surrounding the notion of explainable agency (i.e., explaining the reasoning of goal-driven agents and robots). In their work, Langley et al. [9] advocated for the idea that explainable agency demands four distinct functional abilities. Among those, one can find the ability to explain decisions made during plan generation, involving the comparison of alternatives. Related to this, there are some literature efforts towards investigating how to boost explainable agency by narrating or verbalizing robots' internal knowledge about plans (e.g., a plan's sequence, the rationale to include a task in the plan, etc.) [10, 11, 12, 13]. However, it is still unexplored how robots may explain their reasoning when comparing competing plans as a whole, not just specific plan tasks.

These two research domains, ontologies and explainable agency, are certainly connected. Langley et al. [9] also discussed that an explainable agent would need three main elements: a *representation* of the knowledge that supports explanations, an *episodic memory* to store agent experiences, and the ability to access the memory and retrieve and manipulate the stored content to *construct explanations*. Olivares-Alarcos et al. [4], proposed a novel methodology comprising those three elements for the construction of ex-

planatory ontology-based narratives for collaborative robotics and adaptation (XONCRA). It consisted of a knowledge base for collaborative robotics and adaptation (*know-cra*) that works as an episodic memory, and an algorithm to generate explanatory narratives using ontological knowledge (AXON). One might wonder whether the XONCRA methodology could be used to model and narrate the divergences between plans. Indeed, XONCRA uses an ontology named OCRA [3] that defines the relationship 'is better plan than', associating two plans denoting that one of the plans is considered to be better. However, OCRA did not model how an agent might find divergences between two plans and decide which one is better. Hence, it is necessary to propose a new theory for plan comparison. Moreover, while being a potential baseline solution, the narratives generated by AXON were not optimized for a contrastive case as the one that concerns this article, and a better approach shall be investigated.

Ontologies have already shown potential to become the integrative framework for explainable robots [4]. However, there are other formalisms that could complement ontologies enabling other types of reasoning. For example, non-monotonic logic supports abductive reasoning and is specifically designed to handle defeasible inferences. These allow reasoners to form tentative conclusions that can be retracted as new evidence emerges. Sridharan [13] proposed to integrate non-monotonic logic and deep learning, exploiting their synergy across representation, reasoning, and learning layers to operationalize the principles of explainable agency. Another example of complementary formalism is causal modeling, which supports reasoning to determine the relationship between causes and their effects. Love et al. [14], developed a system based on causal models to enable a robot to autonomously elicit interactions in multi-person environments, while generating real-time counterfactual explanations for its elicitation decisions. Those works presented interesting approaches to robot reasoning and explanation generation. In future work, we might consider the integration of our ontology-based methodology for explainable robots with those approaches.

3. Model for robot plan comparison

There exist several useful methodologies to construct ontologies, e.g., [15, 16], but none arise as a definite standard. Indeed, not all those methods are suitable for this work, in which the aim is to develop an ontological model from a foundational perspective (i.e. the characterization of the main

concepts is more important than the coverage of the application domain). Hence, this work relies on ontological analysis, an approach which precedes the usual ontology construction process and aims to fix the core framework for the domain ontology. Based on this selection, the steps to perform are: to set the ontology domain and scope (competency questions), to enumerate, analyze and compare existing concepts (identification of shortcomings), to develop and formalize a more solid conceptualization, and to create instances of the concepts and show their use (implementation/validation). Finally, it is also considered the documentation and maintenance of the produced theory.

3.1. Ontological scope of the proposed theory

The target novel model will formalize the ontological classes and relationships to represent knowledge of plans and their characteristics for plan comparison and later contrastive narration. In order to scope the subject domain to be represented in the intended model, a set of competency questions is proposed, which are a set of requirements on the ontology content. The ontological analysis started with a rough ontological scope (competency questions) focused on knowledge related to plan comparison. During the analysis, we discussed the scope until the set of questions became detailed and comprehensive enough for us to model how different plans compare. The analysis revealed that plan comparison requires modeling different characteristics of plans (CQ1), how the characteristics relate (CQ2), and how the plans relate based on their characteristics (CQ3). Specifically, the proposed ontological model is expected to be able to answer the following questions:

- **CQ1 -** Which are the characteristics of a plan?
- **CQ2 -** How do the characteristics of different plans relate?
- **CQ3 -** How do different plans relate to each other?

The new model is going to be built upon OCRA, re-utilizing the existing model and extending it. Therefore, OCRA's upper ontology is inherited, the DOLCE+DnS Ultralite (DUL) foundational ontology [17]. In addition to the proposed competency questions, for this work it is also interesting to represent the sequence of actions included in a plan, which is already covered by the DUL ontology.

3.2. Ontological shortcomings in OCRA and their theoretical remedy

Following the ontological analysis process, this section identifies the need for new ontological concepts and relations to cover the proposed scope (i.e. the competency questions), and defines and formalizes them. The novel theory is the first in the literature for plan comparison representation and reasoning. The model is comprehensive and is built upon existing ontologies, a great practice in ontology development, and is conceived from a foundational perspective.

3.2.1. Which are the characteristics of a plan? (CQ1)

OCRA did not define any ontological classes or relationships to model the properties of plans, thus an extension is required to be able to answer CQ1. For such an extension, a top-down approach is followed and the new model is built upon general entities defined in the upper-level ontology, DUL. In order to represent the features of other entities, DUL defines the class *Quality* as *'any aspect of an Entity (but not a part of it), which cannot exist without that Entity'*. DUL also includes the relationship *'has quality'*, *'a relation between entities and qualities'*. In this work, we specialize both the class and the relation to define the particular qualities of plans.

Plans can have many different qualities that would highly depend on the application domain. Defining all of them is out of the scope of this article. Instead, we aim to find a set of qualities that are usually present in most of the planning domains, with a special focus on those more relevant to robotics. Particularly, we will use temporal planning domains in which actions have a duration and are modeled using PDDL 2.1 [18]. In robotics, finding a plan that is valid is just the first part of the work to do, because the focus is on the execution of the plan. Hence, considering the estimated duration (or makespan) of actions makes much more sense than in other artificial intelligence domains. After carefully studying temporal planning problems, it was discovered that three major generic qualities of plans were: cost, expected makespan, and number of tasks (i.e. number of sequenced actions of a plan). The proposed definition and formalization for each of the qualities is as follows:

Definition 3.1. *Plan Cost is a Quality of a Plan that captures the cost of executing the Plan.*

$$\begin{aligned} PlanCost(q) \equiv dul.Quality(q) \ \ \wedge \\ \exists p \ \ dul.Plan(p) \wedge hasCost(p, q). \end{aligned} \tag{1}$$

Definition 3.2. *Plan Makespan is a Quality of a Plan that captures the expected time that would be required to execute the Plan.*

$$\begin{aligned} PlanMakespan(q) \equiv dul.Quality(q) \ \ \wedge \\ \exists p \ \ dul.Plan(p) \wedge hasMakespan(p, q). \end{aligned} \tag{2}$$

Definition 3.3. *Plan Number Of Tasks is a Quality of a Plan that captures the number of tasks defined in the Plan.*

$$\begin{aligned} PlanNumberOfTasks(q) \equiv dul.Quality(q) \ \ \wedge \\ \exists p \ \ dul.Plan(p) \wedge hasNumberOfTasks(p, q). \end{aligned} \tag{3}$$

The formalization of 'Plan Cost' (Eq. 1) reads as follows: a plan cost is a quality (q) that is the cost of at least one plan (p). The other two formalizations (Eq. 2 and Eq. 3) would read in an analogous manner. The definitions include the notion of 'executing a plan', used here as a primitive which means *'following the sequence of tasks defined in the plan'*. The prefix *'dul'* denotes that a term was re-used from DUL, while novel terms and relations have no prefix. The plan's properties are modeled as qualities (in DUL) that are related to a plan they qualify. New relations between the plans and the qualities were introduced: *'has cost'*, *'has makespan'* and *'has number of tasks'*. Additionally, their inverse relations were also defined: *'is cost of'*, *'is makespan of'*, and *'is number of tasks of'*. These two pairs of three new relations were defined as specializations of the DUL's relations *'has quality'* and *'is quality of'*, respectively.

Of course, one might consider other qualities as relevant (e.g., the expected risk of human injury, the probability of failure/success, the workload percentage among collaborative agents, etc.). We argue that our approach is easy to extend to accommodate the specific details of other applications. This means that our approach can be extended with new qualities by just replicating the same structure that we propose. Consider the case of human-robot collaboration, a new plan's quality might be the robot workload percentage (i.e. the percentage of the total plan's tasks that the robot must do). This can also be defined as Quality and a new relation can be created e.g. hasRobotWorkload(p,w), used to relate a Plan (p) and the workload (w).

3.2.2. How do the characteristics of different plans relate? (CQ2)

The idea here is to be able to model knowledge such as: 'the characteristic Xa of plan Pa is worse than the characteristic Xb of plan Pb'. OCRA does

not provide a formal way to compare the properties of plans. Considering the previously formalized classes, the aimed relations should hold between qualities of plans (e.g. *'PlanCost'*). Looking at the modeled qualities, one notices that all of them are numerical, hence, they could be related with comparative words such as: 'higher', 'lower', etc. However, it would be better to keep the new ontological model as re-usable as possible, for instance, using more generic notions (e.g. worse/better quality). Indeed, qualities between plans can be worse and better, but also equal or equivalent, thus, this should also be modeled. The aim of the new model is to provide the foundations for robots (and agents) to represent and reason about how alternative plans compare. Notions such as 'better' and 'worse' are indeed subjective, but that is exactly why it is relevant to work on their modeling. Hence, we propose a theory that copes with both numerical and categorical qualities.

In total, three new intransitive relations are formalized: *'is better quality than'*, *'is worse quality than'*, *'is equivalent quality to'*. The first two are inversely related, while the third one is symmetric. The three are defined as sub-relationships of the relation *'associated with'* (from DUL). They hold between two qualities, thus, they can be used beyond the scope of this work (e.g. comparing the qualities of robots, drinks, etc.). The proposed ontology is designed to model the outcome of comparing two qualities. However, it falls outside the scope of the model to make any commitment regarding the comparison criteria. Sec. 4.2 proposes some general inference rules for this. Users of the ontology may use them or define their own.

3.2.3. How do different plans relate to each other? (CQ3)

The OWL 2 DL version of OCRA defines the binary relationships *'is better plan than'* and *'is worse plan than'* which relate two plans stating that one of the plans is better or worse to achieve a goal. They were defined in the context of plan adaptations (i.e. events in which an agent decides to adapt an ongoing plan replacing it with a better option). Those two relations might be sufficient for the case of plan adaptations. Nevertheless, one might wonder what would happen in more general cases when two plans have equivalent properties. Neither of the plans would be better or worse than the other, thus, OCRA would fall short of modeling this. Furthermore, OCRA did not make any commitment about how an agent should compare plans and decide which one is better (see Sec. 4.2).

Given the lack of a formalization in OCRA on this matter, this article extends its coverage by formalizing three new sub-relations, one per quality:

'is cheaper plan than', *'is faster plan than'*, *'is shorter plan than'*; and their inverse relations *'is more expensive plan than'*, *'is slower plan than'*, *'is longer plan than'*, respectively. All of them are defined as sub-properties of the relation *'associated with'* (from DUL). Furthermore, it is also added the relation *'is equivalent plan to'*, defined as disjoint with *'is better plan than'* and *'is worse plan than'*. Related to this new relation, other three relations are created as sub-properties of *'associated with'*: *'is plan with same cost as'*, *'is plan with same makespan as'*, *'is plan with same number of tasks as'*. Recall that all these relations hold between two plans, answering CQ3.

3.3. Formalization of the model in OWL 2 DL

For practical use, the proposed ontological theory was formalized in OWL 2 DL. Hence, the axioms presented in this paper were translated to DL, in particular, to the SROIQ(D) fragment. Each of the axioms was translated in the target formalism with the exception of the value of plans' qualities. Note that quality is often used as a synonym for property but not in DUL, where 'qualities are particulars and properties are universals'. In this regard, DUL considers that 'qualities inhere in entities' [19]. Every entity (including qualities) comes with its own exclusive qualities, which exist as long as the entity exists. DUL distinguishes between a quality (the cost of a plan) and its value or quale (a numerical data value). Hence, when saying that two plans have the same cost, their costs have the same quale, but still they are distinct qualities. This is convenient to model and answer CQ2, since OWL 2 DL cannot model relationships between two data values or quales (i.e. one cannot state that 5 is a better cost than 10). However, one can model the relation between the qualities (e.g. *'cost A'* has better quality value than *'cost B'*). Let's consider that a plan 'p' has a cost 'c' whose value is '10'. Hence, the knowledge would be modeled as:

$$PlanCost(c) \wedge dul.Plan(p) \wedge hasCost(p, c) \wedge hasDataValue(c, 10).$$

For consistency, the label of the relations comparing two qualities: *'is better quality than'*, *'is worse quality than'*, *'is equivalent quality to'*; were modified to *'has better quality value than'*, *'has worse quality value than'*, *'has equivalent quality value than'*. Figure 2 shows an overview of the OWL 2 DL formalization of the ontology.

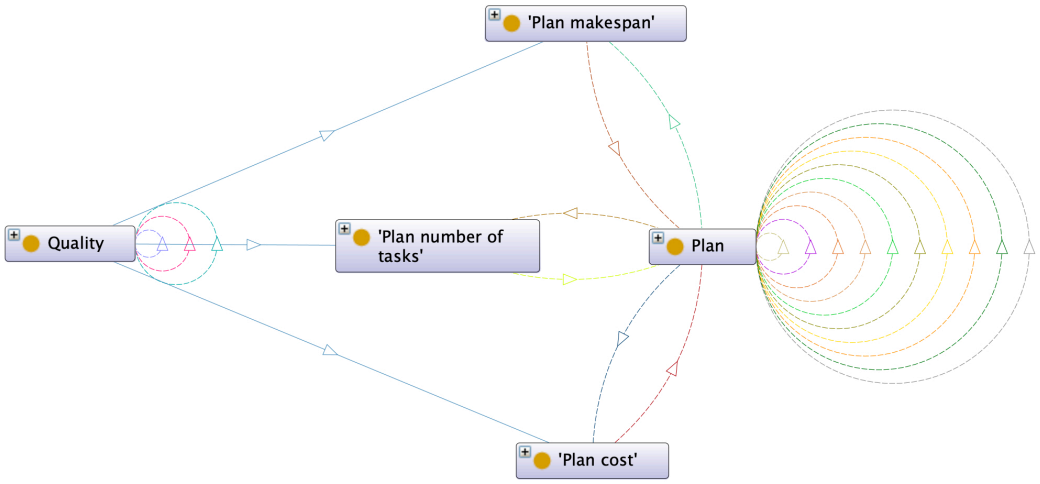

Figure 2: Graph visualization of the proposed ontological model. The defined plan's properties are subclasses of Quality, indicated by the only continuous arrows. The graph depicts the relations between plans and their properties (e.g., *'has cost'*, *'is cost of'*). It also shows multiple relations holding between plans (e.g. *'is cheaper plan than'*), and three relations between qualities (e.g. *'has better quality value than'*).

3.4. Modeling the tasks plans using DUL

The sequence of tasks described in a plan is one of the useful aspects to compare plans. Therefore, modeling such knowledge would allow its use to narrate the differences between plans. The foundational ontology DUL already covers this knowledge, thus, there is no need for a new extension. As an example, let's imagine that there is a plan 'p' that consists in executing three tasks in the following order, 't1', 't2', and 't3'. This knowledge might be represented as follows:

$$\begin{gathered} dul.Task(t1) \wedge dul.Task(t2) \wedge dul.Task(t3) \ \wedge \\ dul.Plan(p) \wedge dul.definesTask(p, t1) \ \wedge \\ dul.definesTask(p, t2) \wedge dul.definesTask(p, t3) \ \wedge \\ dul.directlyFollows(t2, t1) \wedge dul.directlyFollows(t3, t2). \end{gathered}$$

4. The theory at work

The validation of an ontological model consists in creating instances of the concepts and showing their use. In this regard, the knowledge about

instantiated plans (e.g., sequence of tasks, and qualities) would first be asserted to a knowledge base. Then, reasoning rules would be used to contrast the plans and infer which one is better. This section discusses the process of instantiating the model, and also introduces the reasoning rules used for contrasting the plans, and their implementation. Finally, it shows how the model is able to answer the competency questions, validating it.

4.1. Instantiating the ontology with plans

The aim here is to demonstrate how to instantiate the model in a realistic scenario, providing as many resources as possible to potential users of the model. For this, the assertion of the knowledge about the plans was integrated with the planning system of the robot. Therefore, when a robot generates a plan by means of automated planning, it may also assert the knowledge about the plan, both its sequence and qualities. As a technical contribution, it was developed a novel knowledge-based framework that integrates existing robot planning tools and the use of the proposed ontology to model the comparison of plans (*know-plan*). This implementation is publicly available on a Github repository,[2] and illustrates how to instantiate the ontology with automatically generated plans. Planning is done using ROSPlan [20], a commonly used framework for planning in robotics. The ROSPlan framework provides a collection of tools for AI Planning for robots equipped with the Robot Operating System (ROS). Once a plan is found, the sequence of tasks and the plan's qualities are asserted to a knowledge base. As an example, a planning domain inspired by the scenario depicted in Fig. 1 is used, where a robot delivers drinks at an office/apartment.[3] In order to have two plans to compare, the process is executed twice with different planning problems in which the type and location of the drink that the robot should bring change: a tea from the microwave or a cola from the fridge.[4] Since the tea and the coke are in different places, the planner finds plans with different costs, makespan and number of tasks (i.e., the cost of navigating to the coke is higher than to the tea, as it is further away). A summary of the asserted knowledge of the plans' qualities is presented in Tab. 1.

[2] https://github.com/albertoOA/know_plan

[3] https://github.com/albertoOA/know_plan/blob/main/pddl/domain/apartment_domain.pddl

[4] https://github.com/albertoOA/know_plan/blob/main/pddl/problem/apartment_problem.pddl

Table 1: Knowledge from the example of bringing drinks

Plan	*Nº of Tasks*	*Cost*	*Makespan (s)*
'bringing tea'	6	27	27
'bringing cola'	8	59	59

4.2. Reasoning for plan comparison

The proposed ontological theory allows agents to represent the fact that a plan is better, worse or equally good than another one. However, none of the proposed axioms automates the comparison of plans and thus the inference of which plan is better.

4.2.1. Logical rules to infer the relation between plan's qualities

Let's assume that there is a consistent instantiated ontology $\mathcal{O}$ that contains knowledge about the qualities of different plans (P_a, P_b) as a set of triples $\langle subject, relation, object \rangle$ (see Sec. 4.1). The first step would be to compare the qualities' values and infer the relation between them (e.g. the cost of 'bringing cola' has a worse value than the cost of 'bringing tea'). Additionally, it can also be inferred the relation between the plans based on how the qualities relate (e.g. a plan is cheaper than another one). For instance, given the two plans (P_a, P_b), one can obtain their cost (C_a, C_b) and their cost' values (V_a, V_b) such that:

$$hasCost(P_a, C_a) \wedge hasCost(P_a, C_b) \wedge$$
$$dul.hasDataValue(C_a, V_a) \wedge dul.hasDataValue(C_b, V_b).$$

Then, if the values are equal ($V_a = V_b$) two new triples would be added to the knowledge base indicating that both costs have an equivalent quality value and that both plans have the same cost:

$$\mathcal{O} \leftarrow \mathcal{O} \cup \langle Ca, hasEquivalentQualityValueThan, Cb \rangle;$$
$$\mathcal{O} \leftarrow \mathcal{O} \cup \langle Pa, isPlanWithSameCostAs, Pb \rangle.$$

Similarly, when the values are different the asserted knowledge would refer to whether one of the costs/plans is worse/better than the other. Note that the cost's value is numerical, and it is usually assumed that the smaller it is, the better. Hence, when $V_a > V_b$:

$$\mathcal{O} \leftarrow \mathcal{O} \cup \langle Ca, hasWorseQualityValueThan, Cb \rangle;$$
$$\mathcal{O} \leftarrow \mathcal{O} \cup \langle Pa, isMoreExpensivePlanThan, Pb \rangle.$$

The final case would be when $V_a < V_b$, which would equally result in a triples' assertion of the inferred knowledge. As a whole, the complete described process becomes a logical rule to compare two plans' cost, which infers the relations between them and how this relation affects the connection between the plans. For the other qualities of plans formalized in the proposed model (makespan and number of tasks), analogous rules were defined. The criteria for the comparison were the same, since the rest of the formalized qualities are numerical and for all of them, the lower their value is, the better.

4.2.2. Logical rule to infer which plan is better

Having inferred how the qualities and the plans relate, the next step is to infer which plan is better. Here, the criterion to decide if a plan is better than another one is satisfied when all the relations holding between their qualities indicate that the plan has better qualities. This is, if every quality of a plan (Pa) *'has better quality value than'* the same type of quality of a second plan (Pb):

$$\forall Qa, Qb \;\; \exists Pa, Pb, R \;\; \langle Qb, rdf.type, dul.Quality \rangle \wedge$$
$$\langle Qa, rdf.type, dul.Quality \rangle \wedge \langle Pa, R, Qa \rangle \wedge \langle Pb, R, Qb \rangle \wedge$$
$$\langle Qa, hasBetterQualityValueThan, Qb \rangle$$

The previous logical expression reads as follows: all pairs of qualities (Qa, Qb) related to a pair of plans (Pa, Pb) through the same property R, in which Qa *'has better quality value than'* Qb. The relation *'rdf:type'*, which relates an individual with its class, is a standard relation from the Resource Description Framework (RDF) [21]. When the previous logical expression is true, then we can say that Pa *'is better plan than'* Pb, and the knowledge indicating it would be asserted:

$$\mathcal{O} \leftarrow \mathcal{O} \cup \langle Pa, ocra.isBetterPlanThan, Pb \rangle.$$

Hence, the chosen criterion to infer whether a plan is better than another plan is that all their properties shall be better. It is analogously defined for the cases of worse and equivalent plans, which as a whole would be the logical rule to infer, between two plans, which one is better. Note that in situations with conflicting or ambiguous properties, the logical rules will not assert new knowledge, thus it will not be specified whether one of the plans is equivalent

to, better, or worse than the other one. For instance, if the cost of a plan has a better value than the cost of another plan, but in the rest of qualities the second plan is better, then none of the plans is inferred as better, neither equivalent nor worse. Note that the criterion is used here as an example, final users of the ontology might define different criteria. For instance, the effect of the different qualities might have a different weight (e.g. it may be more important to have a cheaper plan than a shorter one).

4.3. Implementation of the inference rules

Inherited from *know-cra*, this work uses Knowrob [22, 23], a general framework for knowledge representation and reasoning for robots. The framework allows to read OWL-based ontologies and load them into a knowledge base that is built using Prolog [24]. Since the knowledge base is accessible through a prolog-based interface, it is possible to use the logical reasoning power of Prolog to make inferences. Therefore, the inference rules introduced in Sec. 4.2 can be implemented in Prolog. This would integrate the decision-making process to compare plans into the knowledge base, augmenting the reasoning capabilities of the proposed ontological model. Note that the implementation of *know-plan* also introduced some extra Prolog predicates to automate the call of the different rules. Specifically, it was implemented a predicate that runs all the rules for all the pairs of different plans stored in the knowledge base. The rules imply (binary) comparisons between pairs of qualities, and their complexity is linear with respect to the number of qualities, which would be added to the ontology (OWL 2 DL) complexity.

4.4. Answering the competency questions

The example of a robot delivering drinks (see Sec. 4.1) was used to showcase how to answer the competency questions. Note that the answers will contain the instantiated knowledge shown in Tab. 1, plus inferred knowledge after applying the implemented rules. The queries are presented in prolog-like syntax (e.g. containing unbounded variables), since the knowledge base is written in Prolog.

4.4.1. Which are the characteristics of a plan? (CQ1)

This competency question can be translated into a query in prolog-like syntax:

$$triple('bringing\ \ tea', dul.hasQuality, Q),\ triple(Q, dul.hasDataValue, V).$$

If the query holds, i.e. the knowledge base contains the query triples, the answer contains an assignment of all the possible combinations of values of Q and V that make the query to be 'true'. Some examples of answers would be: 'cost of bringing tea' (Q) and '27' (V), or 'number of tasks of bringing tea' (Q) and '6' (V).

4.4.2. How do the characteristics of different plans relate? (CQ2)

$$triple('bringing\ \ tea', dul.hasQuality, Qa),$$
$$triple('bringing\ \ cola', dul.hasQuality, Qb),\ triple(Qa, R, Qb).$$

The answer to the query would contain an assignment of all the possible combinations of values of Qa, R and Qb. For instance, 'cost of bringing tea' (Qa), 'has better quality value than' (R), and 'cost of bringing cola' (Qb). Note that this can only be answered after some of the inference rules have been applied (see Sec.4.2.1).

4.4.3. How do different plans relate to each other? (CQ3)

$$triple('bringing\ \ tea', R, 'bringing\ \ cola').$$

The answer would contain all the assignments to R that make the query to be 'true' in the knowledge base. In total, R could take four values: 'is cheaper plan than', 'is shorter plan than', 'is faster plan than', and 'is better plan than'. In order to answer this competency question, all the inference rules should have been applied.

5. Contrastive narratives of plans

In the literature, Miller [25] discusses that the content of explanations is selected from all available knowledge and following some, often biased, criteria. One might argue that such a selection may be hard coded in our case, selecting a set of queries (e.g. the competency questions) to interrogate the knowledge base about how two plans compare. Then, the retrieved knowledge could be presented in a template (e.g. using a table or textual format). However, if the qualities change (e.g. new qualities are modeled), then the selection must be hard coded again. In this section, the objective is to discover a more flexible and advanced strategy to perform knowledge selection, retrieval, and compilation into contrastive explanatory narratives.

5.1. May explanatory narratives do the work?

The ontological model proposed in this work augments the knowledge coverage in *know-cra* to model the divergences between plans (see Sec. 3), and to automate the inference of whether those differences make one plan better than others (see Sec. 4). Hence, one might think that using the new model together with the XONCRA methodology [4] would be enough to narrate what robots know about competing plans. In particular, given the knowledge about two plan instances and how they relate, XONCRA could produce a narrative about each of the plans using the AXON algorithm. The two narratives together would include the relevant knowledge for a robot to infer which plan is better, thus, humans could read the narratives and understand the inference. The differences between the two narratives could even be highlighted, as others have done when contrastively explaining the traces of two plans [26]. However, such an approach would still require humans to extract their own conclusions by reading the complete narratives. Therefore, while being a potential baseline solution, the narratives generated by AXON do not seem to be optimized for the cases that concern this article, and a better approach might be developed.

5.2. Beyond plain explanatory narratives

Miller [25] stated that explanations are *contrastive*, *selected*, and *social*. Contrastive because they are sought in response to counterfactual cases that open questions such as: why a plan is better instead of others. Explanations are selected as they usually contain just part of the reasons, extracted by agents from a larger knowledge and based on specific criteria. Finally, explanations transfer knowledge in a conversational format, being part of a social interaction between agents. These three aspects of explanations set the basis to design a better algorithm, an alternative to AXON that:

- constructs contrastive narratives instead of plain narratives;
- enhances the selection of knowledge, extracting only the differences between the compared plans; and
- reduces the needed time to communicate a narrative, which might boost the (social) interaction.

Naturally, the new algorithm shall preserve AXON's key advantages: being agnostic to the exact ontological objects and relationships, automating

the selection and retrieval of relevant knowledge based on knowledge graph vicinity. Hence, the algorithm will remain applicable when new plan qualities are defined, and it will generalize beyond plan comparison (e.g. comparing two drinks, two tasks, etc.).

5.3. Preliminary notation

Let's assume countable pairwise disjoint sets N_C, N_P, and N_I of class names, property names, and individuals, respectively. The standard relation *'rdf:type'*, which relates an individual with its class, is abbreviated as *'type'* and included in N_P. A knowledge graph $\mathcal{G}$ is a finite set of triples of the form $\langle s, p, o\rangle$ (subject, property, object), where $s \in N_I$, $p \in N_P$, $o \in N_I$ if $p \neq type$, and $o \in N_C$ otherwise. In this work, the knowledge base works as an episodic memory [23, 4], thus it allows the assertion of triples with the time interval in which they hold. Hence, the stored knowledge can be seen as a time-indexed knowledge graph $\mathcal{G}_\mathcal{T}$, which is a finite set of tuples of the form $\langle s, p, o, t_i, t_f\rangle$, where $t_i, t_f \in \mathbb{R} > 0$, and denote the time interval (initial and final time) in which the triple $\langle s, p, o\rangle$ holds. Note that non-asserted knowledge is considered unknown and never false, since knowledge graphs usually comply with the open-world assumption. For this, the Web Ontology Language 2 (OWL 2) was developed to allow the explicit negative assertion of properties: $\langle s, p, o\rangle$ is $false$. Hence, $\mathcal{G}_\mathcal{T}$ may contain, e.g., that during an interval of time, $\langle t_i, t_f\rangle$, a task k is not defined in a plan p: $\langle k, dul : isDefinedIn, p, t_i, t_f\rangle$ is $false$. In this work, querying the $\mathcal{G}_\mathcal{T}$, we build what we called 'contrastive narrative tuples' of a pair of instance plans, $\mathcal{T}_\mathcal{P}$: $\langle s, p, o, t_i, t_f, sign\rangle$, where *sign* indicates whether the time-indexed triple comes from a positive or negative assertion.

5.4. ACXON - An algorithm for contrastive explanatory ontology-based narratives

ACXON is a novel theoretical contribution, an algorithm that retrieves knowledge about divergences between ontological entities (e.g. plans), for later construction of textual contrastive explanatory narratives. The algorithm takes as an input a time-indexed knowledge graph $\mathcal{G}_\mathcal{T}$ (as described in Sec. 5.3), the ontological class (or classes) of the pair of instances to narrate, the temporal locality (time interval of interest), and the level of specificity. Our focus is on contrastive narratives about *Plans*, but ACXON is general enough to work with other OWL 2 DL ontologies and classes. For instance, it might contrastively narrate the capabilities of a pair of agents (e.g. one can move for longer periods), or how two drinks are different to each other

Algorithm 1: ACXON

Input: Time-indexed knowledge graph ($\mathcal{G}_\mathcal{T}$), pairs to narrate ($\mathcal{P}$), temporal locality (L_i, L_f), specificity (S)
Output: Contrastive Narratives ($\mathcal{E}$)

1 $\mathcal{E} \leftarrow \emptyset$
2 $\mathcal{I}_{\mathcal{P}_\mathcal{T}} \leftarrow$ RetrieveInstantiatedPairsWithTime($\mathcal{G}_\mathcal{T}$, $\mathcal{P}, L_i, L_f$)
3 **foreach** $\langle\langle e_a, t_{i_a}, t_{f_a}\rangle, \langle e_b, t_{i_b}, t_{f_b}\rangle\rangle \in \mathcal{I}_{\mathcal{P}_\mathcal{T}}$ **do**
4 $\quad \mathcal{I}_\mathcal{P} \leftarrow \langle\langle e_a, t_{i_a}, t_{f_a}\rangle, \langle e_b, t_{i_b}, t_{f_b}\rangle\rangle$
5 $\quad \mathcal{T}_\mathcal{P} \leftarrow$ RetrieveNarrativeTuples($\mathcal{G}_\mathcal{T}, \mathcal{I}_\mathcal{P}, S$)
6 $\quad \mathcal{D}_\mathcal{P} \leftarrow$ ExtractDivergentNarrativeTuples($\mathcal{T}_\mathcal{P}$)
7 $\quad \mathcal{E}_\mathcal{P} \leftarrow$ ConstructContrastiveNarrative($\mathcal{D}_\mathcal{P}$)
8 $\quad \mathcal{E} \leftarrow \mathcal{E} \cup \mathcal{E}_\mathcal{P}$
9 **end**

(e.g. one is healthier, tastier, etc.). Based on the level of specificity, there are three types of contrastive narratives. In this article, specificity refers to the amount of detail to be used during the narrative construction, i.e., the number of knowledge tuples.

ACXON first retrieves a set $\mathcal{I}_{\mathcal{P}_\mathcal{T}}$ of sets comprising the instantiated pairs of the provided pair of classes $\mathcal{P}$, which exist, at least partially, within the temporal locality $\langle L_i, L_f\rangle$ (line 2). Each $\mathcal{I}_\mathcal{P}$ is a set of tuples $\langle e, t_i, t_f\rangle$, containing the two pair's instances e with their time interval (t_i, t_f) (line 4). Second, for each of the instantiated pairs $\langle\langle e_a, t_{i_a}, t_{f_a}\rangle, \langle e_b, t_{i_b}, t_{f_b}\rangle\rangle \in \mathcal{I}_{\mathcal{P}_\mathcal{T}}$, a set of knowledge tuples $\mathcal{T}_\mathcal{P}$ is retrieved according to the specificity level (line 5). Third, from the initial narrative tuples $\mathcal{T}_\mathcal{P}$, it is selected the sub-set containing only divergent knowledge between the pair of instances $\mathcal{D}_\mathcal{P}$ (line 6). Fourth, for each instantiated pair a contrastive explanation $\mathcal{E}_\mathcal{P}$ is built using their relative tuples (line 7). Finally, the new explanation is added to the set of explanations $\mathcal{E}$ (line 8). The implemented algorithm and examples of its use are available online.[5]

5.4.1. Retrieve instantiated pair with time routine

With a time-indexed knowledge graph $\mathcal{G}_\mathcal{T}$, a pair of ontological existing classes or a set of them, $\mathcal{P} \subset N_C$, and the temporal locality $\langle L_i, L_f\rangle$,

[5] www.iri.upc.edu/groups/perception/#ontology-based-explainable-robots

this routine retrieves a set of sets $\mathcal{I}_{\mathcal{P}_\mathcal{T}}$ comprising all the instantiated pairs of the given pair of classes, $\mathcal{I}_\mathcal{P}$, which contain the two time-indexed instances $\langle\langle e_a, t_{i_a}, t_{f_a}\rangle, \langle e_b, t_{i_b}, t_{f_b}\rangle\rangle$ of each pair such that their time intervals exist within (intersect) the temporal locality:

$$\forall\langle\langle e_a, t_{i_a}, t_{f_a}\rangle, \langle e_b, t_{i_b}, t_{f_b}\rangle\rangle \in \mathcal{I}_{\mathcal{P}_\mathcal{T}} \rightarrow \exists\langle c_a, c_b\rangle \in \mathcal{P} \wedge$$
$$\langle e_a, type, c_a, t_{i_a}, t_{f_a}, sign_a\rangle \in \mathcal{G}_\mathcal{T} \wedge (\langle t_{i_a}, t_{f_a}\rangle \cap \langle L_i, L_f\rangle) \wedge$$
$$\langle e_b, type, c_b, t_{i_b}, t_{f_b}, sign_b\rangle \in \mathcal{G}_\mathcal{T} \wedge (\langle t_{i_b}, t_{f_b}\rangle \cap \langle L_i, L_f\rangle)\ .$$

Examples of time-indexed instances of plans to narrate may be:

$\langle\langle bringing\ \ water, 10.0, 150.0\rangle, \langle bringing\ \ juice, 0.0, 150.0\rangle\rangle$;
$\langle\langle bringing\ \ tea, \_, Inf\rangle, \langle bringing\ \ cola, \_, Inf\rangle\rangle$.

Note that the time interval is not always numerical, see the second example. This happens when a triple has held true in the knowledge base since an undetermined instant of time ('_'), and it will remain true as long as the knowledge base stays active ('Inf').

5.4.2. Retrieve narrative tuples routine

For each instantiated pair to narrate $\mathcal{I}_\mathcal{P}$, with the $\mathcal{G}_\mathcal{T}$, and the level of specificity S, the routine would retrieve all the tuples about the pair, $\langle s, p, o, t_i, t_f, sign\rangle$, that are relevant to build the contrastive narrative. The first level of specificity retrieves tuples comprising all the relations p holding between the two instances of a pair: $\exists p \in N_P \wedge (\langle e_a, p, e_b, t_i, t_f, sign\rangle \in \mathcal{G}_\mathcal{T} \vee \langle e_b, p, e_a, t_i, t_f, sign\rangle \in \mathcal{G}_\mathcal{T})$. In the second level, the routine adds all the tuples in which at least one of the instances $\langle e_a, e_b\rangle$ is related through a property p to an object o: $\exists p \in N_P \wedge (\langle e_a, p, o, t_i, t_f, sign\rangle \in \mathcal{G}_\mathcal{T} \vee \langle e_b, p, o, t_i, t_f, sign\rangle \in \mathcal{G}_\mathcal{T})$. Following a similar logic to the first level, the second level also retrieves tuples that relate the different objects o. When the instances $\langle e_a, e_b\rangle$ are related to two objects $\langle o_a, o_b\rangle$ through the same property p, the tuples relating those two objects $\langle o_a, q, o_b, t_i, t_f, sign\rangle$ are added: $\exists p, q \in N_P \wedge \langle o_a, q, o_b, t_i, t_f, sign\rangle \wedge \langle e_a, p, o_a, t_{i_a}, t_{f_a}, sign_a\rangle \in \mathcal{G}_\mathcal{T} \wedge \langle e_b, p, o_b, t_{i_b}, t_{f_b}, sign_b\rangle \in \mathcal{G}_\mathcal{T}$. These are horizontal links in Fig. 3. Then, in the third level the routine adds all the tuples in which the objects o from the second level are related to other objects o_x: $\langle o, p_x, o_x, t_{ix}, t_{fx}, sign_x\rangle \in \mathcal{G}_\mathcal{T} \wedge \langle t_i, t_f\rangle \cap \langle t_{ix}, t_{fx}\rangle$. Robots' experiences may be tied to a time frame, thus, the search was restricted to tuples whose time interval $\langle t_{ix}, t_{fx}\rangle$ intersected the time interval of the pair's instances $\langle t_{i_a}, t_{f_a}\rangle$ and $\langle t_{i_b}, t_{f_b}\rangle$. This prevented the routine from retrieving tuples with irrelevant knowledge about the pair of instances to

narrate $\langle e_a, e_b \rangle$. The third level finishes collecting the tuples relating the different objects o_x between each other, similarly to how it is done in the second level: $\exists p_x, q_x \in N_P \wedge \langle o_{x_a}, q_x, o_{x_b}, t_i, t_f, sign \rangle \wedge \langle o_a, p_x, o_{x_a}, t_{i_a}, t_{f_a}, sign_{x_a} \rangle \in \mathcal{G}_\mathcal{T} \wedge \langle o_b, p_x, o_{x_b}, t_{i_b}, t_{f_b}, sign_{x_b} \rangle \in \mathcal{G}_\mathcal{T}$. For any of the levels, when the narrative's set of tuples $\mathcal{T}_\mathcal{P}$ already contains a tuple or its inverse, the tuple is not added. Furthermore, the tuples are retrieved incrementally from the first to the third level. Hence, when the specificity level is three, the returned tuples also contain those from the first and second levels. This would equate to moving deeper in the knowledge graph representing the instanced pair (see Fig. 3). Utilizing the ongoing example of bringing a drink (see Fig. 1), some instances of the retrieved narrative tuples $\mathcal{T}_\mathcal{P}$ of an instantiated pair are:

$\boldsymbol{\mathcal{T}_{\mathcal{P}1}}$ ⟨'bringing tea', isBetterPlanThan, 'bringing cola', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}2}}$ ⟨'bringing tea', definesTask, 'T2-grasp object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}3}}$ ⟨'bringing tea', definesTask, 'task 0 - find person', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}4}}$ ⟨'bringing cola', definesTask, 'task 0 - find person', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}5}}$ ⟨'T3-go to waypoint', directlyPrecedes, 'T5-give object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}6}}$ ⟨'T7-give object', isTaskDefinedIn, 'bringing cola', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}7}}$ ⟨'bringing tea', definesTask, 'T3-got to waypoint', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}8}}$ ⟨'bringing tea', definesTask, 'T5-give object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}9}}$ ⟨'T3-go to waypoint', directlyFollows, 'T2-grasp object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}10}}$ ⟨'bringing tea', isCheaperPlanThan, 'bringing cola', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}11}}$ ⟨'bringing cola', hasCost, 'cola cost', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}12}}$ ⟨'bringing tea', hasCost, 'tea cost', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}13}}$ ⟨'cola cost', hasDataValue, '59', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}14}}$ ⟨'tea cost', hasDataValue, '27', _, Inf, positive⟩,
$\boldsymbol{\mathcal{T}_{\mathcal{P}15}}$ ⟨'cola cost', hasWorseQualityValueThan, 'tea cost', _, Inf, positive⟩.

5.4.3. Extract divergent narrative tuples

From the initially selected narrative tuples $\mathcal{T}_\mathcal{P}$, the routine would just retrieve the set of knowledge tuples $\mathcal{D}_\mathcal{P}$ that capture divergences between the two pair's instances. The routine identifies the non-divergent tuples that exist in $\mathcal{T}_\mathcal{P}$ and prunes them. A pair of tuples $\langle \langle s_1, p_1, o_1, t_{i_1}, t_{f_1}, sign_1 \rangle, \langle s_2, p_2, o_2, t_{i_2}, t_{f_2}, sign_2 \rangle \rangle \in \mathcal{T}_\mathcal{P}$ will be non-divergent when: $(s_1 \neq s_2) \wedge (p_1 = p_2) \wedge (o_1 = o_2) \wedge (t_{i_1} = t_{i_2}) \wedge (t_{f_1} = t_{f_2}) \wedge (sign_1 = sign_2)$. Note that the routine prunes the whole branch of a non-divergent tuple, which includes tuples in which the shared object ($o_1 = o_2 = o_s$) acts as the subject: $\langle o_s, q_p, o_p, t_i, t_f, sign \rangle$. The process will apply to tuples extracted at any of the specificity levels, as it is depicted in Fig. 3. In the example, after ap-

plying this routine, the tuples $\mathcal{T}_{\mathcal{P}3}$ and $\mathcal{T}_{\mathcal{P}4}$ would be pruned, and the set of remaining tuples would be:

$\boldsymbol{\mathcal{D}_{\mathcal{P}1}}$ ⟨'bringing tea', isBetterPlanThan, 'bringing cola', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}2}}$ ⟨'bringing tea', definesTask, 'T2-grasp object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}3}}$ ⟨'T3-go to waypoint', directlyPrecedes, 'T5-give object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}4}}$ ⟨'T7-give object', isTaskDefinedIn, 'bringing cola', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}5}}$ ⟨'bringing tea', definesTask, 'T3-got to waypoint', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}6}}$ ⟨'bringing tea', definesTask, 'T5-give object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}7}}$ ⟨'T3-go to waypoint', directlyFollows, 'T2-grasp object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}8}}$ ⟨'bringing tea', isCheaperPlanThan, 'bringing cola', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}9}}$ ⟨'bringing cola', hasCost, 'cola cost', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}10}}$ ⟨'bringing tea', hasCost, 'tea cost', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}11}}$ ⟨'cola cost', hasDataValue, '59', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}12}}$ ⟨'tea cost', hasDataValue, '27', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}13}}$ ⟨'cola cost', hasWorseQualityValueThan, 'tea cost', _, Inf, positive⟩.

5.4.4. Construct contrastive narrative routine

Considering the divergent tuples $\mathcal{D}_{\mathcal{P}}$ of a pair of instances $\langle e_a, e_b \rangle$ to narrate, the routine builds the contrastive explanatory narrative applying a set of rules: **casting**, **clustering**, **ordering**, and **grouping**. These rules describe aggregations commonly used by humans in natural language [27].

Casting consists in homogenizing the ontological properties appearing in the tuples. First, ensuring that for all the tuples $\mathcal{D}_{\mathcal{P}}$ that concern any of the pair's instances $\langle e_a, e_b \rangle$ the instances are the tuple's subject. In the ongoing example: $\mathcal{D}_{\mathcal{P}1}$, $\mathcal{D}_{\mathcal{P}2}$, $\mathcal{D}_{\mathcal{P}4}$, $\mathcal{D}_{\mathcal{P}5}$, $\mathcal{D}_{\mathcal{P}6}$, $\mathcal{D}_{\mathcal{P}8}$, $\mathcal{D}_{\mathcal{P}9}$, and $\mathcal{D}_{\mathcal{P}10}$. Hence, when $\mathcal{D}_{\mathcal{P}}$ contains a tuple in which any of the instances e acts as the object, $\langle s, p, e, t_i, t_f, sign \rangle \in \mathcal{D}_{\mathcal{P}}$, the casting rule reverses the tuple to: $\langle e, p^{-1}, s, t_i, t_f, sign \rangle$, where p^{-1} states for the inverse property of p. In the list of tuples from before, the tuple $\mathcal{D}_{\mathcal{P}4}$ that contains the property *'isTaskDefinedIn'* would be reversed using *'definesTask'*. Then, all the tuples concerning any of the two instances, both reversed tuples and those that did not need to be inverted, are added to a new set $\mathcal{D}_{\mathcal{P}Cast}$ of cast tuples. Casting has a second step that involves the tuples not concerning the pair's instances ($\mathcal{D}_{\mathcal{P}3}$, $\mathcal{D}_{\mathcal{P}7}$, $\mathcal{D}_{\mathcal{P}11}$, $\mathcal{D}_{\mathcal{P}12}$, and $\mathcal{D}_{\mathcal{P}13}$). Guaranteeing that the properties of the tuples to add are consistent with those that already exist in the cast tuples. Hence, if it is not the case, the tuple is reversed before adding it to $\mathcal{D}_{\mathcal{P}Cast}$. In the example, $\mathcal{D}_{\mathcal{P}3}$ is added to $\mathcal{D}_{\mathcal{P}Cast}$ (following the order) thus, $\mathcal{D}_{\mathcal{P}7}$ needs to be inverted before added. $\mathcal{D}_{\mathcal{P}11}$, $\mathcal{D}_{\mathcal{P}12}$, $\mathcal{D}_{\mathcal{P}13}$ are just added.

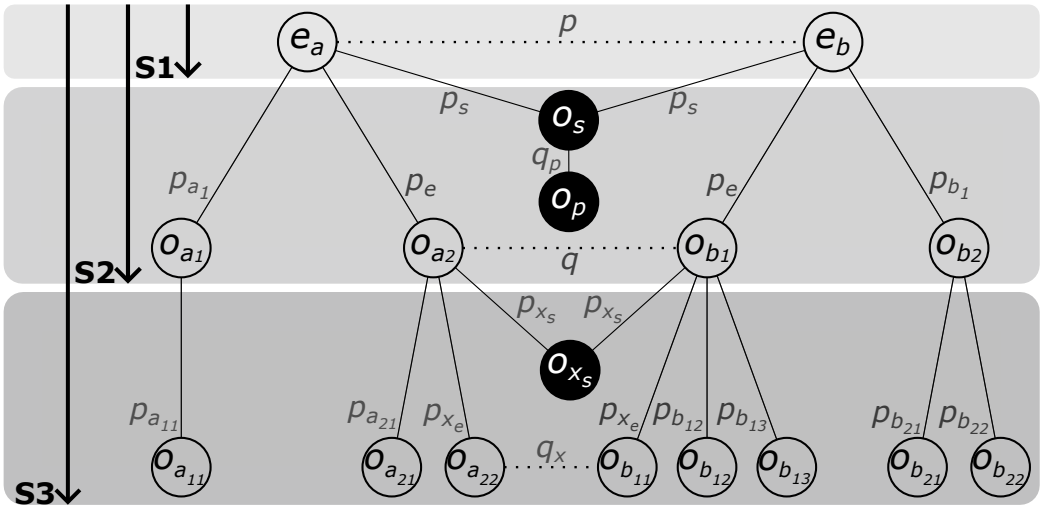

Figure 3: Graphical representation of the different levels of specificity (S1, S2, S3) and their respective depth in the knowledge graph. Nodes in black correspond to initially retrieved tuples that are later pruned because they are non-divergent. S1 contains direct relations between the ontology instances to be compared (e.g. Plan A is cheaper than Plan B). S2 compares how the target instances relate to other objects, and the relations between those objects (e.g. Plan A has a specific cost (Cost A) while Plan B has a different cost (Cost B), or Cost A has better quality value than Cost B). S3 goes deeper in the knowledge graph, comparing the relationships of the objects of the previous level with other ontological objects (e.g. Cost A has data value 27).

Next, this routine applies **clustering**, which structures the tuples in a way that each cluster will later be used to form a single sentence within the whole narrative. The narratives shall contrast the knowledge relating the pair's instances, revealing the divergences between them. Therefore, an effective strategy for cluster generation is to utilize the structure of the knowledge graph formed by the retrieved tuples (see Fig. 3). First, a cluster is created with the tuples $\langle e_a, p, e_b, t_i, t_f, sign\rangle$ that relate the pair's instances $\langle e_a, e_b\rangle$, in the example, $\mathcal{D}_{\mathcal{P}1}$ and $\mathcal{D}_{\mathcal{P}8}$. Second, the routine clusters the remaining tuples by property p in three different steps named: *direct*, *indirect*, *unrelated*. The tuples *directly* related through p to the instances to compare: $\langle e_a, p, o_a, t_i, t_f, sign\rangle$, $\langle e_b, p, o_b, t_i, t_f, sign\rangle$, are clustered together with the tuples relating their objects: $\langle o_a, q, e_b, t_i, t_f, sign\rangle$. Note that when only one of the instances $\langle e_a, e_b\rangle$ is related to an object through p, e.g., $\langle e_a, p, o_a, t_i, t_f, sign\rangle \in \mathcal{D}_{\mathcal{P}} \wedge \langle e_b, p, o_b, t_i, t_f, sign\rangle \notin \mathcal{D}_{\mathcal{P}}$; the knowledge will be clustered later at the *unrelated* step. In the example, $\mathcal{D}_{\mathcal{P}9}$, $\mathcal{D}_{\mathcal{P}10}$,

$\mathcal{D}_{\mathcal{P}13}$ form a cluster, and $\mathcal{D}_{\mathcal{P}2}$, $\mathcal{D}_{\mathcal{P}5}$, $\mathcal{D}_{\mathcal{P}6}$ and the reversed $\mathcal{D}_{\mathcal{P}4}$ another one. New clusters are created for the tuples *indirectly* related to the instances to compare, i.e. those related to the objects $\langle o_a, o_b \rangle$ of the previous step: $\langle o_a, p_x, o_{x_a}, t_{i_a}, t_{f_a}, sign_{x_a} \rangle$, $\langle o_b, p_x, o_{x_b}, t_{i_b}, t_{f_b}, sign_{x_b} \rangle$. As before, those clusters include the tuples holding between their objects $\langle o_{x_a}, o_{x_b} \rangle$. Recall that the tuples are only clustered if the objects $\langle o_a, o_b \rangle$ are each related to one of the main instances to compare. In the example, $\mathcal{D}_{\mathcal{P}11}$ and $\mathcal{D}_{\mathcal{P}12}$ form a cluster. Finally, (*unrelated*) clusters are created with the remaining tuples sharing the same property p, in the example, $\mathcal{D}_{\mathcal{P}3}$, and the inverted $\mathcal{D}_{\mathcal{P}7}$ are clustered.

Subsequently, the clustered tuples are **ordered** externally (between clusters) and internally (between tuples). When externally ordering, the set of clusters is ordered according to the sequence followed during the clustering: first the cluster relating the pair's instances followed by the clusters obtained in the *direct*, *indirect*, and *unrelated* steps. Then, within the clusters from a single step, the clusters are ordered from more knowledge (more tuples) to less. The internal ordering just assures that the tuples with the property $p = type$ are at the front of each of the clusters. In the example, after applying all these rules the set of tuples would now be:

$\boldsymbol{\mathcal{D}_{\mathcal{P}1}}$ ⟨'bringing tea', isBetterPlanThan, 'bringing cola', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}2}}$ ⟨'bringing tea', isCheaperPlanThan, 'bringing cola', _, Inf, positive⟩,

$\boldsymbol{\mathcal{D}_{\mathcal{P}3}}$ ⟨'bringing tea', definesTask, 'T2-grasp object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}4}}$ ⟨'bringing tea', definesTask, 'T3-got to waypoint', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}5}}$ ⟨'bringing tea', definesTask, 'T5-give object', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}6}}$ ⟨'bringing cola', definesTask, 'T7-give object', _, Inf, positive⟩,

$\boldsymbol{\mathcal{D}_{\mathcal{P}7}}$ ⟨'bringing cola', hasCost, 'cola cost', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}8}}$ ⟨'bringing tea', hasCost, 'tea cost', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}9}}$ ⟨'cola cost', hasWorseQualityValueThan, 'tea cost', _, Inf, positive⟩,

$\boldsymbol{\mathcal{D}_{\mathcal{P}10}}$ ⟨'cola cost', hasDataValue, '59', _, Inf, positive⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}11}}$ ⟨'tea cost', hasDataValue, '27', _, Inf, positive⟩,

$\boldsymbol{\mathcal{D}_{\mathcal{P}12}}$ ⟨'T3-go to waypoint', directlyPrecedes, 'T5-give object', _, Inf, pos.⟩,
$\boldsymbol{\mathcal{D}_{\mathcal{P}13}}$ ⟨'T2-grasp object', directlyPrecedes, 'T3-go to waypoint', _, Inf, pos.⟩.

Finally, the tuples of each cluster pass through several **grouping** steps, obtaining the sentences of the final textual contrastive narrative $\mathcal{E}_{\mathcal{P}}$. First, object grouping, the tuples sharing subject, property, interval, and sign are united. Thus, given $\langle s, p, o_a, t_i, t_f, sign \rangle$ and $\langle s, p, o_b, t_i, t_f, sign \rangle$, at this step

the routine unites them to: $\langle s, p, o_a \textit{ and } o_b, t_i, t_f, sign \rangle$. In the example, $\mathcal{D}_{\mathcal{P}3}$, $\mathcal{D}_{\mathcal{P}4}$, and $\mathcal{D}_{\mathcal{P}5}$ are grouped into: ⟨'bringing tea', definesTask, 'T2 - grasp object' and 'T3 - got to waypoint' and 'T5 - give object', _, Inf, positive⟩. The second step consists in grouping tuples by predicate (property), thus tuples sharing subject, object, interval, and sign are joined. Considering two tuples: $\langle s, p_a, o, t_i, t_f, sign \rangle$ and $\langle s, p_b, o, t_i, t_f, sign \rangle$, the routine unites them to: $\langle s, p_a \textit{ and } p_b, o, t_i, t_f, sign \rangle$. In the example, $\mathcal{D}_{\mathcal{P}1}$ and $\mathcal{D}_{\mathcal{P}2}$ would be grouped. The final grouping stage generates textual contrastive sentences for the knowledge stored in each of the clusters of tuples by translating the tuples into text and connecting them. For instance, a cluster may contain the tuples $\langle s, p, o, t_i, t_f, sign \rangle$ and $\langle o, q, o_x, t_{i_x}, t_{f_x}, sign \rangle$. Hence, the knowledge is connected as a subordinate sentence using the pronoun 'which'. In the example, this happens with $\mathcal{D}_{\mathcal{P}7}$ and $\mathcal{D}_{\mathcal{P}9}$. Note that when the subordinate is introduced in tuples that have gone through object grouping, it is also added the phrase 'and also' for readability purposes. The explanations are contrastive, thus the conjunction 'while' is added to emphasize the divergent (contrastive) knowledge: e.g. $\langle e_a, p, o_a, t_{i_a}, t_{f_a}, sign_a \rangle$ and $\langle e_b, p, o_b, t_{i_b}, t_{f_b}, sign_b \rangle$. In the example, 'while' is used to compare the knowledge from the grouped $\mathcal{D}_{\mathcal{P}7}$ and $\mathcal{D}_{\mathcal{P}9}$, and $\mathcal{D}_{\mathcal{P}8}$; and $\mathcal{D}_{\mathcal{P}6}$ and the grouped $\mathcal{D}_{\mathcal{P}3}$, $\mathcal{D}_{\mathcal{P}4}$, $\mathcal{D}_{\mathcal{P}5}$. The the same applies for *indirectly* related clusters such as the one including tuples $\mathcal{D}_{\mathcal{P}10}$ and $\mathcal{D}_{\mathcal{P}11}$. The tuples from *unrelated* clusters, e.g. $\mathcal{D}_{\mathcal{P}12}$ and $\mathcal{D}_{\mathcal{P}13}$, are connected using 'and'. The propositions 'from' and 'to' are also added to introduce the tuples' time intervals, but only if they are different to the interval of the pair's instances. Indeed, if the interval is undetermined (_, Inf), it is obviated. The names of ontological entities (instances, classes and properties) are kept, only some properties are slightly changed to more understandable terms (e.g. using 'includes task' instead of 'definesTask', or 'has a higher value than' instead of 'hasWorseQualityValueThan'). The final narrative for the ongoing example would be:

> 'bringing tea' is better plan than and is cheaper plan than 'bringing cola'. 'bringing tea' includes task 'T2-grasp object' and 'T3-got to waypoint' and 'T5-give object', while 'bringing cola' includes task 'T7-give object'. 'bringing cola' has cost 'cola cost', which has a higher value than 'tea cost'; while 'bringing tea' has cost 'tea cost'. 'cola cost' has value '59', while 'tea cost' has value '27'. 'T3-go to waypoint' directly precedes 'T5-give object', and 'T2-grasp object' directly precedes 'T3-go to waypoint'.

6. Evaluating explanatory narratives

To evaluate the quality of the narratives generated by ACXON, the AXON [4] algorithm was used as a baseline. Specifically, both algorithms were used to narrate the knowledge about contrasting plans. Following the ideas discussed in Sec. 5.1, these two algorithms can be compared by using AXON twice (i.e. to narrate each of the plans independently). By construction, ACXON is expected to reduce the amount of knowledge used in the explanations, and also to ensure shorter explanation communication times.

6.1. Evaluation procedure and setup

The evaluation was done using a set of temporal planning PDDL domains from recent international planning competitions (IPC) [28]. The set included the IPC'02 Rovers domain [28] (10 instantiated problems), the IPC'08 Crew planning domain [29] (15 problems), and the IPC'14 Match cellar domain [30] (20 problems). First, given a domain and two problems, we run a planner with both problems to obtain two plans for which their respective knowledge (sequence and qualities) is instantiated as described in Sec. 4.1. Then, the inference rules are applied, performing the comparison of the two plans and asserting the inferred knowledge (e.g. which plan is better). Finally, both algorithms are used to extract the knowledge from the active knowledge base and construct the explanations. The algorithms have the same inputs: a time-indexed graph (the active knowledge base), a time interval (_, Inf) and the level of specificity (the three levels were used). A set of metrics discussed in Sec. 6.2 are computed for each of the generated narratives. Note that for each of the planning domains, ten pairs of instances were randomly selected without replacement (thirty in total). The software for the test was run on a desktop PC with an Intel Core i7-8700K CPU (12x 3.70 GHz), 16 GB DDR4 RAM, and an NVIDIA GeForce GT 710/PCIe/SSE2 GPU.

6.2. Metrics for explanation evaluation

To evaluate the explanations we have selected a set of offline objective evaluation metrics, aligned with the existing literature. The metrics aim to evaluate two of the main features of explanations: the selection of content (number of attributes), and the social aspect (communication time and readability).

Number of attributes. The metric is commonly used to evaluate explainable models. Especially when evaluating the explainability of black box

models (e.g. machine learning (ML) models) [31]. Nevertheless, this metric has also been used to evaluate non-ML explanatory systems [32]. In this work, the number of attributes is equal to the number of tuples $\mathcal{D}_{\mathcal{P}}$ used to construct the narratives.

Communication time. Explanations are social, thus a good quality index is to measure how much time would require an agent to communicate them. In this work, the communication time is computed as a combination of the *construction time* C_T and the actual *interaction time* I_T. For the interaction, two channels are considered: auditory (I_{TA}) and visual (I_{TV}). For retaining information, people are comfortable with a speaking pace of 150-160 words per minute (wpm), while the pace for silent reading is 250-400 wpm [33]. In this work, the interactions times are estimated by counting the number of words in the narratives and considering the fastest pace for each channel: 160 wpm (auditory), and 400 wpm for (visual).

Readability metric. Since the narratives are generated using natural language, the Dale–Chall readability R_{DC} formula [34], a well-known readability metric, is also used. Most of the readability metrics use a similar formula including two terms: (a) the proportion of 'complex words' relative to the total number of words; and (b) the number of words per sentence. Usually, the word length or number of syllables is used to decide whether a text's words are 'complex' (i.e. difficult to understand). More interestingly, Dale-Chall defines words as 'complex' if they are not familiar (i.e. not included in a list of 3000 common words) [34]. The resulting score indicates the reading level by educational grade needed to comprehend the text. Other readability metrics were considered and tried but results were similar, so we decided to use Dale-Chall because we consider 'familiarity' to be a more interesting indicator of complexity than 'length'.

6.3. Results of the evaluation and discussion

A statistical analysis was conducted to evaluate the significance of the proposed algorithms's improvement in reducing the amount of knowledge used in the explanations, and the explanation communication times, with respect to the baseline. We manipulated the two different algorithms for each specificity level (independent variable) and assessed them with respect to the metrics introduced in Section 6.2 (dependent variables). Normality on the difference between the results obtained by both methods was assessed using the Shapiro-Wilk Test ($\alpha = 0.05$). For the construction time, across all specificity levels, it is not possible to reject the normality assumption ($p >$

0.05), while for the other metrics the normality assumption can be rejected ($p < 0.05$). For the construction time and all three specificity levels, the results of the *paired-t test* indicated that there is a significant large difference between the two algorithms, $p < 0.05$. For the rest of metrics, the results of the *Wilcoxon Signed Rank test* indicated a statistically significant difference in scores between the two algorithms for each of the specificity levels, $p < 0.05$. The average evaluation metric results for the thirty pairs of plans, and each algorithm and level of specificity are summarized in Tab. 2. A summary of the specific statistical analysis results is provided in Tab. 3. More detailed results are available online.[6]

ACXON outperforms the baseline method in most cases, especially for levels two and three of specificity. Regarding the **number of tuples** $\mathcal{D}_{\mathcal{P}}$, using ACXON results in an overall reduction of more than 40% and 70% for levels 2 and 3 respectively. This is because ACXON does a better selection of the narrative tuples. First, avoiding repeated tuples by collecting them for the whole pair instead of individually selecting tuples for each of the instantiated plans (see Sec. 5.4.2). Second, pruning the non-divergent knowledge between the plans (see Sec. 5.4.3). Hence, the contrastive narratives only contain what makes the plans different without undesired repetitions. For the **construction time** C_T, there are no major differences. However, it is worth commenting that the generation for level 3 would even require more than two seconds on average for any of the algorithms. Note that in some cases, the narrated plans contained more than 50 actions, hence, narratives of level 3 were really long. ACXON produces a decrease in the **interaction times** I_{TA} and I_{TV} of approximately the 40% and 70% for specificity 2 and 3, respectively. The average times for the baseline method would be completely prohibitive for realistic interaction with humans. For level 3 of specificity, it would take 13 and 5 minutes for a human to listen and read the narratives, respectively. Indeed, although ACXON reduces those times to 4 and 1.5 minutes, the improvement still falls short of ensuring a fluent and socially acceptable interaction. To overcome this, the robot might provide the short explanation of level 1, and only more details if required. Concerning this, ACXON produces longer times when the specificity is 1. This is because ACXON is more informative than the baseline method, since it includes the relationships between the plans at that level (e.g. shorter, better plan, etc.).

[6] www.iri.upc.edu/groups/perception/ontology-based-explainable-robots

Table 2: Average evaluation results for each metric and the the 30 pairs of plans. The differences are statistically significant.

Method	**Baseline** (AXON)			**ACXON**			**p<0.05**		
Specificity	1	2	3	1	2	3	1	2	3
$\mathcal{D}_{\mathcal{P}}$	**2.00**	77.40	305.73	3.80	**44.27**	**91.33**	✓	✓	✓
C_T *(s)*	0.73	0.87	3.50	**0.56**	**0.75**	**2.53**	✓	✓	✓
I_{TA} *(s)*	**5.25**	170.55	782.80	8.68	**101.00**	**258.85**	✓	✓	✓
I_{TV} *(s)*	**2.10**	68.22	313.12	3.47	**40.40**	**103.54**	✓	✓	✓
R_{DC}	9.62	**8.86**	**2.81**	**8.78**	9.24	7.88	✓	✓	✓

Table 3: Statistical evaluation results for each metric and specificity level. The specific statistical test metrics for each case is denoted between square brackets.

Specificity	1	2	3
$\mathcal{D}_{\mathcal{P}}$ $[W, p, r]$	$[0, < .001, 1.1]$	$[0, < .001, -1.1]$	$[0, < .001, -1.1]$
C_T $[t(29), p]$	$[84.9, < .001]$	$[36.5, < .001]$	$[21, < .001]$
I_{TA} $[W, p, r]$	$[0, < .001, 1.1]$	$[0, < .001, -1.1]$	$[0, < .001, -1.1]$
I_{TV} $[W, p, r]$	$[0, < .001, 1.1]$	$[0, < .001, -1.1]$	$[0, < .001, -1.1]$
R_{DC} $[W, p, r]$	$[0, < .001, -0.9]$	$[0, .046, 0.4]$	$[0, < .001, 1.1]$

Meanwhile, the baseline algorithm only states that both plans are instances of the class *'Plan'*, refer to Olivares-Alarcos et al. [4] for more details about the baseline. Finally, in relation to the **readability index** R_{DC} both behave similarly with a metric value close to 9, denoting a high portion of complex words. Specifically, such a value indicates that the explanations would be easily understood by an average college student. Hence, people with a lower education level may require a higher effort to interpret the narratives. Interestingly, the baseline method obtains a better value for specificity 3, because its narratives contain multiple short sentences, which is favored in the metric. In ACXON, since the narratives are contrastive, they contain several subordinate sentences, which results in a higher degree of complexity.

7. May explanations be more selective?

The evaluation results demonstrated that ACXON enhances the selection of knowledge for explanatory contrastive narratives with respect to the baseline. However, it still requires long communication times for specificity

levels 2 and 3, thus it shall be more selective. For instance, one might use the structure of the knowledge to constrain the tuples retrieval explained in Sec. 5.4.2, focusing only on part of the contrastive knowledge (e.g. only the plans' qualities).

Following this rationale, it is proposed here a modification to the *retrieve narrative tuples* routine to focus on specific aspects of plans. Specifically, at the second level of specificity, when the routine adds all the tuples in which at least one of the instances $\langle e_a, e_b \rangle$ is related through a property p to an object o. Instead of considering tuples containing any object, the ontological class c of the object is restricted: $\exists p \in N_P \wedge \exists c \in N_C \wedge (\langle e_a, p, o, t_i, t_f, sign \rangle \in \mathcal{G}_\mathcal{T} \vee \langle e_b, p, o, t_i, t_f, sign \rangle \in \mathcal{G}_\mathcal{T}) \wedge \langle o, type, c, t_i, t_f, sign \rangle$. This minor modification reduces the number of tuples retrieved at level 2, but its effect is also propagated to level 3, producing a larger decrease in the final number. Let's imagine that in the ongoing example from before, the algorithm is asked to compare the plans only using the qualities of plans (i.e. instances of *dul.Quality*), the retrieved tuples would be reduced from 15 to 7:

$\mathcal{T}_{\mathcal{P}_1}$ ⟨'bringing tea', isBetterPlanThan, 'bringing cola', _, Inf, positive⟩,
$\mathcal{T}_{\mathcal{P}_2}$ ⟨'bringing tea', isCheaperPlanThan, 'bringing cola', _, Inf, positive⟩,
$\mathcal{T}_{\mathcal{P}_3}$ ⟨'bringing cola', hasCost, 'cola cost', _, Inf, positive⟩,
$\mathcal{T}_{\mathcal{P}_4}$ ⟨'bringing tea', hasCost, 'tea cost', _, Inf, positive⟩,
$\mathcal{T}_{\mathcal{P}_5}$ ⟨'cola cost', hasDataValue, '59', _, Inf, positive⟩,
$\mathcal{T}_{\mathcal{P}_6}$ ⟨'tea cost', hasDataValue, '27', _, Inf, positive⟩,
$\mathcal{T}_{\mathcal{P}_7}$ ⟨'cola cost', hasWorseQualityValueThan, 'tea cost', _, Inf, positive⟩.

The rest of algorithm's routines would be applied as it was shown before, producing a shorter narrative:

'bringing tea' is better plan than and is cheaper plan than 'bringing cola'. 'bringing cola' has cost 'cola cost', which has a higher value than 'tea cost'; while 'bringing tea' has cost 'tea cost'. 'cola cost' has value '59', while 'tea cost' has value '27'.

With this modification ACXON does, by construction, a more selective knowledge retrieval to build the contrastive narrative, which would shorten the explanations and reduce the communication time. Furthermore, the algorithm now allows to constrain the preferred content of the contrastive narratives, which might be used to provide personalized explanations. Hence, the modification, while being simple, it contributes to generate potentially more socially acceptable explanations.

Note that there is a trade-off between brevity of an explanation and its informativeness, and this trade-off is conditioned by the preferences of the

final user. Shortening explanations too much risks omitting relevant knowledge, which can lead to poor perceived quality, so how the selection is done is essential. From social sciences, we know that explanations are often selected in a biased way [25], so having the ability to be more selective is desirable. However, this makes the objective evaluation of the modified algorithm less insightful than in Section 6. To assess how a more selective strategy affects this balance, it would be sensible to conduct a user study with different selective explanations, as we plan to do in future work. Additionally, we aim to explore how user preferences might be identified or learned, through interaction, so that the algorithm can tailor the selection process accordingly. In cases where preferences are unknown, providing longer explanations might help prevent oversimplification.

8. Conclusion

This work presents a method for robots to model and reason about the differences between plans, to infer which one is better and to narrate the inferences to other agents (e.g. humans). The approach comprises a novel ontological model for robots to describe plans and their qualities for reasoning during plans comparison, and a new algorithm to construct ontology-based contrastive explanatory narratives. The approach is general to be used with other ontologies, beyond the case of contrasting plans (e.g. modeling the qualities of two drinks and narrating the differences). The model is validated by instantiating it to answer a set of competency questions. The algorithm is evaluated against a baseline with respect to a set of objective metrics. Our solution outperforms the baseline in general, doing a better selection of knowledge tuples to build the explanation (avoiding non-divergent knowledge), and producing explanations that would require less time for the robot to communicate them. It is also discussed a final refinement of the algorithm to be more selective, shortening the narratives and opening the door for more personalized explanations. In the future, we will look at making more accessible the narratives' language, and a user study will be conducted to evaluate their quality. Note that robots add the possibility of physically executing the plan, which opens issues to investigate: the 'preferred' moment to explain (e.g. before or after executing), or the context in which the competing plans are conceived (e.g. due to an adaptation while executing).

Acknowledgments

This work was partially supported by Project PID2023-152259OB-I00 funded by MCIN/ AEI /10.13039/501100011033; Project PCI2020-120718-2 funded by MCIN/ AEI /10.13039/501100011033 and by the "European Union NextGenerationEU/PRTR"; Project EP/V062506/1 funded by EPSRC (UK); and also by the European Union under the project ARISE (HORIZON-CL4-2023-DIGITAL-EMERGING-01-101135959). G. Canal was supported by the Royal Academy of Engineering and the Office of the Chief Science Adviser for National Security under the UK Intelligence Community Postdoctoral Research Fellowship programme. A. Olivares-Alarcos was supported by the European Social Fund and the Ministry of Business and Knowledge of Catalonia through the FI 2020 grant.

References

[1] S. Izquierdo-Badiola, G. Canal, C. Rizzo, G. Alenyà, PlanCollabNL: Leveraging Large Language Models for Adaptive Plan Generation in Human-Robot Collaboration, in: IEEE International Conference on Robotics and Automation (ICRA), 2024, pp. 17344–17350.

[2] L. Yuan, X. Gao, Z. Zheng, M. Edmonds, Y. N. Wu, F. Rossano, H. Lu, Y. Zhu, S.-C. Zhu, In situ bidirectional human-robot value alignment, Science Robotics 7 (68) (2022) eabm4183.

[3] A. Olivares-Alarcos, S. Foix, S. Borgo, Guillem Alenyà, Ocra – an ontology for collaborative robotics and adaptation, Computers in Industry 138 (2022) 103627.

[4] A. Olivares-Alarcos, A. Andriella, S. Foix, G. Alenyà, Robot explanatory narratives of collaborative and adaptive experiences, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2023, pp. 11964–11971.

[5] A. Olivares-Alarcos, S. Foix, J. Borràs, G. Canal, G. Alenyà, Ontological modeling and reasoning for comparison and contrastive narration of robot plans, in: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2024, p. 2405–2407.

[6] A. Olivares-Alarcos, D. Beßler, A. Khamis, P. Goncalves, M. K. Habib, J. Bermejo-Alonso, M. Barreto, M. Diab, J. Rosell, J. Quintas, J. Olszewska, H. Nakawala, E. Pignaton, A. Gyrard, S. Borgo, G. Alenyà, M. Beetz, H. Li,

A review and comparison of ontology-based approaches to robot autonomy, The Knowledge Engineering Review 34 (2019) e29.

[7] J. Bermejo-Alonso, Reviewing task and planning ontologies: An ontology engineering process, in: International Conference on Knowledge Engineering and Ontology Development (KEOD), 2018, pp. 181–188.

[8] K. Moulouel, A. Chibani, Y. Amirat, Ontology-based hybrid commonsense reasoning framework for handling context abnormalities in uncertain and partially observable environments, Information Sciences 631 (2023) 468–486.

[9] P. Langley, B. Meadows, M. Sridharan, D. Choi, Explainable agency for intelligent autonomous systems, in: Proceedings of the AAAI Conference on Artificial Intelligence, AAAI Press, 2017, p. 4762–4763.

[10] S. Rosenthal, S. P. Selvaraj, M. M. Veloso, Verbalization: Narration of autonomous robot experience., in: IJCAI, Vol. 16, 2016, pp. 862–868.

[11] J. G. Flores, I. Meza, É. Colin, C. Gardent, A. Gangemi, L. A. Pineda, Robot experience stories: First person generation of robotic task narratives in sitlog, Journal of Intelligent & Fuzzy Systems 34 (2018) 3291–3300, 5.

[12] G. Canal, S. Krivić, P. Luff, A. Coles, PlanVerb: Domain-Independent Verbalization and Summary of Task Plans, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 9698–9706.

[13] M. Sridharan, Integrated knowledge-based reasoning and data-driven learning for explainable agency in robotics, in: D. Aha, S. Tulli (Eds.), Explainable Agency in Artificial Intelligence: Research and Practice, CRC Press, 2023, p. to appear.

[14] T. Love, A. Andriella, G. Alenyà, What would i do if...? promoting understanding in hri through real-time explanations in the wild, in: 2024 33rd IEEE International Conference on Robot and Human Interactive Communication (ROMAN), 2024, pp. 504–509.

[15] M. Fernández-López, A. Gómez-Pérez, N. Juristo, Methontology: from ontological art towards ontological engineering, Technical Report SS-97-06, AAAI Press, Menlo Park, California, p. 34–40 (1997).

[16] P. Spyns, Y. Tang, R. Meersman, An ontology engineering methodology for dogma, Applied Ontology 3 (1-2) (2008) 13–39.

[17] S. Borgo, R. Ferrario, A. Gangemi, N. Guarino, C. Masolo, D. Porello, E. M. Sanfilippo, L. Vieu, Dolce: A descriptive ontology for linguistic and cognitive engineering, Applied Ontology 17 (1) (2022) 45–69.

[18] M. Fox, D. Long, Pddl2. 1: An extension to pddl for expressing temporal planning domains, Journal of Artificial Intelligence Research 20 (2003) 61–124.

[19] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, Sweetening wordnet with dolce, AI Magazine 24 (3) (2003) 13.

[20] M. Cashmore, M. Fox, D. Long, D. Magazzeni, B. Ridder, A. Carrera, N. Palomeras, N. Hurtos, M. Carreras, Rosplan: Planning in the robot operating system, in: Proceedings of the international conference on automated planning and scheduling, Vol. 25, 2015, pp. 333–341.

[21] B. McBride, The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 51–65.

[22] M. Tenorth, M. Beetz, Knowrob – knowledge processing for autonomous personal robots, in: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, IEEE, 2009, pp. 4261–4266.

[23] M. Beetz, D. Beßler, A. Haidu, M. Pomarlan, A. K. Bozcuoğlu, G. Bartels, Knowrob 2.0 — a 2nd generation knowledge processing framework for cognition-enabled robotic agents, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 512–519. doi:10.1109/ICRA.2018.8460964.

[24] W. F. Clocksin, C. S. Mellish, Programming in Prolog: Using the ISO standard, Springer-Verlag Berlin Heidelberg, 2012.

[25] T. Miller, Explanation in artificial intelligence: Insights from the social sciences, Artificial Intelligence 267 (2019) 1–38.

[26] B. Krarup, S. Krivic, D. Magazzeni, D. Long, M. Cashmore, D. E. Smith, Contrastive explanations of plans through model restrictions, Journal of Artificial Intelligence Research 72 (2021) 533–612.

[27] H. Dalianis, E. Hovy, Aggregation in natural language generation, in: G. Adorni, M. Zock (Eds.), Trends in Natural Language Generation An Artificial Intelligence Perspective, Springer Berlin Heidelberg, Berlin, Heidelberg, 1996, pp. 88–105.

[28] M. Fox, D. Long, The 3rd international planning competition: Results and analysis, Journal of Artificial Intelligence Research 20 (2003) 1–59.

[29] J. Barreiro, G. Jones, S. Schaffer, Peer-to-peer planning for space mission control, in: 2009 IEEE Aerospace conference, 2009, pp. 1–9.

[30] K. Halsey, D. Long, M. Fox, Crikey-a temporal planner looking at the integration of scheduling and planning, in: Workshop on Integrating Planning into Scheduling, ICAPS, Citeseer, 2004, pp. 46–52.

[31] A. Rosenfeld, Better metrics for evaluating explainable artificial intelligence, in: Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '21, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2021, p. 45–50.

[32] A. Georgara, J. A. Rodriguez Aguilar, C. Sierra, Building contrastive explanations for multi-agent team formation, in: Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS '22, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2022, p. 516–524.

[33] K. Rayner, E. R. Schotter, M. E. J. Masson, M. C. Potter, R. Treiman, So much to read, so little time: How do we read, and can speed reading help?, Psychological Science in the Public Interest 17 (1) (2016) 4–34.

[34] J. Chall, E. Dale, Readability Revisited: The New Dale-Chall Readability Formula, Brookline Books, 1995.
URL https://books.google.es/books?id=2nbuAAAAMAAJ