Integrating Ontologies and Planning for Cognitive Systems

Gregor Behnke[1], Pascal Bercher[1], Susanne Biundo[1], Birte Glimm[1], Denis Ponomaryov[2], and Marvin Schiller[1]

[1] Institute of Artificial Intelligence, Ulm University, Germany
[2] A.P. Ershov Institute of Informatics Systems, Novosibirsk, Russia

Abstract. We present an approach for integrating ontological reasoning and planning within cognitive systems. Patterns and mechanisms that suitably link planning domains and interrelated knowledge in an ontology are devised. In particular, this enables the use of (standard) ontology reasoning for extending a (hierarchical) planning domain. Furthermore, explanations of plans generated by a cognitive system benefit from additional explanations relying on background knowledge in the ontology and inference. An application of this approach in the domain of fitness training is presented.

1 Introduction

Cognitive systems aim to perform complex tasks by imitating the cognitive capabilities of human problem-solvers. This is typically achieved by combining specialised components, each of which is suited to fulfil a specific function within that system, such as planning, reasoning, and interacting with the user or the environment. Each component requires extensive knowledge of the domain at hand. Traditionally, several representations of knowledge are used by the different components, each well suited for their respective components. As a result, domain knowledge is distributed across various parts of such a system, often in different formalisms. Thus, redundancy and maintaining consistency pose a challenge.

In this paper, we present an approach for using an ontology as the central source of domain knowledge for a cognitive system. The main issue we address is how the planning domain (representing procedural knowledge) and other (ontological) domain knowledge can be suitably combined. The approach uses the ontology and ontological reasoning to automatically generate knowledge models for different components of a cognitive system, such as the planning component or an explanation facility. In particular, the planning domain is automatically extended using ontological background knowledge. The same ontology is used by an explanation mechanism providing coherent textual explanations for the generated plans and associated background knowledge to the user. The planning component uses Hierarchical Task Network (HTN) planning [7, 9], which is well-suited for human-oriented planning.

This paper is organised as follows. We commence with the relevant preliminaries in Section 2. A general outline of the approach and its foundations is presented in Section 3, illustrated by a case study using a real-world fitness training scenario. In

Section 4, the explanation mechanism is described by using accompanying examples from the case study. Related work is discussed in Section 5 and Section 6 concludes the paper.

2 Preliminaries

In the paper, we refer to the Description Logic $\mathcal{ALC}$ [24], which, as we assume, is familiar to the reader. We briefly introduce the relevant concepts of HTN planning that help to understand the context of our work. HTN planning [7, 9] is a discipline of automated planning where tasks are hierarchically organised; tasks are either "primitive" (they can be executed directly) or abstract (also called complex or compound in the literature), i.e., they must be decomposed into sets of subtasks which can in turn be abstract or primitive. Plans are represented by so-called task networks, which are partially ordered sets of tasks. Planning is done in a top-down manner, such that abstract tasks in a task network are refined stepwise into more concrete courses of action. This approach to planning is deemed similar to human problem solving, and thus considered appropriate for use in cognitive systems [4]. HTN planning problems consist of an initial task network, a planning domain – a set of operators (specifying the preconditions and effects of primitive actions) together with a set of decomposition methods (which specify how each abstract task can be decomposed by replacing it with a network of subtasks) – and an initial state (specifying the state of the world before the tasks in the plan are carried out). A decomposition method m is denoted as $\mathcal{A} \mapsto_{\prec} \mathcal{B}_1, ..., \mathcal{B}_n$, stipulating that the abstract task $\mathcal{A}$ may be decomposed into the plan containing the subtasks $\mathcal{B}_1$ to $\mathcal{B}_n$ adhering to the partial order $\prec$. The subscript $\prec$ is omitted if no order is defined on the subtasks. Applying a method m to a plan containing $\mathcal{A}$ refines it into a plan where $\mathcal{A}$ is replaced by the subtasks of m with the given order $\prec$. All orderings imposed on $\mathcal{A}$ are inherited by its subtasks $\mathcal{B}_i$. A plan is a solution for a planning problem if it consists of a fully executable network of tasks.

3 Integrating Ontologies and Planning

When bringing ontologies and planning together, a key challenge is to find a representation that suitably links general ontological knowledge with information in the planning domain. The aim is to enable both a specialised reasoner and a planner to play to their strengths in a common application domain of interest, while the consistency of the shared domain model remains ensured. The first part of our approach is to embed planning knowledge (a hierarchical planning domain) into a general ontology of the application domain. To address differences in the formalisms of planning and DL, we present suitable modelling patterns and translation mechanisms. As a second step, an off-the-shelf DL reasoner is applied to infer new knowledge about the planning domain in terms of new decomposition methods. Thanks to the integrated model, the structure of decompositions in the planning domain is accessible to DL reasoning, i.e., new methods are implied using knowledge of both domains. This modelling yields a standard planning domain such that an off-the-shelf planner can be used. We have applied this approach to a fitness scenario and have developed a system which creates a training

plan for a user pursuing some fitness objective. Such a plan defines a training schedule, comprising training and rest days, as well as the exercises and their duration which are necessary to achieve a certain goal, e.g., to train the lower body. A similar scenario is used by Pulido et al. [21], where exercises for physiotherapy of upper-limb injuries are arranged by a planner.

3.1 Embedding Planning Methods into Ontologies

In this section we describe how the knowledge contained in a hierarchical planning domain can be represented in an ontology such that its contents are described declaratively and thus become amenable to logical reasoning.

First, a link between the planning domain and corresponding/additional information in the ontology is defined by a common vocabulary – there is a distinguished set of *task concepts* in the ontology which correspond to planning tasks. If $\mathcal{T}$ is a planning task, then a corresponding task concept is denoted as T. In addition to task concepts, the ontology allows for concepts to model further aspects of the application domain. Task concepts are provided with definitions/told subsumptions in the ontology which are determined by the set of predefined decomposition methods for the counterpart planning tasks. Based on these predefined methods, new ones are derived from the ontology, as further described in Section 3.2. A simple decomposition method $\mathcal{A} \mapsto \mathcal{B}$ is interpreted as a subsumption $B \sqsubseteq A$ between task concepts A, B. This pattern, however, works only for such simple decomposition methods. In general, a method has the form $\mathcal{A} \mapsto \mathcal{B}_1, \ldots, \mathcal{B}_n$ and can be viewed as an instruction that $\mathcal{A}$ is achieved by "executing" the tasks $\mathcal{B}_1, \ldots, \mathcal{B}_n$. When representing such decomposition methods in an ontology one needs to take into account that a decomposition method is a precise definition of the tasks created. It specifies the subtasks required to achieve an abstract task, but simultaneously states that these tasks are also *sufficient*. Ontologies, on the other hand, are built on the open world assumption – representing a task decomposition by the tasks $\mathrm{B}_1, \ldots, \mathrm{B}_n$ is insufficient. The ontology must also contain the explicit information that only these tasks belong to the decomposition. For this purpose we use the `onlysome` construct (see, e.g., Horridge et al. [13]), which is a short-hand notation for a pattern combining the existential and universal restrictions to represent collections of concepts.

Definition 1 *Let r be a role, I an index set, and $\{C_i\}_{i \in I}$ concepts. We define the* `onlysome` *restriction $\mathfrak{O}r.(\{C_i\}_{i \in I})$ by $\mathfrak{O}r.(\{C_i\}_{i \in I}) := \bigsqcap_{i \in I} \exists r.C_i \sqcap \forall r.\left(\bigsqcup_{i \in I} C_i\right)$.*

With the `onlysome` restriction we can give a decomposition method in a definition stating that a collection of tasks corresponds to a task concept to be decomposed. Thus, a method $\mathcal{A} \mapsto \mathcal{B}_1, \ldots, \mathcal{B}_n$ can be expressed by using the axiom $\mathrm{A} \equiv \mathfrak{O}\mathrm{includes}.(\mathrm{B}_1, \ldots, \mathrm{B}_n)$.

Let us now turn to our application domain of fitness training. Here, elementary training exercises are represented as planning tasks whose preconditions and effects follow certain rules (e.g. muscles must be warmed up before being exercised, intense exercises must precede lighter ones, …). The ontology further encompasses concepts for training exercises, equipment, workouts, training objectives, and a part of the NCICB corpus [17], describing muscles etc. Contained in the ontology are four planning-related

types of concepts: exercises, workouts, workout templates, and trainings. *Workouts* are predefined (partially ordered) sets of exercises, modelled by a domain expert using `onlysome`-definitions. This implicitly defines decomposition methods for each workout. Workouts are, e.g., defined by:

$$\text{Workout1} \equiv \text{Ɔincludes.}(\text{FrontSquat}, \text{SumoDeadlift})$$

This axiom postulates that Workout1 includes front squats and sumo deadlifts but nothing else. At a more abstract level, *workout templates* serve to specify groups of workouts with similar properties. For example, there exist many similar variants of squats and deadlifts with similar training effects, which can be grouped together. For instance, consider WorkoutTemplate1:

$$\text{WorkoutTemplate1} \equiv \text{Ɔincludes.}(\text{Squat}, \text{Deadlift})$$

This workout template subsumes Workout1 wrt the ontology (since FrontSquat $\sqsubseteq$ Squat and SumoDeadlift $\sqsubseteq$ Deadlift). As detailed in the next subsection, these subsumptions are the basis for generating corresponding decomposition methods, e.g., in this case a method decomposing the task *WorkoutTemplate1* into *Workout1* which again is decomposed into its concrete subtasks. Finally, *trainings* define abstract training objectives, such as improving strength or training the lower body. Here, the requirements that need to be met are formulated using `onlysome` restrictions. For instance, the following axiom postulates that lower body training contains at least one and only exercises targeting muscles in the lower body:

$$\text{LowerBodyTraining} \equiv \text{Ɔincludes.}\exists\text{engages\_target.}(\exists\text{part\_of.LowerBody})$$

The cornerstone of our approach is to establish a correspondence between subsumptions among task concepts in the ontology and corresponding decompositions in the planning domain. For two task concepts representing collections of tasks using `onlysome`, we require that one is subsumed by the other if and only if the tasks defined by the first serve to achieve all requirements specified by the second. In more detail, a collection $\mathfrak{C}_1$ (representing a set of tasks) should be subsumed by a collection $\mathfrak{C}_2$ if and only if for any task concept (requirement) from $\mathfrak{C}_2$, there is a task concept in $\mathfrak{C}_1$, which achieves it (i.e., $\mathfrak{C}_1$ is subsumed by $\mathfrak{C}_2$), and there are only those task concepts in $\mathfrak{C}_1$ that meet some requirement from $\mathfrak{C}_2$. For this property to hold, we require that in `onlysome` restrictions $\text{Ɔ}r.(\{C_i\}_{i\in I})$ the role r is *independent* of concepts C_i, i.e., has no semantic relationship with them, as captured by the following definition.

Definition 2 *Let $\mathcal{O}$ be an ontology, r a role, and $C_1, \ldots, C_n$ concepts. We call r* independent of $C_1, \ldots, C_n$ wrt $\mathcal{O}$ *if, for any model $\mathcal{I}$ of $\mathcal{O}$ and any binary relation $[s]$ on the domain of $\mathcal{I}$, there is a model $\mathcal{J}$ of $\mathcal{O}$ with the same domain such that r is interpreted as $[s]$ in $\mathcal{J}$ and the interpretation of C_i, $1 \leq i \leq n$, in $\mathcal{I}$ and $\mathcal{J}$ coincides.*

Next, we show that the given intuition holds for the collections defined with the `onlysome` restriction.

Theorem 1 *Let $\mathcal{O}$ be an ontology, $C_1, \ldots, C_m$ concepts satisfiable wrt $\mathcal{O}$, $D_1, \ldots, D_n$ concepts, and r a role independent of $C_1, \ldots, C_m, D_1, \ldots, D_n$ wrt $\mathcal{O}$.*

Then it holds $\mathcal{O} \models \Im r.(C_1, \ldots, C_m) \sqsubseteq \Im r.(D_1, \ldots, D_n)$ *if and only if*
(1) $\forall i, 1 \leq i \leq m,\ \exists j, 1 \leq j \leq n$, *such that* $\mathcal{O} \models C_i \sqsubseteq D_j$ *and*
(2) $\forall j, 1 \leq j \leq n,\ \exists i, 1 \leq i \leq m$, *such that* $\mathcal{O} \models C_i \sqsubseteq D_j$.

Proof (Sketch). The if direction can be shown using monotonicity of existential/universal restrictions and Conditions (1) and (2). Using contraposition, we can show the only-if direction by constructing a countermodel for the subsumption. Since each C_i is satisfiable, there are models of $\mathcal{O}$ with some instance c_i of C_i. Using the negation of Condition (1), there is further a model with an instance x of some C_i that is not an instance of any D_j. We now build a new model as the disjoint union of these models and take an arbitrary element d in the constructed model. Using independence of r, we obtain a model in which r contains $\langle d, x\rangle$ and the tuples $\langle d, c_i\rangle$. We obtain the desired contradiction since d is an instance of $\Im r.(C_1, \ldots, C_m)$, but not an instance of $\forall r.(D_1 \sqcup \ldots \sqcup D_n)$ (due to $\langle d, x\rangle$) and, hence, of $\Im r.(D_1, \ldots, D_n)$. We can proceed similarly for the case of Condition (2) not holding. □

Until now, only the tasks contained in a decomposition method have been regarded, while their partial ordering was ignored. To incorporate it in the ontology, the notion of collections must be extended to partially ordered sets of concepts. Unfortunately, many DLs (also $\mathcal{ALC}$) are not well suited to represent partial orders, since their expressivity is limited by the tree-model property [28], stating that non-tree structures cannot fully be axiomatized. Since our aim is to completely represent the planning domain in the ontology, we propose a syntactic encoding for this information that is opaque to DL reasoners and has no influence on the semantics. A task $\mathcal{A}$ following a task $\mathcal{B}$ is expressed by replacing the concept A in `onlysome` expressions by $\mathrm{A} \sqcup (\bot \sqcap \exists \mathrm{after}.\mathrm{B})$. The latter disjunct is trivially unsatisfiable and the given expression is semantically equivalent to just A. This makes order opaque to any reasoner. Preconditions and effects of primitive tasks are modelled using auxiliary roles, making them accessible for logical reasoning, too.

3.2 Extending Planning Domains by DL Inference

Embedding the planning domain into the ontology enables us to infer new decomposition methods using off-the-shelf DL reasoners. More precisely, subsumption relations between task concepts inferred from an ontology $\mathcal{O}$ result in decomposition methods being added to the planning domain. Suppose there are task concepts A and B such that $\mathcal{O} \models \mathrm{B} \sqsubseteq \mathrm{A}$ and there is no other task concept C such that $\mathcal{O} \models \{\mathrm{B} \sqsubseteq \mathrm{C}, \mathrm{C} \sqsubseteq \mathrm{A}\}$. Then, a decomposition method $\mathcal{A} \mapsto \mathcal{B}$ is created in analogy to the way such methods are encoded in the ontology. This simple scheme provides only for methods decomposing into a single subtask. Further, we interpret `onlysome`-definitions provided by the ontology as told decompositions that provide collections of tasks. Besides these two simple cases, we are also interested in knowing whether an abstract task $\mathcal{A}$ can be achieved by combining some task concepts $\mathrm{B}_1, \ldots, \mathrm{B}_n$ into a new decomposition method $\mathcal{A} \mapsto \mathcal{B}_1, \ldots, \mathcal{B}_n$. If so, some concept expression E describing this combination should be subsumed by A. In keeping with the principle of matching task collections described by Theorem 1, we consider combining concepts using `onlysome` restrictions.

The concepts $\mathrm{B}_1, \ldots, \mathrm{B}_n$ describe requirements, which another concept A might fulfil. If for some tasks $\mathcal{A}, \mathcal{B}_1, \ldots, \mathcal{B}_n$ it holds $\mathcal{O} \models \mathfrak{O}r.(B_1, \ldots, B_n) \sqsubseteq A$, then a decomposition method of $\mathcal{A}$ into the collection $\mathcal{B}_1, \ldots, \mathcal{B}_n$ is created. Let us once again consider our use-case to discuss an example. Consider the definition of lower body training introduced in the previous subsection:

$$\text{LowerBodyTraining} \equiv \mathfrak{O}\text{includes}.\exists\text{engages\_target}.(\exists\text{part\_of.LowerBody})$$

Since our ontology respects the requirement that the role includes is independent of concepts occurring in `onlysome` expressions, Theorem 1 applies and we know that any workout solely comprised of lower body exercises is subsumed by LowerBodyTraining. This results in decomposition methods for LowerBodyTraining into every possible workout for the lower body. Furthermore, consider the following definition of full body training:

$$\begin{aligned}\text{FullBodyTraining} \equiv \mathfrak{O}\text{includes}.(&\\ &\exists\text{engages\_target}.(\exists\text{part\_of.LowerBody}),\\ &\exists\text{engages\_target}.(\exists\text{part\_of.UpperBody}))\end{aligned}$$

Using reasoning, one can now establish whether combinations of different workouts achieve such a training objective, such that a corresponding decomposition method is automatically introduced into the planning domain. For instance, we can infer that a workout solely training the lower body and a workout solely training the upper body in combination constitute this training. Hence a decomposition method for a full body training into these two workouts is added to the planning domain. The following theorem clarifies the relationship between the obtained decomposition methods and entailed concept inclusions.

Theorem 2 *Let $\mathcal{A}$ be an abstract task, which can be refined into some plan $\mathcal{P}$ by applying decomposition methods created from the ontology $\mathcal{O}$. Let the plan $\mathcal{P}$ contain the tasks $\mathcal{B}_1, \ldots, \mathcal{B}_n$, $n \geq 1$. Then there is a concept P such that $\mathcal{O} \models P \sqsubseteq A$, $B_1, \ldots, B_n$ occur in P, and P is either of the form:*

1. *a task concept from $\mathcal{O}$, or*
2. *an expression $\mathfrak{O}r.(F_1, \ldots, F_m)$ with each F_i a concept of the form 1 or 2.*

Proof. The claim is proved by induction on the number m of (decomposition) steps made to obtain $\mathcal{P}$ using $\mathcal{O}$. In the induction base, for $m = 0$, A is the required concept. For $m > 0$, let $\mathcal{P}' = \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$ be a set of tasks obtained in $m - 1$ decomposition steps and let P' be a concept satisfying the claim for $\mathcal{P}'$. Let $\mathcal{P}$ be obtained from $\mathcal{P}'$ by decomposing some task $\mathcal{C}_i$, for $i \in \{1 \ldots, k\}$. Assume that a decomposition method for $\mathcal{C}_i$ was created from the concept inclusion $D \sqsubseteq C_i$ entailed by $\mathcal{O}$. Then D is one of the task concepts $B_1, \ldots, B_n$, we have $\mathcal{O} \models P'_{C_i/D} \sqsubseteq P'$, hence, $\mathcal{O} \models P'_{C_i/D} \sqsubseteq A$ and thus, $P'_{C_i/D}$ is the required concept (denoting P' with every occurrence of C_i substituted with D). If a decomposition method for $\mathcal{C}_i$ was created from a concept inclusion $\mathfrak{O}r.(D_1, \ldots, D_\ell) \sqsubseteq C_i$ entailed by $\mathcal{O}$ (and possibly obtained from an axiom $C_i \equiv \mathfrak{O}r.(D_1, \ldots, D_\ell)$), then every D_j, $j = 1, \ldots, \ell$, is a task concept among $B_1, \ldots, B_n$ and we have $\mathcal{O} \models P'_{C_i/\mathfrak{O}r.(D_1,\ldots,D_\ell)} \sqsubseteq A$, so $P'_{C_i/\mathfrak{O}r.(D_1,\ldots,D_\ell)}$ is the required concept. □

Taking into account all possible combinations of tasks in the ontology presents a problem, since there are exponentially many. We propose a pragmatic solution. First, the maximal number of task concepts to be combined in `onlysome` expressions can be restricted by some number k. Second, most real-world domains (including our application example) have restrictions on which tasks can be combined. In our case study using the fitness training scenario, we only considered combinations of two task concepts defined by an `onlysome` axiom. This already enabled a considerable number of methods to be inferred.

Essentially, the ontology used in our system consists of two parts $\mathcal{O}_1$ and $\mathcal{O}_2$, with $\mathcal{O}_1$ containing definitions as above and $\mathcal{O}_2$ representing the core knowledge about the subject domain: exercises (being task concepts), training equipment, body anatomy, etc. The ontology is built in such a way that it guarantees independence of the role includes from any concept occurring under $\exists$includes, which means that the principle of matching task collections outlined in Section 3.1 correctly applies. The general shape of our ontology is formally described in the following theorem, where the role includes is abbreviated as r.

Theorem 3 *Let r be a role and $\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_2$ an ontology such that $\mathcal{O}_1$ is an acyclic terminology consisting of definitions $A \equiv \exists r.(C_1, \ldots, C_m)$, where r does not occur in $C_1, \ldots, C_m$ and A, r do not occur in $\mathcal{O}_2$. Then the role r is independent of any concepts appearing under $\exists r$ in $\mathcal{O}_1$.*

Proof (Sketch). Let $\mathcal{A}$ be the set of concept names occurring on the left-hand side of axioms in $\mathcal{O}_1$. Let $\mathcal{I}$ be a model of $\mathcal{O}$ and $[s]$ a binary relation on the domain of $\mathcal{I}$. Let $\mathcal{I}'$ be an interpretation obtained from $\mathcal{I}$ by changing the interpretation of r to $[s]$. Since r and $\mathcal{A}$-concepts do not occur in $\mathcal{O}_2$, we have $\mathcal{I}' \models \mathcal{O}_2$. It remains to consider an expansion of the terminology $\mathcal{O}_1$ (cf. [2, Prop. 2.1]) to verify that there exists a model $\mathcal{J}$ of $\mathcal{O}_1 \cup \mathcal{O}_2$ obtained from $\mathcal{I}'$ by changing the interpretation of $\mathcal{A}$-concepts (and leaving the interpretation of other symbols unchanged). For any concepts $C_1, \ldots, C_m$ appearing under $\exists r$ in $\mathcal{O}_1$, their interpretations in $\mathcal{J}$ and $\mathcal{I}$ coincide, which shows the required statement. □

The initial planning domain of our case-study scenario encompasses 310 different tasks and a few methods. Its formalisation in the ontology consists of 1 230 concepts and 2 903 axioms, of which 613 concepts and 664 axioms are imported from the NCICB corpus. Initially, 9 different training objectives and 24 workout templates are specified. For extending the ontology with inferred decompositions, we employ the OWL reasoner FaCT++ [27], which requires 3.6 seconds on an up-to-date laptop computer (Intel® Core™ i5-4300U). After being extended, the planning domain contains 471 tasks and 967 methods, of which 203 are created based on subsumptions between workouts and workout templates, and further three methods have been created by `onlysome`-combinations of task concepts. In addition, 59 decomposition methods for training objectives into workout templates are created of which 24 are `onlysome`-combinations of task concepts.

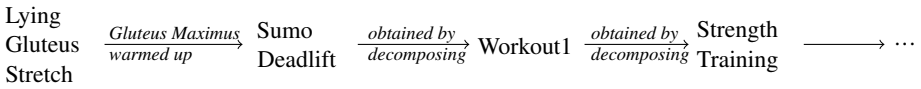

Fig. 1. Example of a plan explanation for the task *lying gluteus stretch*

4 Explanations

Cognitive systems that interact with users need to be able to adequately communicate their solutions and actions. In the field of HCI and dialog modelling, it was shown that systems that provide additional explanations receive increased trust from their users [16, 20]. Bercher et al. [3] empirically investigated the role of plan explanations in an interactive companion system in a real-world scenario.

In our integrated approach, both DL reasoning and planning work together, thereby using the procedural and declarative information contained in the integrated knowledge model, and generating new information artefacts of different flavours (in particular, inferred facts and refined plans). We now describe how explanations are generated from this information by combining techniques from plan explanation (specifically, an approach for explaining hybrid plans [25]) and an approach for explaining ontological inferences [23]. Together they offer complimentary views on a given application domain.

Plan explanation focuses on dependencies between tasks in a plan, in particular how tasks are decomposed into subtasks, and how these tasks are linked by preconditions and effects. To provide an explanation why a particular task is part of a generated plan, the information relevant to justify its purpose is extracted from the causal dependencies and the decompositions applied to the plan. Technically, this information (which internally is represented in a logic formalism) is considered the "explanation". It is guaranteed that plan explanations are always linear chains of arguments, each based on its predecessor. For instance, consider that in our application scenario, the user asks to justify a particular action, for example "Why do I have to do a lying gluteus stretch?" Fig. 1 shows the dependencies that establish the use of the lying gluteus stretch within a plan that achieves a strength training. This information is further converted into text to be communicated to the user by using simple templates. Causal dependencies between two tasks A and B, where A provides a precondition l for B, are verbalised as "A is necessary as it ensures that l, which is needed by B." Similarly, decompositions are justified by patterns such as "Task A is necessary, since it is part of B." In the running example, this yields:

```
The lying gluteus stretch is necessary as it ensures that
the gluteus maximus is warmed up, which is needed by the sumo
deadlift. The sumo deadlift is necessary, since it is part
of the Workout1. The Workout1 is necessary, since it is part
of the strength training. ...
```

The mechanism presented so far represents the part of "traditional" plan explanation. Here, method decompositions are treated as facts (e.g. "Workout1 is part of strength

$$R_{\sqsubseteq\text{Def}}\ \frac{X \sqsubseteq Y \qquad Y \equiv Z}{X \sqsubseteq Z}\ \begin{array}{l}\text{Y is an}\\ \text{atomic}\\ \text{concept}\end{array}$$

"... $\langle\!\langle X \sqsubseteq Y \rangle\!\rangle$. Thus, $\langle\!\langle X \sqsubseteq Z \rangle\!\rangle$ according to the definition of $\langle\!\langle Y \rangle\!\rangle$."

Fig. 2. Sample inference rule (left) together with corresponding text template (right). Guillements indicate the application of a function translating DL formulas into text.

training"); however, in our approach, they can be justified further, since they correspond to subsumptions inferred from background knowledge in the ontology. Therefore, the reasoning behind the subsumption can be used as an explanation to justify the decomposition. For example, the user may ask the question "Why is Workout1 a strength training?" For this purpose, we use the second explanation mechanism, which has been implemented as a prototype. Its aim is to generate stepwise textual explanations for ontological inferences. In the running example, it outputs:

```
According to its definition, Workout1 includes front squat
and sumo deadlift. Furthermore, since sumo deadlift is an
isotonic exercise, it follows that Workout1 includes an isotonic
exercise. Given that something that includes an isotonic
exercise has strength as an intended health outcome, Workout1
has strength as an intended health outcome. Thus, Workout1
is a strength training according to the definition of strength
training.
```

To generate such explanations, first a consequence-based inference mechanism is used to construct a derivation tree for a given subsumption. This is done in two stages. First the relevant axioms in the ontology (the "justifications") are identified using the implementation provided by Horridge [12], which is done efficiently using a standard tableau-based reasoner. Then, a (slower) proof search mechanism using consequence-based rules is applied for building a derivation tree. This tree is then linearised to yield a sequence of explanation steps. The ordering of the explanation steps corresponds to a post-order traversal in the tree structure of inference rule applications (where the inference step that yields the conclusion is taken to be the root). As an example of a consequence-based inference rule used for explanation generation and its corresponding text template, consider Fig. 2. This template generates the last statement in the sample text shown above. Note that even though the presented inference rule is quite simple, it represents a (logical) shortcut, since the conclusion $X \sqsubseteq Z$ could also be obtained in two steps by inferring $Y \sqsubseteq Z$ from the equivalence axiom and then using the transitivity of $\sqsubseteq$. For the generation of explanations, this (logically redundant) rule is given precedence over the "standard" inference rules, in the interest of smaller proofs and greater conciseness of the generated text. By contrast, some inference rules are specified to never generate output, since it would be considered uninformative for a user. Consider the rule deriving $X \sqsubseteq (Y \sqcup Z)$ from $X \sqsubseteq Y$, which is always ignored during text generation. Such considerations provide ample scope for further work into adjusting the generated texts according to pragmatics and user preferences.

To answer the general question how well people understand automatically generated verbalisations of ontological axioms and inferences, studies have already been per-

formed as part of related work. For example, in experiments, Nguyen [18] found that the understandability of verbalised inference steps depends on the employed inference rules, where some kinds of inference rules were found considerably more difficult than others. Furthermore, if the more difficult-to-understand inference rules were verbalised in a more elaborated manner, understanding was improved. Such work hints at a general challenge for empirical evaluations of the general "usefulness" of such verbalisations (to be addressed as future work); their accessibility partially depends on the complexity of the formalised domain and on the prerequisites of the user.

5 Related Work

Past research on coupling ontological reasoning and planning mainly addressed the aim of increasing expressivity/efficiency. There is a large body of research on integrating ontology and action formalisms, see e.g., the overview in Calvanese et al. [6], which aims at bringing together the benefits of static and dynamic knowledge representation. Gil [10] provides a survey of approaches joining classical planning and ontologies and names a number of planners that use ontological reasoning to speed-up plan generation. Hartanto and Hertzberg [11] use a domain-specific ontology to prune a given HTN model. However, additional content in the domain can not be inferred in their paradigm. A number of approaches use ontologies to enrich the structure of the planning domain. Typically, ontologies provide hierarchies of tasks and plans and are often used to represent states under the open world assumption [22, 26]. For a survey, we refer to Sirin [26, Chapter 8]. Further approaches (e.g. [14, 8]) use OWL as a representation language for HTN planning domains but do not employ ontology reasoning to extend these domains. Sirin [26] describes HTN-DL, combining HTN and description logics to solve Web Service composition problems. HTN-DL planning domains are encoded in an ontology by representing tasks as concepts and decomposition methods as individuals. Both are augmented with axioms describing their preconditions and effects. Here Sirin's view of methods differs from standard HTN, as they define a partially ordered list of actions, but not an abstract task they decompose. Further, preconditions and effects are assigned to methods, which are not necessarily related to the contents of the plan. Although Sirin's and our approach are similar in the idea of using an ontology and DL reasoning to generate planning domains, there are conceptual differences. In HTN-DL all decomposition methods are provided in the domain by a modeller, while the presented approach infers new decomposition methods. Sirin applies reasoning to determine whether a decomposition method can be applied to an abstract task, using their preconditions and effects. His approach does not allow inference based on the tasks in a method nor their decompositions or other properties, which is the cornerstone of ours.

Related work on the generation of explanations from ontologies encompasses a number of approaches to verbalise the formalised contents with the goal of imitating natural language. Systems targeting non-expert users include, e.g., the *NaturalOWL* system [1] and the *ontoVerbal* verbaliser [15]. By contrast, the generation of explanations for entailments is addressed only by some approaches, e.g., by Horridge [12], whose approach targets expert users. The generation of stepwise explanations of entailments for non-expert users is considered by Borgida et al. [5], Nguyen et al. [19, 18] and

Schiller and Glimm [23], whose work provided a basis for the approach to explanations presented in this paper.

6 Conclusion

We presented an approach to integrating hierarchical planning knowledge into ontologies that encompass a general representation of the application domain. A central issue of this paper is to establish the semantic correspondence between the constructs of the planning domain (in particular, decomposition methods) and their representation in the ontology. Our results enable a cognitive system to use a coherent knowledge model for both planning and reasoning, which at the same time enables coherent and detailed explanations for the user, as demonstrated in the application scenario. Our investigation also highlights avenues for future work. One such topic is incorporating mixed-initiative planning into the approach, such that communication with and participation of the user would benefit from explanations by the system. Second, the explanations generated by the system raise the issue of selecting the right level of verbosity. Future work should address this from the viewpoint of pragmatics (e.g. that explanations for "obvious" inferences should generally be omitted) and user modelling (e.g. taking prior knowledge of the user into account). While our approach enables the hierarchical structure of a planning domain to be exploited by DL reasoning, the partial order of tasks is not amenable to DL reasoning. The question how this limitation could be addressed can be taken up by future work.

Acknowledgements. This work was done within the Transregional Collaborative Research Centre SFB/TRR 62 "A Companion-Technology for Cognitive Technical Systems" funded by the German Research Foundation (DFG).

Bibliography

[1] Androutsopoulos, I., Lampouras, G., Galanis, D.: Generating natural language descriptions from OWL ontologies: The NaturalOWL system. JAIR 48, 671–715 (2013)

[2] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)

[3] Bercher, P., Biundo, S., Geier, T., Hoernle, T., Nothdurft, F., Richter, F., Schattenberg, B.: Plan, repair, execute, explain - How planning helps to assemble your home theater. In: Proc. of the Int. Conf. on Automated Planning and Scheduling. pp. 386–394. AAAI Press (2014)

[4] Biundo, S., Bercher, P., Geier, T., Müller, F., Schattenberg, B.: Advanced user assistance based on AI planning. Cognitive Systems Research 12(3-4), 219–236 (2011), Special Issue on Complex Cognition

[5] Borgida, A., Franconi, E., Horrocks, I.: Explaining ALC subsumption. In: Proc. of the European Conf. on Artificial Intelligence. pp. 209–213. IOS Press (2000)

[6] Calvanese, D., De Giacomo, G., Montali, M., Patrizi, F.: Verification and synthesis in description logic based dynamic systems. In: Proc. of the 7th Int. Conf. on Web Reasoning and Rule Systems (RR 2013). LNCS, vol. 7994, pp. 50–64. Springer (2013)

[7] Erol, K., Hendler, J.A., Nau, D.S.: Complexity results for HTN planning. Annals of Mathematics and Artificial Intelligence 18(1), 69–93 (1996)

[8] Freitas, A., Schmidt, D., Panisson, A., Meneguzzi, F., Vieira, R., Bordini, R.H.: Semantic representations of agent plans and planning problem domains. In: Engineering Multi-Agent Systems, pp. 351–366. Springer (2014)

[9] Geier, T., Bercher, P.: On the decidability of HTN planning with task insertion. In: Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011). pp. 1955–1961. AAAI Press (2011)

[10] Gil, Y.: Description logics and planning. AI Magazine 26(2), 73–84 (2005)

[11] Hartanto, R., Hertzberg, J.: Fusing DL reasoning with HTN planning. In: KI 2008: Advances in Artificial Intelligence. LNCS, vol. 5243, pp. 62–69. Springer (2008)

[12] Horridge, M.: Justification Based Explanations in Ontologies. Ph.D. thesis, University of Manchester, Manchester, UK (2011)

[13] Horridge, M., Drummond, N., Goodwin, J., Rector, A., Stevens, R., Wang, H.: The Manchester OWL syntax. In: Proc. of the Workshop on OWL Experiences and Directions. Athens, GA, USA (2006)

[14] Ko, R.K.L., Lee, E.W., Lee, S.G.: Business-OWL (BOWL) – A hierarchical task network ontology for dynamic business process decomposition and formulation. IEEE Transactions on Services Computing 5(2), 246–259 (2012)

[15] Liang, S.F., Scott, D., Stevens, R., Rector, A.: OntoVerbal: a generic tool and practical application to SNOMED CT. Int. J. of Advanced Computer Science and Applications 4(6), 227–239 (2013)

[16] Lim, B.Y., Dey, A.K., Avrahami, D.: Why and why not explanations improve the intelligibility of context-aware intelligent systems. In: Proc. of the SIGCHI Conf. on Human Factors in Comp. Systems. pp. 2119–2128 (2009)
[17] NCICB, N.: (2015), http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl (accessed February 9, 2015)
[18] Nguyen, T.A.T.: Generating Natural Language Explanations For Entailments In Ontologies. Ph.D. thesis, The Open University, Milton Keynes, UK (2013)
[19] Nguyen, T.A.T., Power, R., Piwek, P., Williams, S.: Predicting the understandability of OWL inferences. In: The Semantic Web: Semantics and Big Data, LNCS, vol. 7882, pp. 109–123. Springer (2013)
[20] Nothdurft, F., Richter, F., Minker, W.: Probabilistic human-computer trust handling. In: Proc. of the Annual Meeting of the Special Interest Group on Discourse and Dialogue. pp. 51–59. Association for Computational Linguistics (2014)
[21] Pulido, J.C., González, J.C., González-Ferrer, A., García, J., Fernández, F., Bandera, A., Bustos, P., Suárez, C.: Goal-directed generation of exercise sets for upper-limb rehabilitation. In: Proc. of the Workshop on Knowledge Engineering for Planning and Scheduling (2014)
[22] Sánchez-Ruiz, A.A., González-Calero, P.A., Díaz-Agudo, B.: Abstraction in knowledge-rich models for case-based planning. In: Case-Based Reasoning Research and Development, LNCS, vol. 5650, pp. 313–327. Springer (2009)
[23] Schiller, M., Glimm, B.: Towards explicative inference for OWL. In: Proc. of the Int. Description Logic Workshop. vol. 1014, pp. 930–941. CEUR (2013)
[24] Schmidt-Schauss, M., Smolka, G.: Attributive concept descriptions with complements. AIJ 48, 1–26 (1991)
[25] Seegebarth, B., Müller, F., Schattenberg, B., Biundo, S.: Making hybrid plans more clear to human users – a formal approach for generating sound explanations. In: Proc. of the Int. Conf. on Automated Planning and Scheduling. pp. 225–233. AAAI Press (2012)
[26] Sirin, E.: Combining Description Logic Reasoning with AI Planning for Composition of Web Services. Ph.D. thesis, University of Maryland at College Park (2006)
[27] Tsarkov, D., Horrocks, I.: Fact++ description logic reasoner: System description. In: Proc. of the Third Int. Joint Conf. on Automated Reasoning (IJCAR). pp. 292–297. Springer (2006)
[28] Vardi, M.Y.: Why is modal logic so robustly decidable? In: Descriptive Complexity and Finite Models. vol. 31, pp. 149–184. American Mathematical Society (1997)