KickOffTUG - Team Description Paper 2011

Stephan Gspandl, Andreas Hechenblaickner, Michael Reip, Máté Wolfram, and Christoph Zehentner

Institute for Software Technology, Graz University of Technology, Austria,
sgspandl@ist.tugraz.at

Abstract. This paper describes our continous effort to automatically generate behavioral models for agents and robots. Our previous research into sketch recognition and interpretation is extended by concepts to populate natural language descriptions into a domain ontology. This research is embedded in the scope of an ontology lifecycle in RoboCup.

1 Introduction

In every RoboCup league domain knowledge is necessary for the agents or robots to perform intelligent tasks. Many teams use expert literature for the purpose of transforming soccer tactics into specific models, plans or source code. Unfortunately, this is a very time-consuming and laborious job which yields many hybrid descriptions of the same concepts.
As ontologies store data in a structured way under a semantic context, they provide a suitable basis for knowledge-based agents to operate on. To face the above problems, we strongly argue for an automation of knowledge generation by transforming arbitrary knowledge artefacts (like images and text) from books, websites or other sources into an ontology the agents can execute.
Take the following tactic description in natural language from [1] accompanied by the sketch shown in Fig. 1 as example. Both yield valuable information for a team of soccer agents.

```
* 5 passes the ball to 9 (who has come towards him) and moves
  diagonally to get the return pass;
* 9 passes the ball back to 5; ...
```

This tactic will also serve as running example throughout the paper. It is easy for humans to understand it as they are acquainted with soccer, graphs and language. They can easily ground the words and components of the sketch in the soccer domain. The ideas described in this paper follow the same intuition: tactic sketches and descriptions should be processed and grounded in the RoboCup soccer domain using a domain ontology as well as Sketch Recognition/Interpretation and Natural Language Processing (NLP) techniques. The resulting populated ontology can then be employed as knowledge base for the agents in the environment and should be verified and improved during gameplay.
The efforts in this paper cover the pre-processing of sketches and text to enable the population of a soccer ontology and its deployment in the 2D RoboCup Simulation League.

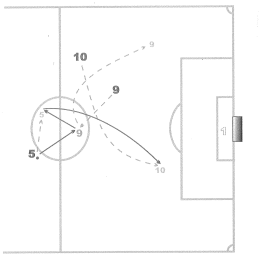

Fig. 1. A sketch accompanying the instructions from [1].

2 Related Research

Ontologies [2] in software engineering are employed to provide a semantic level to a set of data models. Their main purpose is to query the database, perform reasoning on it as well as to present it to and prepare it for users, or to provide a knowledge base for software (knowledge agents). Naturally, there has been effort to employ ontologies in multi-agent systems.
Commonly, in most knowledge engineering applications, the users (or knowledge engineers) have to encode the required knowledge or employ large and bulky ontologies not specifically designed for the special domain of application. Thus, one has to accept many drawbacks. In order to keep the knowledge engineering as intuitive, simple and therefore little time-consuming as possible, multiple concepts have to be combined into an integrated process.
A system as proposed which is able to support the complete knowledge engineering process, has to deal with various fields of ontology research: (1) automatic or semi-automatic creation and management of a domain ontology [3, 4], (2) population of ontologies from various sources [5, 6], (3) grounding the ontology knowledge [5, 7], (4) adopting the populated ontology for deployment in the specific multi-agent system [8], as well as (5) refinement, improvement, enrichment and verification of the ontology in the environment [5].

3 The Ontology Lifecycle

Related research yielded an integrated process (shown in Fig. 2) dealing with all necessary steps to automatically generate a populated ontolgy which can be deployed in a multi-agent system and is able to self-optimize in its environment. We call this the Ontology Lifecycle.
In the first stage of the ontology lifecycle, knowledge artifacts are pre-processed to provide more structured data for the remaining steps. This data will then be interpreted and grounded onto a domain ontology. This means that the ontology is populated with concrete concepts derived from the data. The domain ontology may be generated out of suitable data, but is created manually for our tests. The resulting populated ontology can then be deployed in a concrete environment.

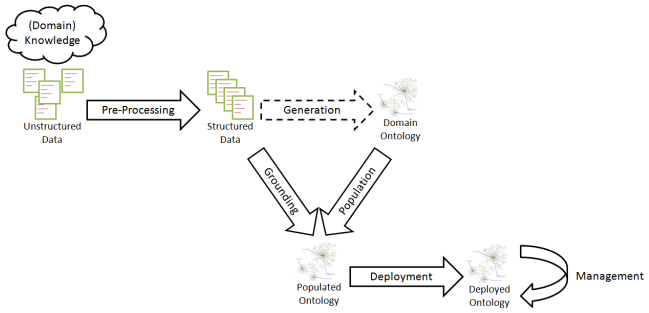

Fig. 2. An integrated process for automatic transformation of knowledge and its deployment in multi-agent systems.

The process is concluded by concepts to verify, enrich and improve this basic ontology therein.
We have implemented this process up to the population and deployment step. The 2D RoboCup Simulation League and our team KickOffTUG serve as environment for the experiments. Input to the ontology generation are the sketches and descriptions of tactics from [1].

4 Populating and Deploying a RoboCup Ontology

The automatic population proposed in this paper builds upon a domain ontology. It contains general soccer concepts relevant for games in the RoboCup Soccer Simulation League as well as more specific entities for the pre-processing: players and player types, field components, positions and the tactic model. This model comprises a pre-condition (under which the tactic should be executed), a set of actors (participating in the tactic) and a compound of actions each player has to perform, where actions can be executed in sequence or in parallel. As the textual input does not convey information on when a team should perform a certain tactic, the pre-condition can only be derived from the sketch.

4.1 Pre-Processing

Before a description can be actually populated on the base of and into a domain ontology, it has to be pre-processed to reduce the complexity of grounding correctly in the semantics of the ontology. This pre-processing is specific to the type of knowledge artefact.

In order to prepare a sketch for the population of the domain ontology [9], it has first to be transformed into a domain-specific quantitative model: For this purpose the primitives of the sketch, i.e. circles, boxes, etc., are first detected domain-independently and domain-specifically in a second step. For example, a circle is then mapped to an agent or a certain box to the penalty area. The

resulting set of entities in quantitative relation is then transformed into a qualitative description of the sketch. In our experiments we used the already available set of predicates of our team KickOffTUG for this transformation.

The pre-processing of text combines state-of-the-art NLP techniques: First, the text is split into atomic actions by stop-words and stop-chars and the Stanford POS (part-of-speech) tagger is used to assign the lexical category to each word. Now, irrelevant words or categories are removed. The resulting atomic sentence stubs are transformed into a tuple $\{S, P, O\}$ (simply referred to as SPO; with S being the subject, P the predicate and O the object of the sentence). With subject and object identified, pronouns can now be exchanged by their corresponding entity.
As the descriptions of the tactics are not linear in time and involve parallel activities as well, temporal adjustments are necessary. By evaluating the tenses and identifying jumps as well as using keywords in the original sentence stubs, a timestamp is thus assigned to each SPO.
In the last step of the pre-processing, synonyms are resolved so that different words or phrases with the same meaning are mapped onto the same semantic concept in the domain ontology. This is done using WordNet in supervised or unsupervised mode depending on the size of the vocabulary.

4.2 Population

For the purpose of population, test sequences are generated on the base of the ontology's taxonomy (this concept is easily extended to follow the restrictions as well). That is, the root classes corresponding to each element in the SPO tupel are retrieved from the ontology and specialized test strings are concatenated from simple identifiers (names or specializing qualifiers) associated with each class while visiting the subclass relations. This way, each test string is automatically associated with the corresponding ontology class. The test sequences are then generated as all possible combinations of the subject, predicate and object test strings, yielding for example '5 run left wing'.
Each SPO from the pre-processing is then compared to each test sequence by an adapted Levenshtein distance on word level. The test sequence with the highest equality is saved as target concept. The SPOs are then traversed to create an action compound for the new tactic in the ontology: for each action a new subclass is derived and the restrictions to subject and object class are set. These classes are put in temporal context depending on the timestamp. The result is the action compound which is associated with the tactic. In order to complete the tactic, it is connected to the set of participating actors and the pre-condition as derived from the corresponding sketch. As the descriptions contain implicit semantics, an additional set of rules is provided to fill these semantic gaps.

4.3 Deployment

After successfully transforming the tactics descriptions into a populated ontology, execution is rather straightforward. Before it can be deployed, though, it has

to be embedded properly into its execution environment by defining semantics for execution. Consider the first two action stubs of the running example. Two questions arise here: (1) what methods the agents should call when executing the ontology action associated with each verb, and (2) when exactly player 5 should pass the ball. We thus have to (1) define how the ontology actions map onto methods and (2) provide rules concretizing the behavior of the actions, for example that 'come towards' makes a player move halfways to another.
On the base of these semantics, a condition tupel {Pre-Condition, Invariant, Post-Condition} has to be defined for each action of the compound. The pre-condition adopts the semantics of the action defined above and in a concrete situation thus grounds how the action has to be executed.
Synchronization can be created very naturally on the base that all executed actions have to respect these semantics. This means that actions cannot be executed (stated by pre-condition) as long as actions which have to be performed earlier in the sequence, have been finished (stated by post-condition) [Note: experiments have shown that this constraint is too hard, so that additional rules state that some actions can be performed semi-consequently]. The positions the actions arrive at, serve as synchronization points that are expressed in the conditions tupel. They are progressed on base of the action compound hierarchy using the action semantics. The initial positions a tactic is grounded on, suffice to calculate all further positions so all agents share this knowledge.

The actual execution is rather simple. Whenever a tactic should be performed (checked by its pre-condition), each participating agent (grounded by the tactic's pre-condition) retrieves a linear plan containing only what it has to do. It then executes this plan step-by-step while checking whether the other agents are still committed to the mutual tactic (by invariants).

5 Experiments

In order to evaluate the concepts developed in this work, we have evaluated our methodology for sketch recognition and interpretation as well as the transformation process from natural language descriptions to a populated ontology and its deployment in the 2D RoboCup Simulation League.

We have conducted experiments to test various aspects of the first in previous research which yielded promising results (details can be found in [9]). Its focus comprised solely the transformation of sketches to a team specific plan structure. The results demonstrate the status of our research in this direction, but are also interesting as a sketch yields far more information than just the pre-condition of a tactic: It provides an additional perspective on the complete tactic. In the future, we are thus interested in fusing sketch and text knowledge to further improve the performance of the ontology.

To demonstrate the performance of the transformation, population and deployment of ontologies in RoboCup, 16 tactics descriptions (containing 129 actions) have been chosen randomly and populated automatically into the prepared domain ontology. The ontology tactics were then compared to the original de-

scriptions. The results are that (1) 71% of all actions were identified perfectly, (2) an additional 20% of all actions were identified well enough (that is with few flaws which could be corrected by simple rules at execution) and (3) the temporal sequence was adjusted correctly in all cases. Most errors left can be explained by tagging flaws and concepts too complex to be expressed as SPO.
Post-processing and execution of the populated ontology were evaluated in simulation (without the influence of quantification). For this purpose, semantics as described in the previous setting were created and remaining transformation errors were corrected manually. The agents were positioned and provided with the populated ontology. After simulation, the logs were evaluated whether the geometric relations were as defined by the original tactic description. The results were satisfying as well, yielding that 86% of the tactics were executed correctly.

6 Outlook

In the future, we will deploy the tactics in tournaments and complete the ontology lifecycle in RoboCup. Verification and improvement of the populated ontology will thus be based on simulations: The defined semantics will be used as model to monitor the correct execution of tactics in the environment. If an interruption is detected, the reason is inferred from this model and the tactic adjusted.

References

1. Fascetti, E., Scaia, R.: Soccer Attacking Schemes and Training Execrcises. Reedswain Publishing (1998)
2. Gruber, T.: Ontology. Online (http://tomgruber.org/writing/ontology-definition-2007.htm) (2009) last visited 25th of February 2011.
3. Poon, H., Domingos, P.: Unsupervised ontology induction from text. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. (2010)
4. Maedche, A., Staab, S.: Learning ontologies for the semantic web. In: Semantic Web 2001. (2001)
5. Petasis, G., Karkaletsis, V., Paliouras, G.: Ontology population and enrichment: State of the art. Technical report (2007)
6. Maynard, D., Li, Y., Peters, W.: Nlp techniques for term extraction and ontology population. In: Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge. (2008)
7. Chen, D.L., Mooney, R.J.: Learning to sportscast: A test of grounded language acquisition. In: Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finnland (July 2008)
8. Mathieu, P., Routier, J.C., Secq, Y.: Towards a pragmatic use of ontologies in multi-agent platforms. In: Knowledge-Based Intelligent Information and Engineering Systems. Volume 2773 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2003) 1395–1402
9. Gspandl, S., Reip, M., Steinbauer, G., Wotawa, F.: From sketch to plan. In: 24th International Workshop on Qualitative Reasoning. (2010)