Fanan, A and McCluskey, T.L.

Towards Learning Operator Schema from Free Text

Towards Learning Operator Schema from Free Text

A.Fanan and T.L.McCluskey,
Department of Informatics,
School of Computing and Engineering,
University of Huddersfield

Abstract

In automated planning current research is focused on developing *domain-independent* planning engines. These require domain models, written in a standard input language such as PDDL to supply knowledge of the planning application and task, before they can be used. The main component of a domain model is the representation of actions in the form of lifted operator schema. The acquisition and engineering of these schema is an important area of research, as this process is recognised as being difficult and laborious even for planning experts.

A fruitful line of research is to investigate mechanisms to automatically learn planning domain models. Recent research has studied learning from structured or refined inputs supplied by a training agent (Cresswell, McCluskey, and West 2011; Zhuo et al. 2010; Wu, Yang, and Jiang 2005; McCluskey et al. 2010). An alternative method would be to allow planning agents to learn and develop the domain models by observation. One freely available source for learning actions is selected web text; here actions are represented as verbs in natural language. This project aims to investigate the possibility of extracting formal structures representing actions from free text. We intend to utilise large text corpuses available on-line from which to extract such action knowledge, and learn operator schema in a formal language that can be converted to PDDL.

A. Introduction

Over the last decade there has been some progress in producing tools to help in the acquisition of domain models for planning. These fall into three areas:

- *knowledge engineering tools for supporting experts:* this area covers user interface tools which help users in domain formulation, domain analysis, and planning simulation. GIPO (Simpson, Kitchin, and McCluskey 2007) provides a diagrammatic interface for the user in order to avoid the need to write operator schema explicitly. From user drawn state machines and annotations representing the domain, the system generates a formulation automatically, relieving the user of the burden of writing detailed parameters, logical expressions or predicates. GIPO is analogous in functionality to visualisation tools that perform code generation in software engineering environments. itSimple (Integrated Tools Software Interface for Modelling Planning Environment) (Vaquero et al. 2009) is a similar tool, but closer to the software engineering community than GIPO with its emphasis on the use of UML. Unlike GIPO, itSimple continues to be maintained and upgraded. The latest version of itSimple is 3.5 featuring an integrated environment with representation languages such as XML, Petri Nets and PDDL.
- *creating a planning domain model via translation:* There are situations where a formal model of a domain already exists, for instance in the areas of business modelling, workflow, or web services, and therefore the opportunity exists to create translation tools that when input with a model described in an application area specific language, output a full or partial planning domain model. The 2009 run of ICKEPS focused on this specific aspect of knowledge engineering technology, clearly showing ways that planning engines can be used as "black boxes" within environments that assemble the inputs to the planner automatically, and exploit the output plan in an application-dependent way (Bartak, Fratini, and McCluskey 2010).
- *tools that learn the domain model from engineered examples and partial domain models:* this includes tools that build models automatically from plan traces and background knowledge which may lead to reduction in timescales and effort over handcrafting. Several systems that learn and refine domain models from examples have been developed in recent years. For example, the systems ARMS (Action Relation Modelling System), LOCM (Learning Object-Centred Models) and LAMP (Learning Action Models from Plan traces) were all built specifically to support the learning of domain models from many examples of plan traces.

Our research is related to the last area, and we explore current systems in a little more detail. Wu et al (WU, YANG, and JIANG 2007) have presented the process of ARMS in the following phases: it converts all action instances to their schema forms and finds the frequent predicate-action and action sets from the converted plans that share the same object types. It then transforms the frequent sets into a weighted representation to be input to a SAT solver, and synthesises operator schemas from the results. Systems such as ARMS need to input other information as well as example plan traces before learning can be effective. This can include predicates, initial, goal or intermediate state descriptions, target action names, or other domain information.

At the extreme of these systems is LOCM (Cresswell, McCluskey, and West 2011), an inductive tool which automatically induces a domain model from a set of training examples of plans without the need for any background information. The training input to LOCM (sets of valid action sequences) does have to satisfy certain constraints, however: each action is specified as a name followed by a sequence of affected objects, each instance of a named action has parameters in a consistent order as well as some assumptions on the inductive learning process. Quite strict assumptions are made of the output model also: objects are assumed to belong to a "sort" which, via a state machine, defines the identical behaviour of each object in the sort.

There are other systems that utilise other types of learning, but they too still need well engineered input. For example, LAWS (H.H.Zhuo, Q.Yang, R.Pan and L.Li 2011) learns using analogy: it inputs an existing domain model and uses it to synthesize a new domain model. It still requires other information, such as target action and predicate templates. Hence, common to all of these tools is the necessity for a trainer or teacher to prepare the input. These inputs in some cases have to be refined or engineered themselves before the tools can learn effectively. Invariably there are also many assumptions made on the form of the output model.

Research Programme

In contrast to the work above, we hypothesise that it is possible to learn the main part of a domain model, that is the operator schema, by observation alone, without the need for a trainer or specifically engineered background knowledge. Recently, in the broader area of AI, techniques are being developed to systemize the learning of action knowledge from free text such as extracting script or narrative event schemas (Chambers and Jurafsky 2009).

Closest to our proposal is the work of Sil and Yates: they have implemented a system called PREPOST that works in identifying the preconditions and the effects of actions and events (Sil, Huang, and Yates 2010). PREPOST is a text mining system that involves two distinct learned classifiers for both preconditions and postconditions. The PREPOST technique is dependent on the ability of the search engine to find a collection of documents that contains a specific word that should be a verb or an event. Therefore, PREPOST uses an English language progressive form of verbs and is used almost exclusively with events and action verbs. PREPOST uses a search engine to find a huge collection of documents for a selected word (X) by considering the pattern "is/are/were/was + X+ing", then uses techniques from the field of inductive learning to identify preconditions and effects. The system is supported by other techniques such as semantic role labelling and has been used with some success to identify predicates and event conditions in text. Recently, Sil and Yates have improved upon their previous work by presenting a system for extracting a full-formed STRIPS representation of actions (Sil and Yates 2011). This system has demonstrated that it has the ability to identify the preconditions of previously unseen actions using web documents that are automatically downloaded. Under certain constraints, the precision of the system in recognising preconditions is high.

The research programme that we have embarked on aims to investigate the feasibility of utilising techniques in text mining and natural language processing to identify the characteristics of actions and events from free text, and extract pre- and post conditions within a formal language in a similar manner to PREPOST. We plan, however, to add to the former work constraints and ideas from domain model learning systems such as LAWS and ARMS, in order to leverage more knowledge to improve the system's precision. The use of analogical learning, for instance, which draws on a database of existing models, may be able to provide a more accurate action template for a new verb that is semantically close to one that has previously been encoded in the database.

The long term aim is to embed inside virtual agents the potential to acquire and maintain their own domain theories. This entails encouraging more work into the under-developed field of *learning planning domain models by observation.* Work to date has been aimed at rationally re-constructing the PREPOST system, and connecting such efforts with work on domain model learning and knowledge engineering carried out previously in the AI Planning community. A major research question is: *Is it feasible to utilise within an action learning algorithm a combination of:*

- *text mining tools,*
- *natural language semantic tools such as WordNet,*
- *assumptions and constraints about actions*

in order to induce accurate STRIPS models? In addition to investigating the case with free text, we are considering tackling this question in restricted domains where the vocabulary is limited or controlled, and the a set of actions is connected up, as might be the case in instruction manuals.

References

Bartak, R.; Fratini, S.; and McCluskey, L. 2010. The third competition on knowledge engineering for planning and scheduling. AI Magazine, Spring 2010.

Chambers, N., and Jurafsky, D. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, 602–610. Stroudsburg, PA, USA: Association for Computational Linguistics.

Cresswell, S.; McCluskey, T.; and West, M. M. 2011. Acquiring planning domain models using LOCM. *Knowledge Engineering Review (To Appear).*

H.H.Zhuo, Q.Yang, R.Pan and L.Li. 2011. Cross-Domain Action-Model Acquisition for Planning Via Web Search. In *Proceedings of ICAPS*.

McCluskey, T. L.; Cresswell, S. N.; Richardson, N. E.; and West, M. M. 2010. Action Knowledge Acquisition with Opmaker2. In *Agents and Artificial Intelligence*, volume 67 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg. 137–150.

Sil, A., and Yates, A. 2011. Extracting STRIPS representations of actions and events. In *Recent Advances in Natural Language Learning (RANLP).*

Sil, A.; Huang, F.; and Yates, E. 2010. Extracting action and event semantics from web text. AAAI Fall Symposium on Commonsense Knowledge.

Simpson, R. M.; Kitchin, D. E.; and McCluskey, T. L. 2007. Planning Domain Definition Using GIPO. *The Knowledge Engineering Review* 22(1).

Vaquero, T.; Silva, J.; Ferreira, M.; and Tonidandel, F. 2009. From requirements and analysis to pddl in itsimple3. 0. *Artificial Intelligence*.

Wu, K.; Yang, Q.; and Jiang, Y. 2005. Arms: Action-relation modelling system for learning acquisition models. In *Proceedings of the First International Competition on Knowledge Engineering for AI Planning*.

WU, K.; YANG, Q.; and JIANG, Y. 2007. Arms: an automatic knowledge engineering tool for learning action models for ai planning. *Knowl. Eng. Rev.* 22:135–152.

Zhuo, H. H.; Yang, Q.; Hu, D. H.; and Li, L. 2010. Learning complex action models with quantifiers and logical implications. *Artificial Intelligence* 174(18):1540–1569.