Concept Learning in Engineering based on Refinement Operator*

Yingbing Hua[0000−0002−6857−5586] and Björn Hein[0000−0001−9569−5201]

Faculty of Informatics, Karlsruhe Institute of Technology
76131 Karlsruhe, Germany
{yingbing.hua|bjoern.hein}@kit.edu

Abstract. Semantic interoperability has been acknowledged as a challenge for industrial engineering due to the heterogeneity of data models in the involved software tools. In this paper, we show how to learn declarative class definitions of engineering objects in the XML-based data format AutomationML (AML). Specifically, we transform AML document to the description logic OWL 2 DL and use the DL-Learner framework to learn the concepts of named classes. Moreover, we extend the $\mathcal{ALC}$ refinement operator in DL-Learner by exploiting the syntax specification of AML and show significant better learning performance.

Keywords: Concept Learning · Description Logics · AutomationML

1 Introduction

Engineering is referred to as the activities for designing, testing and commissioning complex industrial plants [3]. The engineering life cycle requires efficient data exchange between software tools, where semantic interoperability plays a central role. A lot of standardization groups are working at the semantic unification in various engineering subfields, but a "Super Data Model" that meets the needs of all tasks would not appear in the near future [2]. To enable data exchange between engineering tools, the international standard AML[1] (IEC 62714) proposes an XML-based approach [5]:

- Firstly, AML employs the XML schema from CAEX (IEC 62424) to define the syntax of engineering data, including classes, attributes, data objects and their relationships. We call this schema as the *abstract conceptual model* of AML.
- Secondly, fundamental engineering concepts are standardized as AML role and interface classes according to the schema. Role classes define the semantics of engineering objects e.g. Robot. Interface classes define the semantics

* The authors acknowledge support by the European Unions Horizon 2020 research and innovation programm under grant agreement No 688117 Safe human-robot interaction in logistic applications for highly flexible warehouses (SafeLog).

[1] https://www.automationml.org

of interfaces e.g. SignalInterface. They are modeled in subsumption hierarchies and can be extended to cover tool-specific terminologies. We call the role and interface classes as the *concrete conceptual model* of AML.
– Finally, AML specifies how to use the abstract and the concrete conceptual model to exchange data between engineering software tools.

The motivation of AML is to neutralize tool-specific data using the aforementioned conceptual models. Nevertheless, the standardized role and interface classes can not satisfy individual modeling requirements, since they lack of sufficient semantic expressiveness for describing tool-specific engineering objects. In practice, the user has to extend these classes and leave the semantic interpretation as a functionality of the data importer. This problem encourages the study of concept learning from engineering data in AML [6], i.e. inducing descriptions of named engineering classes in terms of AML class-, attribute- and interface names.

Semantic web technologies such as RDF(S) and OWL are popular tools to achieve interoperability in the World Wide Web. Several studies from the engineering domain have shown that transforming XML-based data to RDF(S)/OWL ontologies enhances the semantic interoperability with automated reasoning [1][8]. In this paper, we transform AML to the description logic OWL 2 DL and derive descriptions of named engineering classes using the well-known framework DL-Learner [4]. DL-Learner is an open-source project for concept learning in description logics. Among other important features, a downward refinement operator ρ for $\mathcal{ALC}$ was proposed, which was extended to an operator for $\mathcal{ALCQ}$ with the support of concrete roles [10]. Based on ρ, two algorithms OCEL [10] and CELOE [9] are implemented to learn concepts in description logics. It is worth noting that DL-Learner employs a partial closed-world reasoner for two reasons: a) it is much faster than a standard OWL reasoner, and b) machine learning tasks often desire closed world reasoning. In this paper, we study the performance of DL-Learner and argue that for our particular task, the refinement operator ρ is not quite efficient since it disregards the constraints exposed in the underlying XML schema of AML. Therefore, we propose an extension to ρ in its $\mathcal{ALC}$ part in section 3 and compare the results with the original one in section 4.

2 Preliminaries

Figure 1 shows the main components of AML. Modeling in AML usually begins with the hierarchy of user-specific roles and interfaces. Subsumption relations can be defined among these classes using the XML attribute `RefBaseClassPath`. The next step is the modeling of system units which represent reusable engineering objects, for example the robot model kr5 from the manufacturer KUKA. Each system unit may consist of interfaces (called as external interfaces) and nested internal structures (called as internal elements). The composition is illustrated by the connections with a filled diamond endpoint in figure 1. The star symbol over

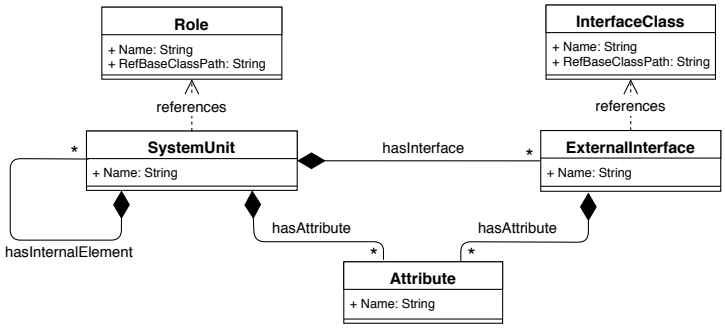

Fig. 1. The XML scheme of AutomationML. Some details are omitted for brevity.

one connection means that the cardinality of the composition is unlimited. Optionally, the semantics of a system unit or an external interface can be specified as a reference to an AML role or interface class respectively. While an internal element and an external interface can reference only one single class, a system unit can reference more than one to allow the modeling of multi-functional devices. Finally, attributes can be added to describe properties of objects, e.g. kr5.weight.

In order to learn concepts from AML, we firstly need a formal semantic representation of the XML-based data. The Web Ontology Language (OWL) is the W3C standard for knowledge representation on the web and its subset OWL 2 DL is semantically compatible with the $\mathcal{SROIQ}$ description logic. Besides of its decidability, OWL 2 DL provides a rich vocabulary for complex class expressions, making it a rational choice for the semantic lifting. In particular, the support of nominals and concrete roles allows us to describe meaningful engineering concepts. For example, *robots which have at least 6 axis from the manufacturer KUKA* can be stated as:

$$\mathsf{Robot} \sqcap \exists \mathsf{hasManufacturer}.[\text{"}\mathsf{KUKA}\text{"}] \sqcap \exists \mathsf{hasNumAxis}.[\geq 6] \tag{1}$$

Table 1 shows the mapping strategy from AML to OWL 2 DL. Specifically, AML role and interface classes are mapped to OWL classes, while system units and external interfaces are mapped to OWL individuals. Intuitively, relations between objects are mapped to object properties, and attributes are mapped to data properties. In this paper, we restrict the scope of the lifted AML ontology and focus on just two object properties: hasInternalElement (hasIE in short) and hasExternalInterface (hasEI in short). Because internal elements and external interfaces can be independent of any AML class, both hasIE and hasEI have the range as `OWL:Thing`. The abstract conceptual model of AML is not transformed since it does not carry useful semantics for engineering. However, we add an annotation to each lifted AML entity to indicate its role in the XML schema. For example, each lifted system unit has an annotation of SystemUnit. After the transformation, we obtain a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ including a terminological part $\mathcal{T}$ (T-Box) and an assertional part $\mathcal{A}$ (A-Box). The T-Box contains the concept definitions of AML role and interface classes, and the A-Box contains

AML	OWL 2 DL	DL	Example
role class	class	concept	Robot
interface class	class	concept	SignalInterface
system unit	individual	object	kr5
external interface	individual	object	kr5_digitalIn1
relationship	object property	abstract role	hasIE, hasEI
attribute	data property	concrete role	hasWeight

Table 1. Overview of the conceptual mapping from AML to OWL 2 DL.

the ground facts, i.e. system units and their internal structures. For example, following assertions are generated for the system unit kr5:

$$\begin{aligned} &\mathsf{hasManufacturer}(\mathsf{kr5}, \text{"KUKA"}) \\ &\mathsf{hasExternalInterface}(\mathsf{kr5}, \mathsf{kr5\_digitalIn1}) \\ &\mathsf{hasInternalElement}(\mathsf{kr5}, \mathsf{kr5\_arm}) \\ &\mathsf{DigitalIOInterface}(\mathsf{kr5\_digitalIn1}) \\ &\mathsf{Robot}(\mathsf{kr5\_arm}) \end{aligned} \tag{2}$$

3 Concept learning in AutomationML

Concept Learning is a subfield of machine learning which aims to induce a concept description for a set of positive and negative examples. In this paper, we focus on concept learning in description logics, and follow the definition of a learning problem as proposed in [10].

Definition 1 (concept learning in description logics). *Let* Target *be a concept name and* $\mathcal{K}$ *be a knowledge base (not containing* Target*). Let* $E = E^+ \cup E^-$ *be a set of examples, where* E^+ *are the positives examples and* E^- *are the negative examples. The learning problem is to find a concept* $C \equiv \mathsf{Target}$ *with* $\mathcal{K} \cup C \models E^+$ *and* $\mathcal{K} \cup C \not\models E^-$*.*

In the context of AML, the ultimate goal is to learn the description of a named class from a set of AML system units, since they are models of engineering objects that we are interested in. Figure 2 illustrates the learning procedure. While the AML data is mapped to an OWL 2 DL ontology as described before, the user is expected to select some AML system units as positive examples E^+, while leaving the rest as the negatives E^-. Afterwards, a configuration file has to be generated for the learning system.

Concept Learning methods often employ refinement operators to reduce the search space of concept hypotheses. The essential property of the refinement operator ρ in DL-Learner is its *completeness* in $\mathcal{ALC}$. That means, starting from any concept C (including $\top$), ρ will reach the target concept with sufficient time. Although the support of concrete roles makes ρ incomplete because of

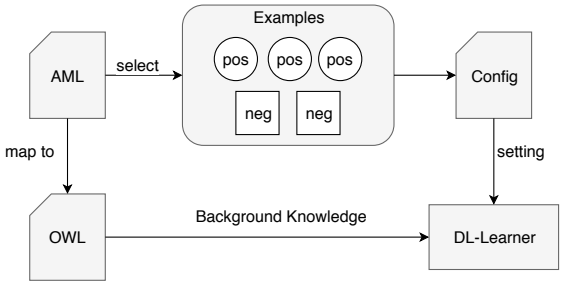

Fig. 2. Procedure of concept learning in AutomationML.

the infiniteness of real numbers, ρ is capable of finding proper concrete roles by computing splits in the space of real numbers. However, the direct use of ρ would be inefficient, because it does not take into account the syntactic constraints defined in the underlying XML schema of AML. Specifically, figure 1 shows that only external interfaces can reference interface classes and each external interface can only reference one interface class. These constraints can be integrated into the refinement operator to improve the performance of learning, especially for a large T-Box. Therefore, we divide the set of named concepts N_C into two subsets N_{ar} and N_{ai}, where N_{ar} is the set of all AML role classes and N_{ai} is the set of all AML interface classes. Then we define the sets M_{op}, M_{ie} and M_{ei} as follows:

$$M_{op} = \{\exists \mathsf{hasIE}.\top, \exists \mathsf{hasEI}.\top, \forall \mathsf{hasIE}.\top, \forall \mathsf{hasEI}.\top\} \tag{3}$$

$$M_{ie} = \{A \,|\, A \in N_{ar}, \nexists A' \in N_{ar} : A \sqsubset A'\} \tag{4}$$

$$M_{ei} = \{A \,|\, A \in N_{ai}, \nexists A' \in N_{ai} : A \sqsubset A'\} \tag{5}$$

M_{ie} is the set of top level AML role classes, and M_{ei} is the set of top level AML interface classes. Further, let $U_{ie} = \{C_1 \sqcup C_2 \sqcup ... \,|\, C_i \in M_{ie} \cup M_{op}\}$, $U_{ei} = \{C_1 \sqcup C_2 \sqcup ... \,|\, C_i \in M_{ei}\}$ and $sh_{\downarrow}(C)$ be the set of direct sub classes of a named concept $C \in N_C$, we extend ρ in the following cases:

- $\rho(C) = U_{ei}$, if $C = \top$ and C is a filler of hasEI
- $\rho(C) = U_{ie}$, if $C = \top$ and C is not a filler of hasEI
- $\rho(C) = sh_{\downarrow}(C)$, if $C \in N_C$ is a filler of hasEI
- $\rho(C) = sh_{\downarrow}(C) \cup \{C \sqcap D | D \in \rho(\top)\}$, if $C \in N_C$ is not a filler of hasEI

In the other cases, we keep ρ as it was in DL-Learner and omit the details of it in this paper for brevity. We call the new refinement operator ρ^{aml} and implemented it based on ρ in the DL-Learner framework. Notice that negated atomic concepts such as $\neg A$ are ignored in both M_{ie} and M_{ei}, since negations are not preferred in engineering and are not used in practice. Moreover, we do not refine the filler of hasEI with concept intersections, because one external interface can reference only one interface class and does not have nested structures. As a result, ρ^{aml} would generate far less refinements than ρ. For example, a refinement chain $\top \rightsquigarrow \exists \mathsf{hasEI}.\top \rightsquigarrow \exists \mathsf{hasEI}.\mathsf{C_1} \rightsquigarrow \exists \mathsf{hasEI}.(\mathsf{C_1} \sqcap \mathsf{C_2})$ could be produced by ρ but not by ρ^{aml}, since $(\mathsf{C_1} \sqcap \mathsf{C_2})$ is not a proper refinement of the filler of hasEI.

	Benchmark 1			Benchmark 2		
	Concepts	Overall	Reasoning	Concepts	Overall	Reasoning
T1	3996(65%)	508(75%)	304(73%)	4733(54%)	451(64%)	308(68%)
T3	4926(13%)	595(25%)	404(28%)	5746(10%)	800(15%)	541(18%)
T17	67614(5%)	5483(5%)	3785(6%)	113914(6%)	9243(5%)	6361(6%)
T12	67918(4%)	5465(4%)	3741(5%)	114427(7%)	8792(8%)	6170(9%)
T14	562705(?%)	84419(?%)	40997(?%)	812225(?%)	105779(?%)	53679(?%)

Table 2. Evaluation of CELOE using both refinement operators.

4 Results

In this section, we compare learning results using the refinement operator ρ from DL-Learner and the extended one ρ^{aml} as described above. Specifically, we measure the time required until the first 100% accurate solution is found. The raw AML document comes from the research project ReApp which was originally used for modeling industrial robot systems [7]. The lifted AML ontology comprises of 222 classes, 497 individuals and 73 data properties. To simulate the heterogeneity in engineering projects, we perform two different benchmarks. The first one has an additional 50 AML role and 25 AML interface classes, while the second one has twice as much. We choose the algorithm CELOE, since its heuristic is configured to produce shorter concepts than OCEL. Generally speaking, CELOE sets a stronger penalty to long refinements and is less rewarded by the immediate accuracy gain. In each benchmark, we run 25 experiments of different target concepts. Axiom 6 show one example of the ground truth.

$$\mathsf{RobotWithServoMotors} \equiv \exists \mathsf{hasIE}.(\mathsf{ArticulatedRobot} \sqcap \exists \mathsf{hasIE}.\mathsf{ServoMotor}) \quad (6)$$

Table 2 summarizes the results of five representative experiments of varying complexity, while the rest ones share similar characteristics and are omitted for brevity. The column *Concepts* is the number of tested concept hypotheses. The column *Overall* is the overall time needed to find the first correct solution, and the column *Reasoning* is the time spent for reasoning, both timers are in milliseconds. All measurements in the table are taken from ρ^{aml}, while the percentages in parentheses represent the ratio in the form of ρ^{aml}/ρ. Unknown values of the ratio means that no solution was found within 10 minutes with ρ. The results show that ρ^{aml} is significant faster than ρ (the ratios are much smaller than 100%), since ρ^{aml} generates much less concept hypotheses for testing. However, for some cases no solution was found within 10 minutes with either ρ or ρ^{aml}, so learning in AML still remains a challenging task[2].

5 Conclusion

In this paper, we studied concept learning from engineering data stored in the XML-based data format AML. We showed how to use the DL-Learner framework

[2] By reducing the expansion penalty of CELOE, we were able to find a solution with the extended refinement operator ρ^{aml} for some of these hard cases.

to learn engineering concepts in description logic, by transforming an AML document to an OWL 2 DL ontology. We proposed an extension of the $\mathcal{ALC}$ downward refinement operator ρ in DL-Learner to exploit the syntactic constraints defined in the XML schema of AML. Experimental results show that the extended operator ρ^{aml} has a significant performance improvement in all test cases. However, learning is still very challenging in some cases and could be worse if the size of the T-Box grows further. In future work, we want to investigate whether we can achieve better learning results using a semantic language other than OWL 2 DL, for example the hybrid system $\mathcal{AL}$-log that merges $\mathcal{ALC}$ and DATALOG. In particular, an ideal refinement operator for $\mathcal{AL}$-log was proposed in [11]. For learning in OWL 2 DL, we want to study if a heuristic can intelligently adapt itself to avoid the laborious fine tuning of learning parameters.

References

1. Abele, L., Legat, C., Grimm, S., Müller, A.W.: Ontology-based validation of plant models. In: 2013 11th IEEE International Conference on Industrial Informatics (INDIN). pp. 236–241 (July 2013)
2. Bigvand, P.G., Drath, R., Scholz, A., Schüller, A.: Agile standardization by means of PCE requests. In: 2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA). pp. 1–8 (Sept 2015)
3. Bigvand, P.G., Fay, A., Drath, R., Carrion, P.R.: Concept and development of a semantic based data hub between process design and automation system engineering tools. In: 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA). pp. 1–8 (Sept 2016)
4. Bühmann, L., Lehmann, J., Westphal, P.: DL-Learner: A framework for inductive learning on the semantic web. Web Semant. **39**(C), 15–24 (Aug 2016)
5. Drath, R., Lüder, A., Peschke, J., Hundt, L.: AutomationML - the glue for seamless automation engineering. In: 2008 IEEE International Conference on Emerging Technologies and Factory Automation. pp. 616–623 (Sept 2008)
6. Hua, Y., Hein, B.: Concept learning in AutomationML with formal semantics and inductive logic programming (accepted). In: 14th IEEE International Conference on Automation Science and Engineering (Aug 2018)
7. Hua, Y., Zander, S., Bordignon, M., Hein, B.: From AutomationML to ROS: A model-driven approach for software engineering of industrial robotics using ontological reasoning. In: 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA). pp. 1–8 (Sept 2016)
8. Kovalenko, O., Wimmer, M., Sabou, M., Lüder, A., Ekaputra, F., Biffl, S.: Modeling AutomationML: semantic web technologies vs. model-driven engineering. In: Emerging Technologies Factory Automation (ETFA), 2015 IEEE 20th Conference on. pp. 1–4 (Sept 2015)
9. Lehmann, J., Auer, S., Bühmann, L., Tramp, S.: Class expression learning for ontology engineering. Web Semantics: Science, Services and Agents on the World Wide Web **9**(1), 71 – 81 (2011)
10. Lehmann, J., Hitzler, P.: Concept learning in description logics using refinement operators. Machine Learning **78**(1), 203 (Sep 2009)
11. Lisi, F.A., Malerba, D.: Ideal refinement of descriptions in $\mathcal{AL}$-log. In: Horváth, T., Yamamoto, A. (eds.) Inductive Logic Programming. pp. 215–232. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)