Memory Nets: Knowledge Representation for Intelligent Agent Operations in Real World

Julian Eggert, Jörg Deigmöller, Lydia Fischer and Andreas Richter
Honda Research Institute Europe, Carl-Legien Strasse 30, 63073 Offenbach, Germany

Keywords: Memory Nets, Semantic Net, Common Sense, Knowledge Graph, Knowledge Base, Property Graph, Grounding, Situated Interaction.

Abstract: In this paper, we introduce Memory Nets, a knowledge representation targeted at Autonomous Intelligent Agents (IAs) operating in real world. The main focus is on a knowledge base (KB) that on the one hand is able to leverage the large body of openly available semantic information, and on the other hand allows to incrementally accumulate additional knowledge from situated interaction. Such a KB can only rely on operable semantics fully contained in the knowledge base itself, avoiding any type of hidden semantics in the KB attributes, such as human-interpretable identifier. In addition, it has to provide means for tightly coupling the internal representation to real-world events. We propose a KB structure and inference processes based on a knowledge graph that has a small number of link types with operational semantics only, and where the main information lies in the complex patterns and connectivity structures that can be build incrementally using these links. We describe the basic domain independent features of Memory Nets and the relation to measurements and actuator capabilities as available by autonomous entities, with the target of providing a KB framework for researching how to create IAs that continuously expand their knowledge about the world.

1 INTRODUCTION

In the artificial intelligence domain, the term **Intelligent Agent (IA)** is commonly used for autonomous entities with sensors and actuators which act within an environment in a purposeful way (Wikipedia, 2019; Franklin and Graesser, 1996; Kasabov and Kozma, 1998). Albeit being very general, there are several aspects of this definition which directly point to the very essence of the underlying scientific and technological problems. First, the agent has some degrees of freedom which it can use to decide about its own behavior in an independent way, pursuing its own goals. Second, it is always embedded within an environment and has to be considered as an integral part of it, and it operates by interacting with the environment. And third, it tries to achieve its goals by exploiting its own and the environment's limited resources (including time) in an efficient manner (Kurzweil, 1999).

In essence, an IA combines three aspects: It incorporates new information about selected parts of its environment by means of its sensors, connects and combines it with already existent, previously known information about the environment, and uses the combined information to derive conclusions which help in achieving its goals and executing the actuators. In any case, an internal **Knowledge Base (KB)** system is involved. This system has two roles: On the one hand, the system provides the KB itself which serves to store and retrieve complex structured and unstructured information about the environment and the world. Its content is updated by incorporating knowledge from external sources as well as from the information fed by the agents' sensors. On the other hand, the KB system provides a reasoning engine which allows to infer or derive additional knowledge from available information. This becomes necessary for the solution of tasks in complex conditions as well as for the incremental expansion of the KB by learning and exploration.

This paper introduces **Memory Nets**, a KB system that enables autonomous IAs to leverage knowledge during their interaction with the real world. Our belief is that future IAs need an understanding of who, what and where they are, what the important things are that they have to care about, and how they use all this understanding to pursue their goals in the best way. The important properties of such IAs supported by the KB system are:

1. **Situatedness**: They know about their current context and task

Eggert, J., Deigmöller, J., Fischer, L. and Richter, A.
Memory Nets: Knowledge Representation for Intelligent Agent Operations in Real World.
DOI: 10.5220/0008068101150126
In *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2019)*, pages 115-126
ISBN: 978-989-758-382-7

2. **Operability**: They use this knowledge to operate in the world
3. **Incrementality**: They expand their knowledge about the world via interaction

In addition, point 1) comprises several subitems, most importantly 1.a) **Grounding** the IAs have to establish links to those real-world items (objects, subjects) which are relevant for a targeted task, 1.b) **Common sense** they need a basic background knowledge about world items and processes similar to humans and 1.c) **Self-reference** they need to know about their role(s), capabilities/limitations and own observations.

Hence, what are representational structures that support a KB system with the indicated properties 1)-3)? And what are processes/operations of an inference engine that support the situatedness, operability and incrementality, since a KB system implies such an engine using the KB always?

In the remainder of the paper, we will approach these questions by sketching some necessary principles of a KB system that supports the described IAs properties. With this in mind, we first shortly review the KB landscape (section 2), highlighting some of the issues that to our knowledge fall short for their implementation as IA's knowledge base. In the following section 3 we explain the main principles of the Memory Nets KB approach. Then we describe implementation details of Memory Nets in sections 4, and show an example in section 5.

2 RELATED WORK

2.1 Knowledge Representation in General Domains

Understanding KB systems as a combination of knowledge representation and reasoning capabilities, with the advent of the internet and its huge amount of interlinked documents, it was proposed that it could be beneficial to provide a unified access to the cross-website information stores. Therefore the **Semantic Web** provides an augmentation of the internet by an additional semantic layer which describes the "meaning" of sites in a standardized, human-interpretable way, and which can be used for search and inference in a more effective way than by inspecting the document data (Berners-Lee et al., 2001). For this purpose, special markup languages based on the XML format have been specified. The Resource Description Framework (RDF) is a specification for a metadata model, as a way to describe formally and conceptually the information that is contained in web resources, with basic capabilities to define classes, subclasses and properties of objects. The Web Ontology Language (OWL) builds on top of the RDF and provides additional levels of semantics to describe ontologies of concepts (Antoniou et al., 2012; Knublauch et al., 2006). The underlying triple structure of RDF relates two objects x, y via a predicate P, which could be interpreted as a logical formula $P(x, y)$. On the other hand, it could be seen as a directed graph with x being the subject and y being the object of a statement, which is a **Semantic Net** (Collins and Quillian, 1969; Simmons, 1973). One famous example of a Semantic Net is WordNet (Miller, 1995).

Later on, the Linked Data (Wood et al., 2014) development focused on interfacing as many RDF databases as possible and using them as one huge knowledge source. This organization of knowledge changed the thinking from carefully built databases to **Knowledge Graphs** that collect any kind of machine-readable structured data. In principle, a Knowledge Graph can be seen as any graph-based representation of knowledge (Paulheim, 2017). In 2012, Google introduced its "Google Knowledge Graph" that is an accumulated information about the searches, query sessions, and clicks that searchers perform on the Web each day (Singhal, 2012). Other examples of Knowledge Graphs are Freebase (Bollacker et al., 2008) or WikiData (Vrandecic and Krötzsch, 2014) that are edited by the crowd. DBPedia (Lehmann et al., 2015) or Yago (Suchanek et al., 2007) extract their knowledge from large-scale, semi-structured web bases such as Wikipedia. The target of all previously mentioned databases is to describe general facts as well as relations of everyday facts, also called **Commonsense Knowledge**, and make them globally accessible.

On the one hand, the Semantic Web facilitates the creation of domain specific databases and shareable ontologies (formal descriptions of knowledge). On the other hand, there are attempts to describe domain independent databases that support a semantic interoperability among the domain specific ones, so-called **Upper Ontologies** (Degen and Herre, 2001). An Upper Ontology usually differentiates between universals and individuals - which are instances of universals located in space and time. Individuals, in turn, cannot have instances. Upper Ontologies are philosophically motivated and try to describe basic categories and relations that are valid among ontologies. Some of the most well-known Upper Ontologies are SUMO (Pease et al., 2002), Cyc, DOLCE, PROTON and Sowa's ontology (Sowa, 2000).

The mentioned standardization and interfacing efforts, together with the growing amount of available

knowledge, suggest that a KB system for IAs is increasingly getting into reach. In the ideal case, a large amount of the knowledge that an IA needs would already be available, providing it a jump start when exploiting it to tackle ambitious problems. However, even if the available knowledge is machine readable, its semantics are not yet fully understandable by IAs operating in real world. Nearly all of the available ontologies have been created with a human designer in the loop, requiring human interpretation. This applies especially to the predicates/relationships (such as e. g. "movingTo", "peformedBy", but also as complex as "hasLastRegisteredMovementAt") which contain implicit semantics that are not directly accessible to an IA, and difficult to connect to daily operation in real world. In addition, even if inference frameworks are able to operate upon the ontologies, again this is used rather as analyzing tool for humans, and not for an autonomous incremental ontology extension.

2.2 Knowledge Representation for Autonomous Intelligent Agents

Nevertheless, for real-world operation, the robotics community is trying to make use of the previously mentioned knowledge databases and frameworks to link conceptual information to the physical world. The main focus here is to fill in missing information required by a robot for executing tasks and reaching goals. Unfortunately, most of the knowledge bases in the semantic web were created for understanding and processing text. Hence, even Upper Ontologies like Cyc or SUMO are not very useful from a robotics point of view which has to connect symbols with sensations and actions (Tenorth et al., 2011; Fischer et al., 2018).

A promising direction in bringing both worlds together is by describing objects by their affordances (Horton et al., 2012) or linking capabilities of robots and actions to symbols (Kunze et al., 2011). This resulted in open robot ontologies built for executing special tasks (Lemaignan et al., 2010; Beetz et al., 2018). Other branches of research tackle the decomposition of real-world objects into their conceptual components to share situational knowledge on a more fundamental and prototypical level (Rebhan et al., 2009; Wyatt et al., 2010; Röhrbein et al., 2009).

However, the gained knowledge representation frameworks are rather domain-specific and constrained in their generalization towards continuous real-world operation. The dynamic extension of an IA's knowledge representation based on situation analysis and a rich existing foundation of available knowledge is still an unresolved problem. The same holds for the ability of an IA to transfer knowledge to different and new situations based on their conceptual description. To tackle these problems, we argue that we likely need KB systems with specific properties. Some of these properties are explained in the following sections.

3 MEMORY NETS: MAIN PRINCIPLES

In this section we outline the main guiding principles which are the basis of Memory Nets representations and processes that operate on it. There are two key ideas which markedly differ from standard KB approaches. First, Memory Nets are targeted to be open for intrinsic extension by the system itself, leading to an open-ended creation of new concepts and problem solving capabilities, and as such to a larger flexibility to react dynamically and adapt to changing situations. Second, as a consequence of the intrinsic extension, the KB can be operated without the knowledge of an external interpreter or designer that needs to make sense of the data.

3.1 General Guiding Principles

- The elementary representational substrate for Memory Nets are **labeled property graphs**, i. e., graphs with defined link types and with nodes and links which can have key-value attributes. The reason is the expressive power of graphs (e. g., an RDF structure can be easily embedded into a graph), its natural handling in terms of visually discernible structures, and the point that much of the interesting information in KBs lies in the structure of the connectivity.
- Concepts are described in the graph by prototypical **patterns**. A pattern is a reoccurring structure (in terms of graph similarity measures, i .e. arrangement of nodes and links) with an identifiable inner part (nodes and links) and further links which connect to other patterns and which embed the concept within a web of related concepts.
- Compositionality and inheritance are used for **redundancy reduction and minimum description length** principles. The idea is to identify common structures which can be reused as common components in different concepts, and to represent repeated occurrence of structures in a compact way.
- The graph representation should serve as a basis for **different inference paradigms** on the same data or on subsets of the data, e. g. first order logic calculus, graph similarity finding, pattern match-

ing, activity spread algorithms, and probabilistic reasoning.

- Graph elements (especially links) have **no hidden identifier semantics**. Identifier do not provide any item-specific semantic meaning, so that concepts of the KB have meaning without the need of a human-interpretable identifier, and can be used without the knowledge of the data designer. The purpose of different types of links is only to provide operational semantics, i .e., setting constraints on graph structures and specifying how we can operate on them.
- Instead of using natural language identifier to connect from outside (real world) to concepts within the graph, we use **measurement patterns**, which by inference (e. g. pattern matching) have to be matched to the most likely concepts.
- Natural language labels ("utterances") are stored in the same way as all other measurable properties, contributing if desired to pattern matching and pattern finding processes.

3.2 Requirements for Real-world Operation

- The KB is based on **open world** assumptions, i. e., everything we do not know is undefined (Mazzocchi, 2005), which is necessary for an open-ended expandability during operation. In addition, the KB should be able to incorporate information from external sources, like semantic web databases and services, which also rely on the open world assumption.
- Within the KB, **new concepts can be created by the system itself** during operation, either by inference or by incorporation of new measurement patterns. The KB should be capable of building new concepts based on existing ones (straightforward examples are hierarchical or chaining compositionality); as well as building specialized or modified concepts from more general ones. This is in synchronization with the assumption of no item-specific semantic information of graph element identifiers.
- The properties of concepts stored in the KB establish a link to the real world. They should be **measurable**, or derived from measurable properties, and information should be contained in the KB on how the properties are measured (e .g. about sensory devices). As a consequence, there would be no completely isolated parts of the graph with no relation to the real world.
- The KB should be capable of operating in synchronization with a highly dynamic real-world. This implies the support of **grounding** processes that allow the creation of short-lived instances of real-world item representations, their life-time management, keeping them in focus and updating them over time, as well as the generation of short-lived prediction, hypothesis generation and testing. It also implies that time plays a special role both in representational as well as in processing aspects.

4 MEMORY NETS: IMPLEMENTATION

Here we provide a description of the main operational principles of Memory Nets. They are based on the principles of no hidden identifier semantics, measurable properties, compositionality and compact representation. We start by identifying a minimal set of link types which provide operational and inference semantics and on which the more complex structures are build upon. Then we explain the notion of concept patterns, specialization, inheritance and transformation chains, and sets.

4.1 Link Types

In knowledge graphs, nodes and relationships in form of (source)–[linktype]→(target) are often read as semantic subject-predicate-object triples where the link types play a special role, as e. g. in ("Bob")–["knows"]→("John") with the link type "knows" connecting "Bob" with "John". This is also the basis for triplestore databases and the atomic data entity for the RDF data model. The link to the external world is then given by the definition of the predicates and the interpretation of their meaning by a human designer. This rapidly turns into complex design decisions, as we can see in expressions like "The car has the color blue", where "has-the-color" now works as a predicate between "car" and "blue".

In Memory Nets, there are no hidden semantics stored in the link types. Therefore, we use a small set of link types which rather provide operational semantics which apply to all KB structures regardless of their content. Differently to e. g. RDF triple predicates, Memory Net link types do not encode directly a possible relationship between a subject (represented by the source node) and an object (represented by the target node), but they encode what can be done with the source and target nodes within the graph.

Table 1 shows the most important basic Memory Net directed link types and their operational mean-

Table 1: The Memory Net KB is based on a labeled property graph structure with purely operational semantics. The meaning of concepts is therefore given by the connectivity pattern, and the link types have consequences for the interpretation of the connectivity pattern and for inference processes. All Memory Net operations and concepts are based on a small set of link types. The labels of the link types do not, however, convey any semantics related to the concepts stored in the database, so that e. g. there are no special link types with hidden semantics like "isMadeOf" or "createdBy", which would require an external interpretation. The table shows the basic relationship types and their operational meaning.

Operational link semantics	Forward link type	Abbr. type	Link properties	Source/target roles
Specialization	specializesTo	specTo	transitive	Parent/child
Transformation	transformsTo	transTo	transitive	Starting point/result
Properties	hasProperty	hasProp		Property holder/property
Composationality	hasPart	hasPart	transitive	Whole / part
Set elements	hasElement	hasEl		Set / element
Binary operator	hasOperator	hasOp		first / second argument
Order/sequentiality	hasNext	hasNext	transitive	Preceeding / successor

ing. They provide the framework for the elementary structures that can be built in the Memory Net KB. Further explanations are given in the following sections 4.2, to 4.5. More complex content semantics should then be created by exploiting the basic link types. E. g., instead of introducing a "isMadeOf" link ("Object")–["isMadeOf"]→("Material"), we would consider a separate concept "Material" which specializes to the concrete material "CMaterial" the object is composed of and a corresponding relationship ("Object")–["hasProperty"]→("CMaterial"). In parallel, we could think of separately representing the process of making the object out of a certain material by using links that indicate transformation. As can be seen, the composition of semantic content usually hidden in complex, human interpretable predicates forces a representation that is closer to the actual processes and causal chains as they occur in the real world, and this is exactly the target of Memory Nets.

From table 1, it can be seen that four link types are transitive, i. e. that if (A)–[linktype]→(B) and (B)–[linktype]→(C) then it follows (A)–[linktype]→(C). This leads to a natural ordering of the involved nodes. In Memory Net, we therefore have at least four dimensions for ordered ontologies: The specialization dimension (which can be used to build class taxonomies), the transformation dimension (for transformation and causality chaining), the compositionality dimension (for building partonomies) and the sequential ordering dimension. All of these provide different semantic views on concepts and coexist within the same KB by crossing at common nodes.

4.2 Concepts as Patterns

Concept patterns are knowledge subgraphs with a number of features, represented by characteristic nodes and links, that together describe an aspect of the world, i. e., a concept, relevant for the IA. The single nodes contain describable, derivable resp. measurable components and properties of the concept. Patterns are represented by subgraphs which start from a central node to which all other nodes are directly or indirectly attached. This is the so-called hub node which is often taken as a proxy for the whole concept. The meaning of a concept name is given by its own nodes and links and by its contextual embedding within the graph (how is it related with other concepts) and not by a text identifier of its hub node (in explanatory images text identifier are included for illustrative purposes).

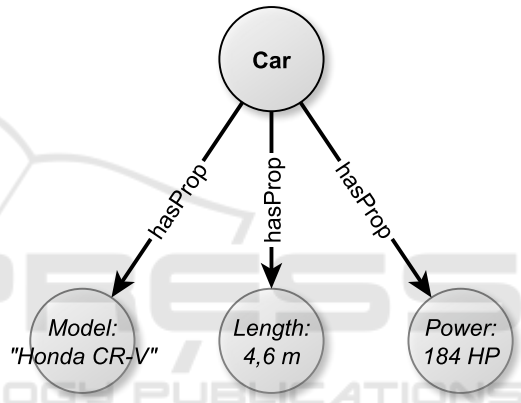

Figure 1: The simplest memory Net concept pattern represented by a hub node ("Car") and attached properties ("Model", "Length", "Power") with values. The node identifier names (e. g. "Car") are indicated here for illustrative purposes only, since the semantics of the Memory Net is purely given by the structure of the KB graph that contains the concept. Representing properties as separate nodes rather than attributes of a single node enables a better embedding in the graph, e. g. by specializing properties or putting them in relation with each other. See section 4.2 for more details.

A property pattern consists of a hub node with "property nodes" attached via "hasProperty" links. An example of such a pattern for the concept of "Car" displays Fig. 1. A property node has a special value attribute which contributes to a measurable quantity and provides the link to the real world. Property nodes are final, in the sense that they themselves cannot have further properties (e. g., it would not be meaningful to speak of a subproperty of "red", however see later the idea of property specialization).

Concept patterns are compositional, i. e. they can be composed to form larger patterns. Inversely, a concept pattern can have subpatterns, and subpatterns of subpatterns. The subpatterns' hub nodes are attached

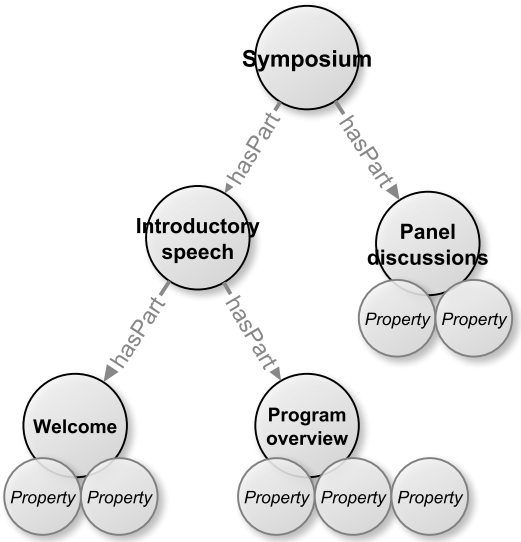

Figure 2: Example of non-physical partonomy and composition of patterns by subpatterns with "hasPart" relationships. Subpatterns help in the repeated usage of common modules in different patterns and enable a compact encoding.

to the higher level concept hub node by inverse "hasPart" links, leading to a part-of hierarchy (partonomy). The exact link designation as "hasPart" is somewhat arbitrary and in natural language it suggests mainly physical composition, here however it is used for arbitrary reusable subpatterns that contribute to the description of the original pattern. An example of a non-physical partonomy can be seen in Fig. 2.

In detail, a pattern is a tree-formed subgraph that starts from a hub node and which has links of type "hasProperty", "hasPart" and "hasElement", with the "hasPart" links forming branches. Further, links between pattern nodes can be used to express additional internal (intra-pattern) relations, and inter-pattern relations embed the concept in the KB. Fig. 3 shows two concept patterns for "Robot" and "Navigation path" with parts and properties as intra-pattern relations and with other intra- and inter-pattern relations (dotted lines). In this case, the dotted lines represent spatial relations and ordering, expressing that in the path one waypoint is followed by the other and that the spatial position of the robot is close to one of the waypoints.

A concept pattern has definitory character, in the sense that it should contain the important features that make up a concept or that allow it to be separated from other concepts. All other possible, not explicitly represented features are assumed to be unknown. One important task is to find the best match between new / unknown and existing concept patterns, e. g. when analyzing sensory measurements.

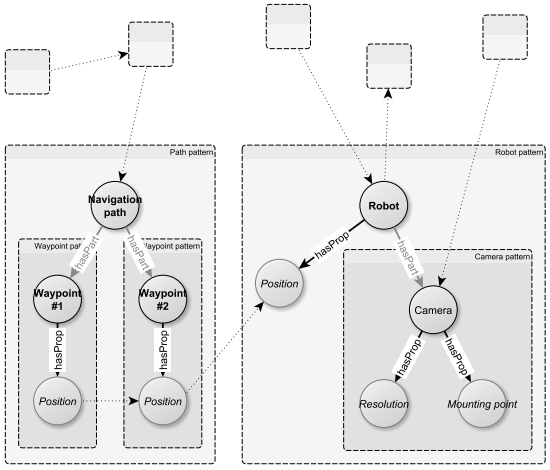

Figure 3: Patterns with parts and exemplary inter- and intra-pattern relationships. In larger networks, patterns form a dense web of interconnected concepts. The meaning of each concept is then given by its internal structure (all nodes of the tree given by outgoing "hasProperty", "hasPart" and "hasElement" links from the hub node with all additional intra-pattern relationships) and its semantic context, which results from the connection with other patterns (indicated by yet unspecified, dotted line relationships).

4.3 Specialization and Inheritance

Memory nets can represent similar concepts in a compact way by specialization. This denotes the operation that starts from a general pattern and derives from it a more specialized pattern by addition of further features. Concretely, it is allowed to specialize a concept by incorporation of additional nodes attached by links of "hasProperty", "hasPart" and "hasElement" type compatible with the pattern tree structure described in section 4.2. This results in a more precise definitory scope of the derived concept. As a consequence, the concept will be applicable in more specific cases only, or, the other way round, if a specialized concept applies to a new pattern, then also does the original concept it was derived from.

Pattern specialization is indicated by a "specializesTo" link from the general pattern hub node to the specialized pattern hub node. For a compact representation, when specializing a concept we do not repeat all the features of the original concept. Instead, we assume all the parent features (given by further pattern links) to be automatically inherited. The "specializesTo" link therefore indicates that the target node should be read as a sort of proxy, as if it was the equivalent to the source node. This applies to all links from the source node with the exception of the outgoing specialization link to the target node and incoming pattern tree links of the known types "hasProperty",

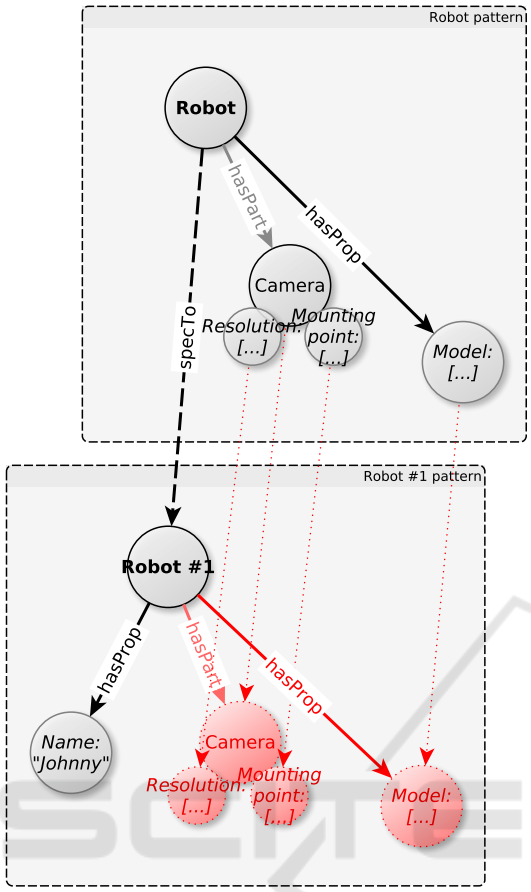

Figure 4: Pattern extension by specialization and inheritance. Patterns that are extensions of existing patterns (in the sense that they have additional properties, parts or elements resp. additional intrapattern relations) are represented via "specializesTo" links. In this example, we have a general pattern "Robot" that describes a specific robot model, and a concrete robot "Robot #1". By the "specializesTo" link, the Memory Net infers that the "Robot #1" concept is a specialization of the concept of "Robot", inheriting all of its pattern features, indicated by the red nodes (these are virtual replicas that do not exist in the network, however during operation the KB treats them as if they would exist). In addition, more specific details can be added to the specialized pattern (e. g. the name property "Johnny").

"hasPart" and "hasElement".

In this way, representations with short description lengths can be gained and synergies can be exploited when matching unknown patterns to available concepts. Fig. 4 shows a general concept pattern and its specialization, as well as an indication of the inherited features of the specialized pattern (red nodes and links).

The same principles apply to partial subpattern and property inheritance. Partial subpattern inheritance occurs when the inheritance of features of the specialized pattern applies to only some branches of the original concept pattern. Property inheritance occurs when the property value specializes, i. e., when the property value of the specialized pattern is a subset of the admissible value(s) of the original pattern. A special case is given when the value of the original pattern is undetermined, which we denote by "{}", in this case any value of the specialized property is a valid specialization. Property inheritance lets you create concepts which describe features that in principle can / should be measured and which might have a certain value range, and the specialized property then having a concrete property value. In this case we speak of truly specialized pattern nodes, as opposed to inherited ones. Fig. 5 shows such a pattern specialization with the different specialization cases. The red nodes and links indicate direct inheritance from the general pattern which are not explicitly represented in the KB but indicated here for explanation. To the contrary, there are truly specialized parts and properties where concrete values of the specialized pattern have been filled, e. g. by sensory measurements.

Specialization in Memory Net can be used to construct an ontology in terms of the classical notion of parent and child classes and instances. However, we do not distinguish classes vs. instances, since for purely measurement-based concepts it is impossible to establish a class vs. instance boundary [1]. Just imagine the concepts of "Dog", "My dog" and "My dog yesterday when I went for a walk with him". Whether "My dog" is an instance (of the class "Dog") or "My dog yesterday ..." is an instance (of a class "My dog") depends on the interpretation of the meaning of the concept "My dog". In standard definitions classes refer to abstract, long-lived stable constructs vs. instances, which are supposed to be concrete exemplars and rather short-lived. Considering that Memory Net concepts ultimately all potentially originate in real-world measurements avoids this problem and the specialization ontologies do not require a class vs. instance decision.

A specialized concept can be derived (inherited resp. specialized) from more than one concepts. This allows for parallel ontologies addressing different specialization aspects. Just consider a KB about persons in a company where there might be a specialization ontology on the associates' competences and another one about their contribution to projects. The corresponding specialization trees would then cross on the specialized concept of a concrete person which then

[1]The deeper underlying reason is the contradiction that arises in classical class / instance ontologies when class concepts are interpreted as categorical prototypes and when they are interpreted as the set of instances that together describe a class.

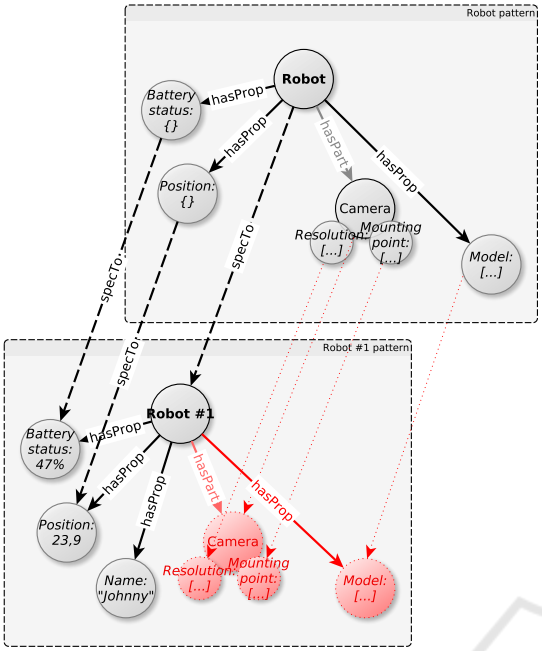

Figure 5: Pattern inheritance with property specialization. In this case, pattern “Robot #1” is specialized from “Robot” in 3 different variants: First, it directly inherits some of “Robot” features which remain valid without modification (red nodes and links). Second, it specializes its concept further by adding new features, in the case the name property with value “Johnny”. And third, it provides concrete values to the measurable properties “Battery status” and “Position”, which from the graph are known to be part of the parent concept “Robot”. The “specializesTo” links between the parent and child properties indicate here that the values of the properties have been specialized.

would simultaneously inherit some competences and participate in some projects.

4.4 Transformation Chains

Over time, in a Memory Net containing items and events registered from the current environment, new measurements constantly flow into the system which have to be interpreted in relation to the existing KB structure. Many of them have a particular property: They are updated measurements of the “same” concept (e. g. describing a real-world object) already registered previously in memory. Processes and structures that support *concept persistence* are therefore important.

For this purpose the link type “transformsTo” is used, which connects two properties or two concept patterns. It expresses the fact that a stored concept undergoes a change to another concept, and it can be used to express concept persistence but also concept transformation. Fig. 6 shows an example of the concept for the “Robot #1” measurement before and after charging its battery. The “transformsTo” links indicate that the pattern and with it some features (in this case the battery status property) are being transformed.

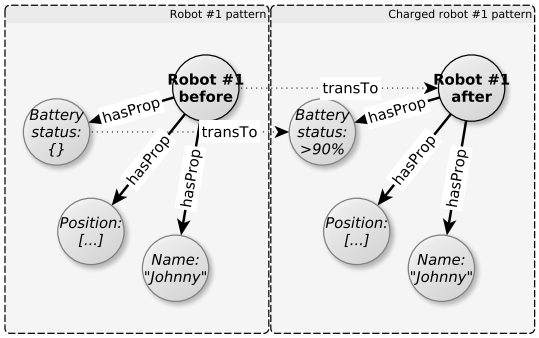

Figure 6: Transformation from one concept pattern to another using the “transformsTo” type links. In this example, concepts of the robot before and after charging are connected bz “transformsTo” links which indicate that the pattern transformation involves the change in its battery status from any ($\{\}$) to $> 90\%$.

Since the transformation usually only applies to a few features of a pattern, the “transformsTo” link can be used in combination with the previously introduced “specializesTo” link to inherit most of the source pattern features to avoid representation redundancy. In addition, if certain transformations occur repeatedly or should be explicitly represented, a **transformation pattern** can be created. Fig. 7 shows such a transformation pattern for the robot example. This can be repeated to gain transformation chains and concatenated transformation patterns, and is an important step towards representing actions and action sequences in an Intelligent Agents’ Knowledge Base.

4.5 Sets of Patterns

A special structure in Memory Nets is that of a **Set**. A Set is a group of objects with similar characteristics or that belong together in some way. Now, when we speak of certain categories or classes of objects, we implicitly and ambiguously do refer not only to the category, but also to the set of objects that make up the category.

On the other hand, there are concepts which explicitly mention the set characteristics of a concept. This is the case e. g. for “fleet”, which is a group of ships that somehow sail together, engage in the same activity, or are under the same ownership - i. e., they are grouped to one thing, and it does not really matter why.

To be able to represent sets, we use the “hasElement” links. Any node with patterns attached to it with

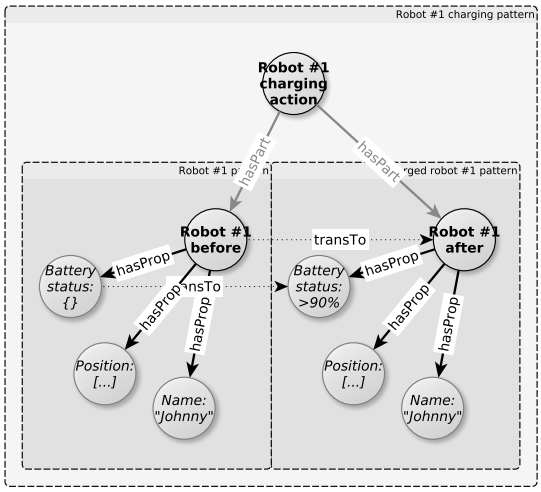

Figure 7: The two patterns from Fig. 6 linked by "transformsTo" relations combined into a prototypical transformation pattern that represents the overall transformation in a way that is accessible and reusable within the KB.

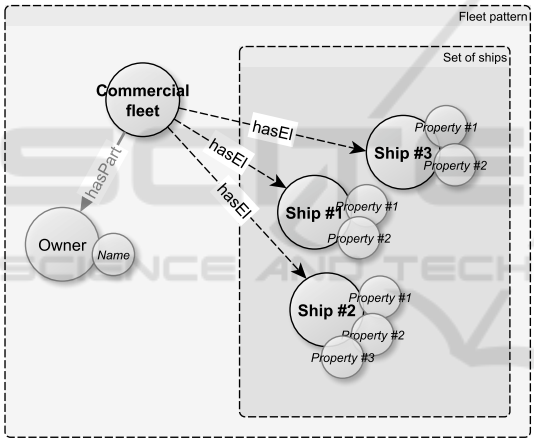

Figure 8: A set is a group of objects that somehow belong together. Nodes with outgoing "hasElement" links acquire the role of a set. In this example we represent the concept of a fleet as a set of ships, with additional properties like its owner.

outgoing "hasElement" links, has the characteristics of a set, so that we can query set-related information from it, like the number of elements in the set. Fig. 8 shows the example of a concept pattern for a concrete "Commercial fleet", with the owner and the single ships as elements that add to the group of ships.

The interesting point is that whereas for the case of a concrete fleet with concrete ships this works, it is not yet sufficient for the definition of a concept of an abstract fleet, where we do not know the concrete ships yet. In this case there are no elements that define the set-based concept. However, it is important that a fleet contains ships or vehicles, but not other things like e. g. animals, so the *types* of elements are characteristic for the set-based concept representation. Conveniently, we can use the introduced specialization and inheritance properties from Memory Nets (section 4.3) together with *element prototypes* for that purpose: If we represent the ships in a compact way by specializing them from some common prototypes (there can be several of them, without limitation), then the prototypes are definitory parts of the fleet concept representation. This is shown in Fig. 9, and we can easily see how the "Commercial fleet" concept representation remains valid even if there are no ship elements in the fleet at all.

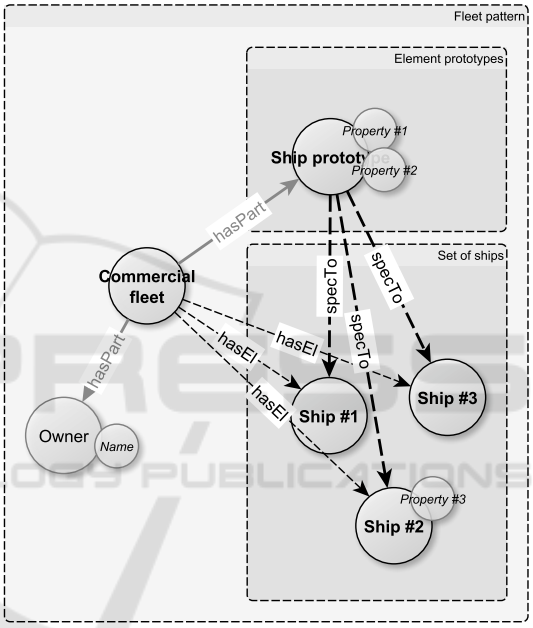

Figure 9: The fleet concept, this time with a compact representation using ship prototypes. The single ships in the set do not define what a fleet is, but the prototypes do (it is relevant that the elements of the fleet are ships and not, say, animals). In this example we have one prototype but there can be several in a set-based concept, and they can even build specialization hierarchies.

5 EXAMPLE IMPLEMENTATION

As an exemplary application, and for future scalability experiments, we implemented a Memory Net as a KB with inference capabilities on top of a graph database (Neo4j, (Neo4j, 2019)). Whereas the graph database provides raw access to nodes, links and attributes, the Memory Net provides functions that operates at the level of concept patterns as outlined in the last section

4 and provides capabilities for dealing with specialization, inheritance and compositionality. In particular, we developed interfaces for creating new patterns with or without inheritance, finding and querying patterns (e. g. by pattern match), inserting a new pattern and resolving missing properties and parts from the parent pattern, as well as rudimentary spatial reasoning.

The IAs in this case are several domestic robots on wheels equipped with various sensors, speaker and an arm with a gripper. In addition, there is an "intelligent room", also equipped with microphones, cameras and loudspeaker.

Each IA is represented within the Memory Net with its sensors and actuators to enable the interaction with the real-world. In addition, each IA gets access to the representation of the other agents, including their current states and possible relations between the agents (including humans as another interacting agent). It also gets basic information of a static spatial representation of its context and ways to operate (e. g. navigate) in this context. The background is that we want to use the Memory Net, to explore problem solving strategies which require cooperative interactions between the agents, therefore inter-agent inference of the corresponding sensor and actuator capabilities.

Each sensor is linked to a description of the abstract sensor measurement concept pattern with properties and value ranges. The actuators are linked via capabilities to executable actions in a similar way. These entry and exit points from the real world to the Memory Nets are specially tagged. Each time a new sensor measurement event is registered, a concrete measurement pattern is specialized and consistently linked to the knowledge graph. Since each measurement is attached to the sensor description of a particular agent, we can then easily infer which aspects have been measured by whom. Using the sensory information, the state of the physical world can be measured and registered at the right places in the Memory Net, whereas the capabilities allow for executing actions which again have an effect on the state of the world.

We equipped the Memory Net with the concepts for waypoints, rooms, persons and entities (identical to IA) that can have several measurable properties like shape, utterance and position. Utterances are understood as a property in form of a the natural language description that could contain the name of an object or a description attached by natural language processing. Position and shape are general purpose properties that allow for spatial reasoning. All those properties are domain independent and each node has a time stamp information.

A special role is given by a concept pattern which describes a capability, and which can be assigned generally to agents (persons or entities). Each capability pattern has a node which provides a reference to an executable function within the system, corresponding to a sort of actuation primitive. For example the "move" capability refers to a function that executes the physical action of moving the robot to a certain place. Fig. 10 depicts the specialization of an entity with its capability. Other capabilities are currently "speak" or "inform person", which recognizes a certain person and provides information to this person. A simple planning framework utilizes these capabilities by taking the state of the world reflected in the KB into account.

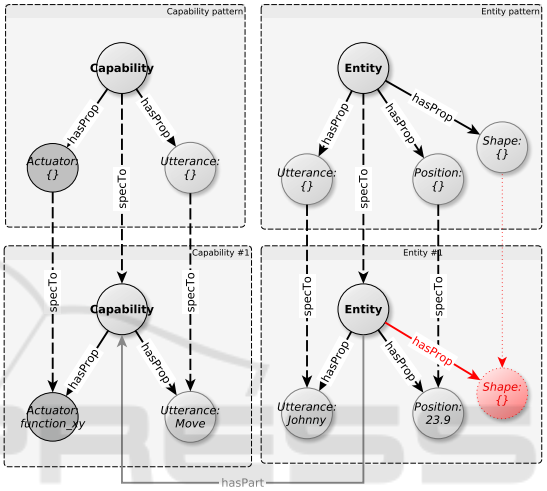

Figure 10: Simplified diagram of the concept patterns for "Capability" and "Entity" within the Memory Net. For concrete entities resp. capabilities, the patterns are specialized. As no shape information has been measured yet about the entity, the specialized pattern still inherits it from its parent. The "Actuator function" property of the abstract capability pattern refers to a specific module that executes a physical action.

The complete Memory Net graph including entities, persons and rooms is depicted in Fig. 11. To illustrate information stored in the graph, we render recent spatial pattern information into a map of our office environment. Fig. 12 shows the positions of persons (red dots) and robots (green triangle). To interact with an IA, we implemented a simple dialog that allows to ask in which spatial concept (e. g. a room) a pattern with a certain utterance (e. g. a certain person) has been seen last time. For example, if the system is asked: "Where did you see Joerg Deigmoeller?", it will answer "Johnny has seen Joerg Deigmoeller in the lobby." , where Johnny is the utterance of the observing robot and lobby is the utterance of the correponding room found by spatial reasoning.

The long-term goal is to use this basic implementation as a starting point and let IAs learn to interact in our office environment. This requires a self-referenced

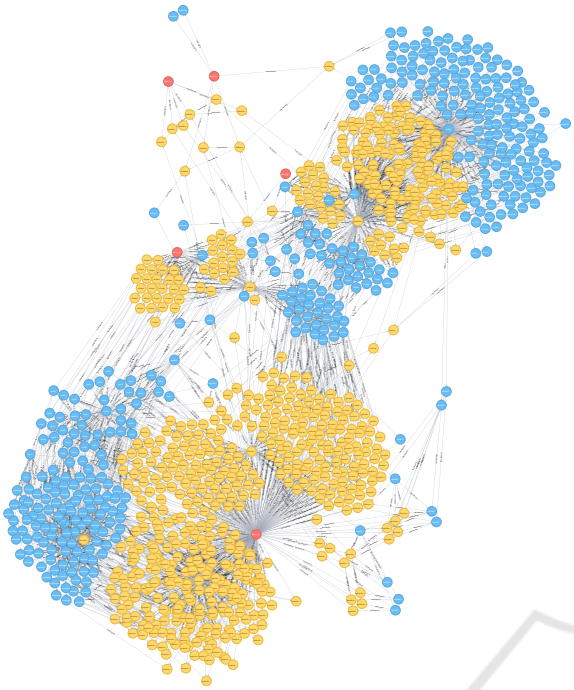

Figure 11: Illustration of whole Memory Net graph with pattern nodes (blue) and property nodes (yellow).

Figure 12: Office map that contains spatial information extracted from the graph about robots (green triangle icon) and persons (red dot icons). The icons are interactive and show the corresponding utterance by clicking on them.

view and situational awareness for a given task. Here, the requirement for a KB is to break down the number of hypothesis that contribute to executing tasks and hence limit the search space for the later planning step. If a goal has been reached, this could be reflected in the KB and serve as individual experience of each IA for future tasks. In other words, we provide IAs with basic patterns that allow to act in the physical world on a high level and leave the assessment of additional own capabilities to the system, which acquires them based on experience. This high level representation leaves room for, on the one hand, enriching the KB by external knowledge sources e. g. from Semantic Web, and, on the other hand, allows for low level machine learning methods to project high dimensional problems on a more abstract, interpretable level.

6 CONCLUSION

We proposed Memory Nets, a new knowledge representation for IAs that mainly encodes semantics in a graph structure and less in natural language identifiers. Memory Nets are defined by a minimal set of link types embedded in patterns with their supporting properties. Such granularity leads to operational semantics and a strong connectedness between nodes - which is the great strength of graphs.

A key component of Memory Nets is the central representation of an IA and its connection through different levels, starting from its own concept, through its capabilities and measurements up to observed patterns and properties. This representation allows for a contextual embedding of a situation an IA is currently in and its operability. An additional important feature is incrementality, that is enabled by possible abstraction from observations or linking of external knowledge sources. Encoding the knowledge in relations and in the amount of connected information also allows for machine learning methods on top of the representation.

Memory Nets facilitates for operations in multiple dimensions like specialization of patterns for class taxonomies, transformation patterns, compositionality (for building partonomies) and sequential ordering.

The future direction of our research is on the one hand using Memory Nets as central backend for multiple physical embodiments of IAs. Another focus is the incrementality of the KB and usage of machine learning approaches.

REFERENCES

Antoniou, G., Groth, P., Harmelen, F. v. v., and Hoekstra, R. (2012). *A Semantic Web Primer*. The MIT Press.

Beetz, M., Beßler, D., Haidu, A., Pomarlan, M., Bozcuoglu, A. K., and Bartels, G. (2018). Knowrob 2.0 – a 2nd generation knowledge processing framework for cognition-enabled robotic agents. In *International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia.

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. A New Form of Web Content that is Meaningful to Computers will Unleash a Revolution of New Possibilities. *Scientific American*, 284(5):34–43.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Collins, A. M. and Quillian, M. R. (1969). Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior*, 8(2):240–247.

Degen, W. and Herre, H. (2001). What is an upper level ontology. In *Workshop on Ontologies*. Citeseer.

Fischer, L., Hasler, S., Deigmoeller, J., Schnuerer, T., Redert, M., Pluntke, U., Nagel, K., Senzel, C., Ploennigs, J., Richter, A., and Eggert, J. (2018). Which tool to use? grounded reasoning in everyday environments with assistant robots. In *Proceedings of the 11th Cognitive Robotics Workshop 2018*, pages 3–10.

Franklin, S. and Graesser, A. C. (1996). Is it an agent, or just a program?: A taxonomy for autonomous agents. In *Intelligent Agents III, Agent Theories, Architectures, and Languages, ECAI '96 Workshop (ATAL), Budapest, Hungary, August 12-13, 1996, Proceedings*, pages 21–35.

Horton, T. E., Chakraborty, A., and Amant, R. S. (2012). Affordances for robots: a brief survey. *AVANT*, 2:70–84.

Kasabov, N. and Kozma, R. (1998). Introduction: Hybrid intelligent adaptive systems. *International Journal of Intelligent Syststems*, 13(6):453–454.

Knublauch, H., Oberle, D., Tetlow, P., Wallace, E., Pan, J., and Uschold, M. (2006). A semantic web primer for object-oriented software developers. *W3c working group note, W3C*.

Kunze, L., Roehm, T., and Beetz, M. (2011). Towards semantic robot description languages. In *2011 IEEE International Conference on Robotics and Automation*, pages 5589–5595.

Kurzweil, R. (1999). *The Age of Spiritual Machines: When Computers Exceed Human Intelligence*. Viking.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al. (2015). DBpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Lemaignan, S., Ros, R., Mösenlechner, L., Alami, R., and Beetz, M. (2010). ORO, a knowledge management platform for cognitive architectures in robotics. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3548–3553. IEEE.

Mazzocchi, S. (2005). Closed world vs. open world: the first semantic web battle. https://web.archive.org/web/20090624113015/http://www.betaversion.org/stefano/linotype/news/91/ [Online; accessed 23-April-2019].

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Neo4j (2019). Neo4j graph platform. https://neo4j.com/product/ [Online; accessed 23-April-2019].

Paulheim, H. (2017). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508.

Pease, A., Niles, I., and Li, J. (2002). The suggested upper merged ontology: A large ontology for the semantic web and its applications. In *Working notes of the AAAI-2002 workshop on ontologies and the semantic web*, volume 28, pages 7–10.

Rebhan, S., Richter, A., and Eggert, J. (2009). Demand-driven visual information acquisition. In *Computer Vision Systems*, Lecture Notes in Computer Science, pages 124–133. Springer, Berlin, Heidelberg.

Röhrbein, F., Eggert, J., and Körner, E. (2009). Child-friendly divorcing: Incremental hierarchy learning in bayesian networks. In *Proceedings of the International Joint Conference on Neural Networks*, pages 2711–2716.

Simmons, R. (1973). Semantic networks: Their computation and use for understanding english sentences. In *Computer Models of Thought and Language*.

Singhal, A. (2012). Introducing the knowledge graph: things, not strings. http://googleblog.blogspot.co.uk/2012/05/introducing-knowledge-graph-things-not.html [Online; accessed 23-April-2019].

Sowa, J. (2000). *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing Co.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706. ACM.

Tenorth, M., Klank, U., Pangercic, D., and Beetz, M. (2011). Web-enabled robots. *IEEE Robotics Automation Magazine*, 18(2):58–68.

Vrandecic, D. and Krötzsch, M. (2014). Wikidata: A free collaborative knowledge base. *Communications of the ACM*, 57:78–85.

Wikipedia (2019). Intelligent agent — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/ Intelligent_agent#CITEREFKasabov1998 [Online; accessed 23-April-2019].

Wood, D., Zaidman, M., Ruth, L., and Hausenblas, M. (2014). *Linked Data: Structured Data on the Web*. Manning Publications, 1 edition edition.

Wyatt, J. L., Aydemir, A., Brenner, M., Hanheide, M., Hawes, N., Jensfelt, P., Kristan, M., Kruijff, G. J. M., Lison, P., Pronobis, A., Sjoo, K., Vrecko, A., Zender, H., Zillich, M., and Skocaj, D. (2010). Self-understanding and self-extension: A systems and representational approach. *IEEE Transactions on Autonomous Mental Development*, 2(4):282–303.