Proceedings of the Twenty-Fifth International Florida Artificial Intelligence Research Society Conference

Learning Artifact Capabilities Via a Hybrid Ontology

Felicitas Mokom and **Ziad Kobti**
School of Computer Science
University of Windsor
401 Sunset Avenue, Windsor, Ontario, N9B-3P4
mokom@uwindsor.ca and kobti@uwindsor.ca

Abstract

Artifact capabilities can play an important role in understanding human cognition. Over time humans learn to use artifacts, evolve the knowledge and combine acquired capabilities with others to form complex capabilities. In this study we present a hybrid ontology for artifacts to facilitate learning artifact capabilities. We develop a framework where agents simultaneously exploit a centralized artifact ontology in the environment and a distributed artifact ontology local to each agent. We demonstrate how both ontologies can be used by agents both in the artifact selection process and in learning artifact use. The local ontology serves as domain knowledge gained by the agent as it learns. We illustrate an example to show how an acquired artifact capability can be stored in an agent's local ontology for future use.

Introduction

It is a common contention among researchers in the cognitive sciences that tool or artifact use has played a significant role in the evolution of human intelligence. These capabilities have aided humans in dealing with changes in the environment ultimately leading to a modification of the environment to accommodate human needs. According to cognitive scientist David Kirsh (2010), the basic process of using a tool is driven by its physical constitution. Baker (2004) provides a defense for the ontological status of artifacts as objects with practical functions that are constituted of parts.

An artifact or tool in this study refers to any physical object in the environment that provides some functionality useful to a human agent towards the satisfaction of a goal (Mokom and Kobti 2011b). These agents are artifact capability-learning agents defined as rational agents (BDI-theory) that employ learning techniques towards discovering how to use artifacts. Singh (1999) argued that residing within the agent's belief are two complementary aspects: "know-that" specifies what the agent knows about the world and "know-how" describes the agent's procedural knowledge for executing plans.

In this study we integrate the use of domain knowledge into the artifact capability learning model presented in Mokom and Kobti (2011b) via a hybrid ontology for artifacts. The model supported agents utilizing exploratory evolutionary methods towards learning artifact capabilities. A centralized ontology for artifacts as objects with immutable properties and constraints is used to create a knowledge base (KB) of artifacts in the environment. Complementary to this, a distributed ontology specific to each agent maintains a dynamic local KB representing each agent's domain knowledge of artifacts. While the centralized KB allows for the filtering of an agent's choice of actions, an evolving local KB facilitates an agent's use of prior discoveries in the learning process.

The next section provides some background on related work. It is followed by a description of our hybrid ontology incorporated within an artifact capability-learning agent. We demonstrate use of the ontologies with an illustrated example, present conclusions deduced and future work.

Related Work

Some existing research efforts in creating models for human cognitive capabilities with respect to artifacts are worth noting. In Stoytchev's (2005) model, robots learned tool use by randomly attempting different actions with the tool, observing and recording the results. The model was however more concerned with robotic sensors and body schema than the reasoning process involved. Omicini, Ricci, and Viroli's (2006) theory of artifact selection and use described an environment in which agents reason about artifacts for the achievement of their goals. An ontology based on their work allowed agents to exploit artifacts in the environment (Acay, Pasquier, and Sonenberg 2007). In the model, artifacts exposed a *usage interface* defined as an artifact's permissible operations, *operating instructions* described as procedures for its use and a *function description* that specified what the artifact is used for. However, the model's requirement that all these aspects exist in the form of *tool manuals* in the environment do not facilitate exploratory agents learning artifact capabilities. Furthermore, the incremental process of learning is not accommodated.

A Hybrid Ontology for Artifacts

Incorporating the use of prior knowledge into learning models is necessary for the development of intelligent agents and learning often builds on background knowledge involving an incremental process (Russell and Norvig 1995).

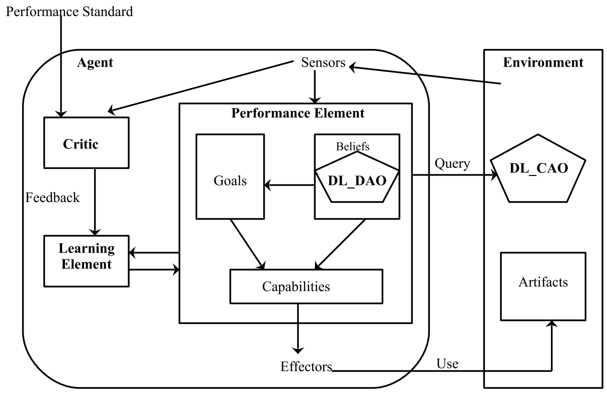

Figure 1: An artifact capability-learning agent

Table 1: Steps of artifact capability-learning agent

```
(1) PE selects a goal
(2) PE selects an artifact
Loop
    (3) PE deliberates / performs action(s) with artifact
    (4) PE performs updates based on percepts
    (5) CE evaluates percepts / renders feedback to LE
    (6) LE advises and/or suggests improvements to PE
    (7)PE applies changes
Until PE is advised goal is achieved
```

The representation of an artifact capability-learning agent (Mokom and Kobti 2011b) expanded to include the exploitation of a hybrid ontology is presented in Figure 1. The general steps of an artifact capability-learning agent are shown in Table 1.

The agent's cognition is composed of three main elements. The *performance* element (PE) deliberates and chooses the agent's actions using its beliefs, goals and capabilities. Once an action is performed, the *critic* element (CE) evaluates resulting percepts against an external predefined standard of performance and provides feedback to the *learning* element (LE) which is responsible for generating suggestions towards improving the performance element.

Using Singh's notions of "know-that" and "know-how" (Singh 1999), artifact domain knowledge discovered by the agent is defined using a distributed local ontology (DL_DAO) within the agent's beliefs. A centralized ontology (DL_CAO) defining immutable properties of artifacts accessible by every agent exists within the environment.

Representing Artifact Knowledge

Description Logic (DL) is chosen to represent the artifact ontologies because it blends simple logical operations with clear semantics (Russell and Norvig 1995) making it useful for practical applications. Additionally, the inference in DL has been proven to be tractable by *sound* and *complete* algorithms based on the logic (Baader and Nutt 2003). A DL-based KB is composed of a Terminology Box (TBox) that provides axioms for concepts and roles in the domain and an Assertion Box (ABox) that specifies axioms for concrete concepts and roles. DL-based description languages use model-theoretic semantics. An interpretation I for any such language is composed of a non-empty set ∇^I known as the domain of interpretation and an interpretation function. The function assigns a set $A^I \subseteq \nabla^I$ to every atomic concept A and a set of binary relations $R^I \subseteq \nabla^I \times \nabla^I$ to every atomic role R. Our ontologies are defined using DL-based basic *AL-language (attributive language)* (Baader and Nutt 2003).

DL_CAO: A Centralized Ontology for Artifacts

Every artifact can be defined in terms of its immutable properties distinguishable from what an agent may know about them. For example an agent may not be aware that a ball can bounce but that has no bearing on the fact that the ball has a *bouncable* property. The concept and role terminologies for DL_CAO are presented in Tables 2 and 3 respectively.

DL_CAO provides definition for five concepts.

Concepts(1-4): An *Artifact* is defined as an object with at least one part. An *ArtifactPart* can be another artifact or a part with at least one physical and one functional attribute. The *PAttribute* concept restricts a physical attribute to a single value. The *FAttribute* concept allows a functional attribute one or more possible values as well as constraints specifying at least one value for which the constraint applies.

CompAttr: Defines the notion of a set of at least 2 functional attributes that are compatible with each other, that is, they can be applied together.

DL_DAO: A Distributed Ontology for Artifacts

Agents maintain knowledge discovered while learning an artifact capability in a local dynamic KB stored in the agent's beliefs. Concept and role terminologies for DL_DAO are presented in Tables 4 and 5 respectively.

The capability is defined as as a sequence of tasks, performable on an artifact for its use in a particular context (Mokom and Kobti 2011b). An agent attempts a task by selecting a value for one or more functional attributes. The agent's objective is to learn the values/combinations that get it closer to fulfill the task and ultimately its goal.

DL_DAO provides definitions for seven concepts.

Concepts(1-4): An *Artifact* is defined as an object whose associated parts are artifact parts. An *ArtifactPart* can be another artifact, otherwise, any associated attributes must be either a physical or functional attribute. A physical attribute can have only one value whereas a functional attribute can have constraints with associated constraint values.

ArtifactFunction: Abstracts the major function of the artifact. It includes the artifact and the context in which it is used. It also specifies artifact parts discovered as unnecessary for the context and those discovered as required with their associated physical attributes.

Procedure: Describes the operational instructions for an artifact use. It has an artifact, the context in which it is used

Table 2: DL_CAO concept terminology

(1)	$Artifact$	$\equiv$	$object \sqcap \exists hasPart.ArtifactPart$
(2)	$ArtifactPart$	$\equiv$	$Artifact \sqcup (\exists hasAttribute.PAttribute \sqcap \exists hasAttribute.FAttribute)$
(3)	$PAttribute$	$\equiv$	$\leq 1\, hasValue$
(4)	$FAttribute$	$\equiv$	$\exists hasValue.\top \sqcap \forall hasConstraint.\,(\exists hasValue.\top)$
(5)	$CompAttr$	$\equiv$	$\geq 2\, hasAttribute \sqcap \forall hasAttribute.FAttribute$

Table 3: DL_CAO role terminology

hasPart	*hasValue*
hasAttribute	*hasConstraint*

and at least one task defined with its position in the sequence of tasks to support the fact that multiple tasks are ordered.

Task: Has at least one functional attribute value represented as an artifact part, a functional attribute and a chosen value.

Exploitation of DL_CAO and DL_DAO

According to Figure 1, DL_DAO resides in the agent's belief as part of PE and DL_CAO exists in the environment. Table 1 indicates that the ontologies are possibly involved in Steps (1, 2, 3, 4, 7). In this study we do not consider the use of the ontologies for goal selection (Step 1) and assume the agent has selected a goal g. We address the use of the ontologies in the remaining steps which involves querying their respective ABox's. For the rest of the paper we refer to DL_CAO's ABox as $\mathcal{A}_\text{c}$ and DL_DAO's ABox as $\mathcal{A}_\text{d}$.

(2) Artifact Selection: The agent performs a query on $\mathcal{A}_\text{d}$ formulated as follows: $\mathcal{A}_\text{d} \models ArtifactFunction\,(f) \wedge \exists useContext\,(f, g) \wedge \exists hasArtifact\,(f, a)$ to find every artifact a that the agent knows to use for goal g. If the returned set is non-empty, the agent can use the details of f to ensure that any chosen artifact from the environment has the necessary parts and attributes. The artifact can be chosen by simply querying $\mathcal{A}_\text{c}$ for a matching artifact: $\mathcal{A}_\text{c} \models Artifact\,(a)$. A more complex query might check for the necessary parts and attributes and use those to identify a possible artifact. If the returned set from the query on $\mathcal{A}_\text{d}$ is empty, then the agent must make a random choice among the available artifacts.

(3) PE's Deliberation: In order to choose what action(s) to perform the agent queries $\mathcal{A}_\text{c}$ for a set of functional attributes defined with the CompAttr concept, or any functional attribute that can be attempted on its own. If the artifact has been used before, the agent can consider constraint knowledge in $\mathcal{A}_\text{d}$ when choosing values and ignore parts that are not required.

(4) PE's Updates Based On Percepts: PE might update $\mathcal{A}_\text{d}$ based on the perception of its actions such as discovered constraints. Since these constraints do not necessarily mean that the artifact cannot still be used for the goal, $\mathcal{A}_\text{d}$ is updated with the new assertions and PE waits for CE to determine the real effect of the constraint.

(7) PE's Changes: PE makes changes to $\mathcal{A}_\text{d}$ based on suggestions provided by LE generated using CE's feedback. If the information indicates a new task has been learned then a new task assertion is added to $\mathcal{A}_\text{d}$ as part of an existing procedure or a new one is formulated. Other derivable assertions include artifact parts that PE gets notified are not necessary for the artifact function.

Once an agent successfully learns an artifact capability $\mathcal{A}_\text{d}$ will contain domain knowledge obtained by the agent that can be applied to future use of similar artifacts including the context in which the artifact was used, abstracted to facilitate artifact selection in the future.

A Simple Illustration

In this section we provide a simple illustration of the acquisition of domain knowledge by an agent. We consider a simple scenario. An agent encounters a pen for first time and sets a goal to write with it. We assume the agent has no information in its local domain knowledge $\mathcal{A}_\text{d}$ at the start of the process. $\mathcal{A}_\text{c}$ contains assertions about the pen the agent finds. Tables 6 and 7 represent $\mathcal{A}_\text{c}$ throughout the learning process and a possible $\mathcal{A}_\text{d}$ at the end of the learning process respectively. Assertions in $\mathcal{A}_\text{d}$ are obtained as explained in the previous section. Integer ranges are assumed for the functional attributes representing position held for *hold* and number of cm moved for *move*. $\mathcal{A}_\text{d}$ shows that at the end of the process, the agent has learned how to use a pen to write with a single task in its procedure that involves holding the pen at position 10 and moving it 1cm. If the agent had held the pen at position 5, it would have been able to assert a constraint as well.

Conclusions and Future Work

This study presents a hybrid ontology for artifacts usable by agents learning artifact capabilities. The objective was to facilitate the learning process for agents learning how to use artifacts for their goals. A framework was developed for agents to exploit a centralized artifact ontology in the environment alongside a distributed ontology local to each agent. A learning agent incrementally used information obtained from the centralized ontology combined with domain knowledge in its local ontology to make informed decisions both in the artifact selection and use processes. An example was illustrated to demonstrate how both ontologies are used.

This study is domain neutral and was conducted in order to establish a general framework for learning artifact use by exploiting domain knowledge. In order to test an implementation of the framework, a more specific domain is needed in a defined case study. It would be useful to integrate the

Table 4: DL_DAO concept terminology

(1)	$Artifact$	$\equiv$	$object \sqcap \forall hasPart.ArtifactPart$
(2)	$ArtifactPart$	$\equiv$	$Artifact \sqcup \forall hasAttribute.\,(PAttribute \sqcup FAttribute)$
(3)	$PAttribute$	$\equiv$	$\leq 1\, hasValue$
(4)	$FAttribute$	$\equiv$	$\forall hasConstraint.\,(\exists constraintValue.\top)$
(5)	$ArtifactFunction$	$\equiv$	$\exists hasArtifact.Artifact \sqcap \exists useContext.\top \sqcap \forall nonreqPart.ArtifactPart \sqcap$ $\forall hasPartAtt.\,(\exists hasPart.ArtifactPart \sqcap \exists hasAttribute.PAttribute)$
(6)	$Procedure$	$\equiv$	$\exists hasArtifact.Artifact \sqcap \exists useContext.\top \sqcap \exists hasTask.\,(Task \sqcap \exists hasTPos.\top)$
(7)	$Task$	$\equiv$	$\exists chosenValue.(\exists hasPart.ArtifactPart \sqcap \exists hasAttribute.FAttribute \sqcap$ $\exists hasValue.\top)$

Table 5: DL_DAO role terminology

hasPart	*hasConstraint*	*nonReqPart*
hasAttribute	*hasArtifact*	*hasPartAttr*
hasValue	*useContext*	*hasTask*
hasTPos	*chosenValue*	

Table 6: DL_CAO's ABox $\mathcal{A}_c$

Artifact(PEN)	*hasValue(INK, RED)*
ArtifactPart(TUBE)	*hasValue(HOLD, 5)*
hasPart(PEN, TUBE)	*hasValue(HOLD, 10)*
PAttribute(INK)	*hasValue(MOVE, 1)*
FAttribute(HOLD)	*hasConstraint(HOLD, 5)*
FAttribute(MOVE)	*CompAttr(HOLD, MOVE)*

Table 7: DL_DAO's ABox $\mathcal{A}_d$

Artifact(PEN)
ArtifactPart(TUBE)
hasPart(PEN, TUBE)
PAttribute(INK)
FAttribute(HOLD)
FAttribute(MOVE)
ArtifactFunction(FWPEN)
hasArtifact(FWPEN, PEN)
useContext(FWPEN, WRITE)
hasPartAttr(FWPEN, (TUBE, INK))
Procedure(PWPEN)
hasArtifact(PWPEN, PEN)
useContext(PWPEN, WRITE)
Task(TWPEN)
hasTask(PWPEN, (TWPEN, 1))
chosenValue(TWPEN, (TUBE, HOLD, 10))
chosenValue(TWPEN, (TUBE, MOVE, 1))

ontologies into the work by Mokom and Kobti (2011a) and Mokom and Kobti (2011b) to evaluate their relevance in individual, social and cultural learning of artifact capabilities. In this study we do not consider the use of the ontologies for goal selection. Goal selection can be driven by domain knowledge of an artifact which have now been explicitly represented. Simulating goal evolution can provide valuable insight on how humans decide on objectives to pursue.

Acknowledgements

This work is made possible by Grant DEB-0816400 from the National Science Foundation and NSERC Discovery No. 327482.

References

Acay, D. L.; Pasquier, P.; and Sonenberg, L. 2007. Extrospection: Agents reasoning about the environment. In *Proceedings of the 3rd International Conference on Intelligent Environments*, 220–227.

Baader, F., and Nutt, W. 2003. Basic description logics. In Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds., *The description logic handbook*. New York, NY, USA: Cambridge University Press. 43–95.

Baker, L. R. 2004. The ontology of artifacts. *Philosophical Explorations* 7(2):99–112.

Kirsh, D. 2010. Explaining artifact evolution. In Malafouris, L., and Renfrew, C., eds., *The Cognitive Life of Things*. Cambridge, England: Cambridge University Press. 1–37.

Mokom, F., and Kobti, Z. 2011a. A cultural evolutionary model for artifact capabilities. In *Proceedings of the European Conference on Artificial Life*, 542–549. Paris, France: MIT Press.

Mokom, F., and Kobti, Z. 2011b. Evolution of artifact capabilities. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 476–483. New Orleans, Louisiana: IEEE Press.

Omicini, A.; Ricci, A.; and Viroli, M. 2006. Agens Faber: Toward a theory of artefacts for MAS. *Electronic Notes in Theoritical Computer Sciences* 150(3):21–36.

Russell, S., and Norvig, P. 1995. *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall.

Singh, M. P. 1999. Know-how. In Rao, A., and Wooldridge, M., eds., *Foundations of Rational Agency, Applied Logic Series*. Kluwer. 105–132.

Stoytchev, A. 2005. Behavior-grounded representation of tool affordances. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 3060–3065.