Teaching Commonsense and Dynamic Knowledge to Service Robots

Stephan Opfer, Stefan Jakob, and Kurt Geihs

University of Kassel, Wilhelmshöher Allee 73, 34121 Kassel, Germany,
{opfer, s.jakob, geihs}@uni-kassel.de

Abstract. Incorporating commonsense and coping with dynamic knowledge are key capabilities of service robots to efficiently interact with humans. In the presented system, we demonstrate how to equip service robots with commonsense knowledge and the dynamic reasoning capabilities of Answer Set Programming (ASP). We investigated the response of our system to basic human needs and evaluated the viability and scalability of the combination of the commonsense knowledge database ConceptNet 5 and the ASP solver Clingo. Our results show the flexibility and versatility of our approach. Further, we identified the need for research on scalability in case of environments that are abundant with objects.

Keywords: Commonsense Reasoning, Dynamic Knowledge, Answer Set Programming, Service Robots, Cognitive Robotics, Human-Robot Interaction

1 Introduction

The research area of Cognitive Robotics is concerned with high-level control of robots and endowing robots with cognitive capabilities like learning and reasoning. In the context of human-robot interaction, these capabilities are key ingredients, to create elaborated dialogues that humans find pleasant to partake. In dialogue systems these capabilities, if present at all, are implemented by submodules of the dialogue manager that controls the state and flow of the conversation.

In this paper, we propose a reasoning system that allows humans to teach robots and to dynamically adapt the knowledge of the robots. Additionally, it provides a vast amount of commonsense knowledge for even understanding implicit service requests. Our approach utilises the database ConceptNet 5 (CN5) [14] as a source of commonsense knowledge and the Answer Set Programming (ASP) solver Clingo [6] as the underlying reasoner. For a detailed specification of ASP, the interested reader is referred to [3, 7]. One reason for choosing ASP is its non-monotonic reasoning capability that makes it flexible regarding dynamic knowledge and helps to keep the knowledge base consistent. Furthermore, Clingo provides an easy-to-use and efficient implementation for

ASP reasoning and the expressiveness of ASP outperforms many other decidable logical formalisms [12].

The application domain considered in this paper is domestic service robots, where humans assign tasks or have implicit requests for the robots. The following short dialogue describes the vision of this paper. *Alice: "I am tired." Robot: "Do you want to sit down?"* The service robot should be able to understand the needs of a human. This has to be done without a priori knowledge specifically provided for this kind of situations and thus rely on a general commonsense knowledge database.

The main contribution in this paper is the improved flexibility and expressiveness of human-robot interaction by combining non-monotonic reasoning, dynamic teaching, human-robot interaction, and a vast amount of commonsense knowledge.

In the next section, we summarize related research that was conducted with a similar setup of tools or tackle related problems. Afterwards, in Sect. 3, we present the foundations *External Statements* in ASP and CN5 necessary to understand the remaining paper. In Sect. 4, we explain the extraction of commonsense knowledge followed by its translation into ASP, necessary for utilising commonsense knowledge in domestic service robot scenarios. For evaluating the applicability of our approach, we investigate the runtime performance of CN5 and Clingo in Sect. 5 for different scenarios. In Sect. 6, we conclude the paper with a critical discussion, identifying potential improvements.

2 Related Work

In the following, we discuss state-of-the-art approaches and frameworks for utilising general knowledge in the field of autonomous service robots and emphasize the differences to our approach.

Erdem et al. present in [5] a hybrid planning approach utilizing ASP, Prolog, ConceptNet 4 (CN4), and a continuous motion planner. Their example scenario, tidying a house, is modelled in ASP, while the necessary commonsense knowledge about usual object locations and object properties are extracted from CN4 and represented within Prolog. In order to avoid inconsistencies, Erdem et al. rely on only two commonsense knowledge relations (*AtLocation* and *HasProperty*) and restrict their queries to a sufficiently small number of answers.

The work of Erdem et al. and our approach have a lot in common in general. However, apart from more current versions of ConceptNet and the ASP solver Clingo, we represent the commonsense knowledge directly in ASP, instead of external Prolog predicates. Furthermore, our approach allows retracting the commonsense knowledge from the ASP knowledge base, in case it causes inconsistencies or is overwritten by temporary exceptions from human input. As a result, our approach is much more suitable for the dynamic knowledge of long-term human-robot interaction.

In contrast to Erdem et al., Chen et al. [4] explicitly address human-robot interaction by processing limited segments of natural language (LSNL). For the automatic translation into ASP, the LSNL are restricted to if-then sentences. In contrast to our approach, other forms of commonsense knowledge cannot be added at runtime. In order to avoid inconsistencies during long-term human-robot interaction, the truth values of all ASP rules depend on a monotonically increasing and discrete time step.

In Lemaignan et al. [11] a framework for an autonomous robot with sophisticated human-robot interaction capabilities is described. The core reasoning formalism is ontology-based and utilises the Pellet reasoner. As a consequence new knowledge can only be added at runtime, by reclassifying the whole knowledge base. However, the knowledge base of the robot is enlarged by databases like WordNet and DBPedia that are also part of CN5. The dialogue management component Dialogs [10] intensively exploits the knowledge base for semantic resolution of nouns, pronouns, anaphors, and verbal phrases. In our opinion, an ASP-based approach would avoid several disadvantages of ontology-based reasoning [12].

As we explicitly address dynamic knowledge, the concepts and principles of truth maintenance systems [2], techniques for intentional forgetting [8], and the research in the field of Ambient Assisted Living [1] are therefore of interest for our approach, too. In order to retract commonsense knowledge from our knowledge base, we utilise the *External Statement* feature of the ASP solver Clingo.

3 Foundations

This section introduces the necessary basics for this paper, divided into two parts: Dynamic reasoning with Answer Set Programming in Sect. 3.1, and in Sect. 3.2 we introduce the commonsense knowledge database CN5.

3.1 Dynamic Reasoning with Answer Set Programming

ASP is a declarative logic programming approach, tackling NP-search problems. ASP can be seen as the result of research in the areas of knowledge representation, logic programming, and constraint satisfaction [3]. ASP is non-monotonic and thus allows to invalidate conclusions if contrary knowledge has been derived. For additional information, the interested reader is referred to [3, 7].

The ASP solver Clingo [6] can be used in a multi-shot fashion, which means that a single solver instance can be used multiple times and derived knowledge can be reused. Furthermore, it makes it possible to dynamically change knowledge by dividing an ASP program into *Program Sections* and by changing the truth value of *External Statements*. An example of the usage of *External Statements* is presented in Listing 1.1.

```
#program birds.
#external -flies(tweety).
bird(tweety).
flies(X) :- bird(X), not -flies(X).
```

Listing 1.1: Example for the usage of *External Statements*.

This example contains the *Program Section* `birds` in Line 1, which can be used to explicitly address this program. The *External Statement* in Line 2 expresses the capability of `tweety` to `fly`, depending on the truth value of the *External Statement*. Initially, Clingo assumes that all *External Statements* are false unless they are assigned otherwise. Solving this program results in the Stable Model [7]: {`bird(tweety)`, `flies(tweety)`}. Afterwards, the truth value of `-flies(tweety)` can be changed to true, resulting in a new Stable Model that contains the following atoms: {`bird(tweety)`, `-flies(tweety)`}.

3.2 ConceptNet 5

ConceptNet 5 (CN5) is a multi-language knowledge database. Its structure is a semantic hypergraph that contains commonsense knowledge expressed in triples containing natural language, which has been collected from different sources like the Open Mind Common Sense project, Wiktionary, WordNet 3.0, DBPedia, and Wikipedia [14]. A node in the CN5 hypergraph is denoted as a concept, which is a combination of a natural language term, a language tag, and a sense label. The sense label distinguishes between nouns, verbs, adjectives, and adverbs. Edges between concepts contain the sources used to create it, e. g., WordNet, its weight, and the concepts connected to it. The weight is a sum of the weights of each source. A weight greater than 1.0 indicates that the edge has been extracted from a verified source like WordNet. Furthermore, each edge contains a relation denoting its meaning. The relations are either automatically extracted from sources like Wikipedia, which can be in any language or are part of a handmade set of English base relations. Table 1 is an excerpt from these base relations and their meaning.

Table 1: 14 from 40 CN5 base relations [14].

Relation	**Meaning**	**Relation**	**Meaning**
IsA	A is a kind of B	SymbolOf	A represents B
UsedFor	A is used for B	HasContext	A is in the context of B
CapableOf	A is able to B	MotivatedByGoal	Doing A to achieve B
RelatedTo	A and B are related	CausesDesire	A would make you want B
Desires	A wants to B	MadeOf	A is made of B
Causes	The effect of A is B	AtLocation	A is located at B
HasProperty	A is B	PartOf	A is part of B

In order to demonstrate the utility of CN5, let us consider the following utterance: "I'm tired!". Understanding the utterance as an implicit service request requires commonsense knowledge that is represented within CN5. For example, CN5 suggests to *rest* is *MotivatedByGoal tired* and that *going to sleep* is *UsedFor rest*.

4 A Commonsense Knowledge Course for Robots

Our automated approach of providing a service robot with commonsense knowledge includes several steps. In Section 4.1, we describe the retrieval of new knowledge from human-robot interaction, knowledge databases, or robotic sensory input. The knowledge is then automatically translated into ASP rules and made available on demand to, e. g., reason about the environment (see Sect. 4.2). The application example in Sect. 4.3 explains how the robot applies commonsense knowledge with or without further human interference.

4.1 Knowledge for Interactive Service Robots

We allow humans to teach service robots through informative utterances like "Cups are located in kitchens.". Therefore, the knowledge should be representable as n-ary predicates, in this case `atLocation(cup, kitchen)`. If the service robot does not know anything about the things or concepts mentioned by the human, our approach fills the lack of knowledge by querying a knowledge database like CN5. A third source of knowledge is the sensory input of the robot. Frameworks like YOLO or the Google Vision API recognise objects in the video stream of the robot. Therefore, it is possible to identify a room, for example, as a kitchen, because of the recognised stove.

The knowledge from the different sources is either commonsense or situational knowledge. While commonsense knowledge is concerned with knowledge that can be considered to hold in general and seldom changes, situational knowledge is concerned with knowledge that is specific for the current situation and changes frequently, e. g., "The blue cup is located on the table in room R1405.". Sensory input always provides situational knowledge, while informative human utterances can include both, situational and commonsense knowledge. Knowledge databases like CN5 only include commonsense knowledge. Please note that, although with different frequencies, we consider both types of knowledge to be subject of change.

4.2 Modelling Knowledge in ASP

Translating the n-ary predicates, representing the relevant knowledge, into ASP is done as shown in Line 1 to 6 of Listing 1.2, for the example of CN5. Summarized in an extra program section, each edge corresponds to one ASP fact including the reliability of the sources of the predicates. Utilising the `cs_AtLocation` edges, e. g., a service robot is already capable of searching for items, without a human that needs to tell it where to start searching.

```
1 #program commonsenseKnowledge.
2 cs_CapableOf(cup,hold_liquids,7).
3 cs_AtLocation(cup,table,4).
4 cs_AtLocation(cup,shelf,3).
5 cs_UsedFor(cup,drinking,3).
```

```
6  cs_AtLocation(coffee,cup,2).
7  ...
8  #program situationalKnowledge(n,m).
9  #external -atLocation(n,m).
10 atLocation(n,m,W):- not -atLocation(n,m), typeOf(n,cup),
       typeOf(m,table), cs_AtLocation(cup,table,W).
11 atLocation(n,m,W):- not -atLocation(n,m), typeOf(n,cup),
       typeOf(m,shelf), cs_AtLocation(cup,shelf,W).
12 ...
13 #program sensorInput.
14 typeOf(blueCup,cup).
15 typeOf(kitchenTable,table).
16 typeOf(kitchenShelf,shelf).
17 ...
```

Listing 1.2: Extract from commonsense and situational knowledge about cups.

The *Program Section* `situationalKnowledge(n,m)`, starting in Line 8, is necessary to apply the commonsense knowledge of the former section to objects and concepts encountered through object recognition or human-robot interaction. `n` and `m` are parameters used for grounding the *Program Section* with objects from the environment of the robot. The rule in Line 10, e. g., informally means that as long as there is no proof that `n` is not located at `m`, `n` is of type `cup`, `m` is of type `table`, and it is commonsense that a `cup` is usually located at a `table` with a reliability of `W`, it is assumed that `n` is located at `m` with the corresponding reliability. The *External Statement* in Line 9 can be set to true, e. g., via human-robot interaction, in order to give proof that a certain object `n` is not located at `m`.

4.3 Synergy of Commonsense and Situational Knowledge

The following example scenario clarifies how our approach helps to achieve the aforementioned vision (see Sect. 1) and illustrates how the presented components intertwine. Imagine a service robot able to extract knowledge from its environment. Therefore, the robot labels specific objects in its environment with categories from its commonsense knowledge, which is shown in Line 14 to 16 of Listing 1.2. Additional commonsense knowledge, for example, the possible locations of a cup, is integrated into the knowledge base on demand and is partially retracted from it according to the current situation. Relying on this knowledge, the robot would search for a cup on the `kitchenTable`, but the human asking for the cup already told it that currently there is no cup on the `kitchenTable` and thereby sets the *External Statement* `-atLocation(blueCup, kitchenTable)` to true. The robot does not know about any other table and therefore is searching for a cup in the `kitchenShelf`, instead.

5 Evaluation

The evaluation has been performed on a Desktop PC equipped with an Intel Core i7-8700K@3.7 GHz Hexa-Core, 64 GB DDR4-2133 MHz RAM, and Ubuntu 18.04 64 Bit as its operating system. The version of CN5 is 5.7 and the ASP solver Clingo 5.3.1 was used, which incorporates the grounder Gringo with version 5.3.1 and the solver Clasp with version 3.3.4. We conducted three experiments, as explained in the next sections. The first two evaluate how effective our approach is regarding performance and scalability. The last experiment investigates the understanding of implicit human requests by using ASP and CN5, only.

5.1 System Performance

The most essential step is to teach new concepts to service robots. We measured the runtime of the two included substeps, namely querying CN5 and translating the results into ASP. As the runtime depends on the size of the taught concept, we chose three exemplary concepts `water` (2425 edges), `dog` (1144 edges), and `cup` (333 edges), since they are a large, medium, and small-sized concept, respectively. The runtime for querying CN5 and translating the results into ASP was measured 2000 times. Both scale linearly with the number of edges (see Table 2), while the runtime for translating the query results into ASP is neglectable compared to the query process itself.

Table 2: Runtime Evaluation of knowledge extraction and translation.

Concept	**CN5 Extraction**	**CN5 SD**	**ASP Translation**	**ASP SD**
water	702.529 ms	15.322 ms	64.412 ms	1.335 ms
dog	391.238 ms	21.164 ms	30.483 ms	1.389 ms
cup	174.783 ms	5.985 ms	9.034 ms	0.305 ms

Another important aspect is the scalability of the knowledge base. Typically thousands of relations between objects need to be taught to service robots operating in an average household scenario. For the following experiment, we engineered a scenario consisting of 102 objects and rooms represented by 39 different concepts. The service robot is supposed to tidy up the household and therefore needs to know the usual locations of the objects. This experiment evaluates the initialization process of the knowledge base of a robot and consists of three main parts. The first part, like before, is the integration of sensor inputs into the knowledge base. Afterwards, the 39 different concepts stemming from the perceived objects are queried from CN5. Finally, 84 typical object locations are extracted from the situational knowledge. We consider the size of the scenario suitable for investigating the scalability of our approach inside small household environments. The runtime of the three consecutive steps was measured 1000 times, resulting in an average runtime of 68.2 s (SD: 0.46 s).

5.2 Implicit Human Service Requests

Within this experiment we approach the vision stated in the introduction of this paper, i. e. making service robots understand implicitly formulated desires of human beings. Therefore, the service robot was equipped with an algorithm to recognise basic human needs and their edges to other concepts within CN5. Our implementation utilises several CN5 base relations for finding solutions to basic human needs. At first, the *MotivatedByGoal* and *CausesDesire* relations identify activities that help to remedy the needs and further the *UsedFor* relation suggests objects and actions that are necessary for executing these activities. Additionally, possible activities are expanded via synonym like relations, which include *SimilarTo*, *InstanceOf* ,and *Synonym*.

Table 3: Excerpt from suggested actions for basic human needs.

Issue	**Activity**	**Used Objects or Actions**
tired	rest, sleep	sitting down, sleeping at night, going to sleep, bed, hotel
thirsty	drink, drink water	water, ice, beer mug, faucet, glass
hungry	cook, eat food, cook meal	fork, forks, indian restaurant, stove, kitchen
hot	cool off, swim	fan, pool, river, stream, ocean
cold	light fire, start fire	match, matches, flint, kindling, lighter
bored	watch television, play games	living room, playroom, computer, family room, basketball court
ill	go to doctor, lie down	illness, bed
smell	take shower, take bath	shower head, shower stall, portable shower head, bathtub, tub
stink	bathe	bars of soap, bathtub, wash cloth, separate shower
lonely	call friend	telephone

Table 3 lists an excerpt from the investigated basic needs and the actions found through the *MotivatedByGoal* relation. A tired human, e. g., is suggested to sleep. Making the robot a real assistant, instead of only a clever advice machine, it further analyses objects that are typically used for the suggested actions. A human that should sleep, e. g., needs a bed or a hotel. Although these results for fulfilling implicit human desires by service robots are very promising, further possible improvements are discussed in the next section.

6 Discussion and Conclusion

In this paper, we introduce our approach to assist humans in teaching service robots commonsense knowledge, while keeping the taught knowledge changeable. Furthermore, we show that a service robot utilising the commonsense knowledge database CN5 is capable of appropriately replying to implicit service requests of humans.

Through the performance evaluation of the teaching algorithm, the runtime for CN5 queries is identified as the most time-consuming step, often taking ten times longer than the translation into ASP. Therefore, caching mechanisms, as provided by web servers like *nginx*, would significantly reduce the runtime. However, the low runtime already allows the application in human-robot interaction, since answers are not delayed by the use of CN5 or the translation into ASP.

The reason for choosing ASP as knowledge representation language is its capability of non-monotonic reasoning and dynamically retracting knowledge via *External Statements* (see Sect. 4.3). Please note that the number of *External Statements* substantially influences the runtime of the reasoning process as shown in our preliminary work [13] and therefore limits the applicability for environments abundant with objects. Furthermore, to the best of our knowledge, it is open how *External Statements* relate to the research of intentional forgetting in ASP [9]. Although current results regarding forgetting knowledge [8] are discouraging from a theoretical point of view, we see a need for further research regarding this issue from a more practical point of view. A reason for retracting knowledge, apart from its invalidity, is to keep the knowledge base syntactically and semantically consistent.

Regarding the semantic consistency of our knowledge base, we plan to further utilise additional CN5 base relations like *Antonym*, *Synonym*, *IsA*, *DefinedAs*, and *PartOf* in order to detect inconsistent concept descriptions induced by the *HasProperty*, *HasContext*, or *RelatedTo* relations. An example is the concept of a `plant`, because it is connected to the concepts `evergreen` and `deciduous`, whose contradiction is represented by their *Antonym* relation. A service robot could conclude that both properties hold and introduce a semantic inconsistency to its knowledge base.

In our opinion, the results of solving implicit human requests are already encouraging (see Table 3) but can be improved by utilising the full potential of CN5. While the suggested activities for *tired* or *thirsty* are suitable, starting a fire in case the human user stated that it is *cold* is not a feasible solution in a house. In order to improve the answer quality of the service robot and increase the number of recognised human needs, adopting additional features of CN5 like Relatedness, optimizing the average edge weight of extracted concept graphs, and considering the optional sense label of concepts are our most promising ideas. Especially the Relatedness feature could improve the results since they rely on word embeddings [14] and thus can be used to either merge closely related concepts like *fork* and *forks* or to increase the number of considered concepts for answering implicit human requests.

Finally, we conclude that teaching commonsense knowledge with the help of existing databases like CN5 should supersede the approach of learning everything from scratch because we have already achieved encouraging results without learning at all. Furthermore, in our opinion, ASP is the only knowledge representation and reasoning technology whose expressiveness and performance scales with our scenarios and at the same time provides ready to use implementations, like Clingo. Nevertheless, further research is deemed necessary concerning the capabilities of ASP regarding the handling of dynamic knowledge.

References

1. Ayari, N., Chibani, A., Amirat, Y., Matson, E.T.: A novel approach based on commonsense knowledge representation and reasoning in open world for intelligent ambient assisted living services. In: IROS 2015 - IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 6007–6013 (2015)
2. Beck, H.: Reviewing Justification-based Truth Maintenance Systems from a Logic Programming Perspective. Tech. Rep. INFSYS RR-1843-17-02, Institute of Information Systems, TU Vienna (July 2017)
3. Brewka, G., Eiter, T., Truszczyński, M.: Answer Set Programming at a Glance. Communications of the ACM **54**(12), 92–103 (2011)
4. Chen, X., Ji, J., Jiang, J., Jin, G., Wang, F., Xie, J.: Developing high-level cognitive functions for service robots. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1. pp. 989–996. AAMAS '10, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2010)
5. Erdem, E., Aker, E., Patoglu, V.: Answer Set Programming for Collaborative Housekeeping Robotics: Representation, Reasoning, and Execution. Intelligent Service Robotics **5**(4), 275–291 (2012)
6. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Clingo=ASP+Control: Extended Report. Tech. rep., Knowledge Processing and Information Systems (2014)
7. Gelfond, M., Kahl, Y.: Knowledge Representation, Reasoning, and the Design of Intelligent Agents: The Answer-Set Orogramming Approach. Cambridge University Press, Cambridge, USA (2014)
8. Gonçalves, R., Knorr, M., Leite, J.: You Can't Always Forget What You Want: On the Limits of Forgetting in Answer Set Programming. In: European Conference on Artificial Intelligence (ECAI). pp. 957–965. Frontiers in Artificial Intelligence and Applications (2016)
9. Leite, J.: A birds-eye view of forgetting in answer-set programming. In: International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR). pp. 10–22 (2017)
10. Lemaignan, S., Ros, R., Sisbot, E.A., Alami, R., Beetz, M.: Grounding the interaction: Anchoring situated discourse in everyday human-robot interaction. International Journal of Social Robotics **4**(2), 181–199 (2012)
11. Lemaignan, S., Warnier, M., Sisbot, E.A., Clodic, A., Alami, R.: Artificial cognition for social humanrobot interaction: An implementation. Artificial Intelligence **247**, 45 – 69 (2017), special Issue on AI and Robotics
12. Motik, B., Horrocks, I., Rosati, R., Sattler, U.: Can OWL and Logic Programming Live Together Happily Ever After? In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) The Semantic Web - ISWC 2006, Lecture Notes in Computer Science, vol. 4273, pp. 501–514 (2006)
13. Opfer, S., Jakob, S., Geihs, K.: Reasoning for Autonomous Agents in Dynamic Domains: Towards Automatic Satisfaction of the Module Property. In: International Conference on Agents and Artificial Intelligence. pp. 22–47 (2017)
14. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: An open multilingual graph of general knowledge (2017)