Process Representation and Planning in Cyc: From Scripts and Scenes to Constraints

STUART AITKEN
AIAI,
University of Edinburgh
02/12/2001

This paper describes on-going work being carried out under the Rapid Knowledge Formation (RKF) program. The representation of process and plan knowledge within the Cyc ontology is the central focus of this paper. Within the RKF program, we are also concerned with the advanced knowledge elicitation techniques that are main theme of the program as a whole. We begin by discussing the inclusion of the Process Specification Language, being developed by NIST, in the Cyc ontology[4, 5], then briefly introduce an alternative approach to process representation based on Scripts. Finally, we link process representation to planning.

1 The Process Specification Language

The Process Specification Language (PSL) is being developed by NIST as a proposed standard for the representation of processes. It is the outcome of an extensive survey of plan and process representations, which included PIF, IDEF3, <I-N-OVA> and CPR. While PSL was developed to describe to concept of *process* in industry and commerce, the basic ideas of activities which are decomposable and orderable can be applied more widely.

1.1 Encoding PSL in CycL

The basic categories of PSL[6] are *Activity*, *ActivityOccurrence*, *Object* and *TimePoint*. These top-level collections are mutually exclusive in PSL.

Important relations in the PSL Core theory include: *occurrenceOf* which relates activity occurrences to activity types; *isOccurringAt* which holds between an activity occurrence and the timepoints between or equal to its begin and end; and *participatesIn* which relates an object to an activity occurrence at a timepoint. Relations specifying the subactivies of a composite activities, and the conditions and effects of an activity occurrence are defined in the Activity Occurrence and States Extensions to the Core. These theories provide instance-level relations, that is, they hold of concrete occurrences or objects. As our contribution to the RKF program, PSL theories have been successfully integrated into the Cyc Upper Ontology. In addition, processes in biochemistry have been encoded using this ontology.

1.2 Repeated Processes

A common property of biochemical processes is the repetition of a step *n* times, to achieve a given effect or state. Repetition needs to be formally encoded. An ontology of properties of repeated processes was developed, and is presented in Figure 1. Briefly, repeated activities are modelled as terminating, uniform (consisting of repetitions of the same activity), and composite. Begin and end points may/may not overlap, the number of repetitions may be specified and a postcondition may be achieved on termination.

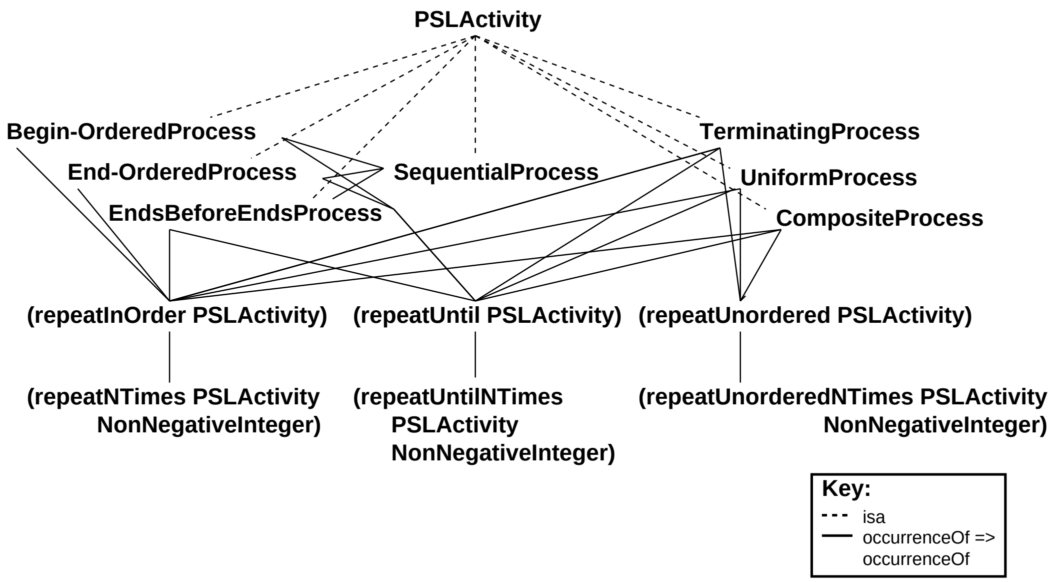

Figure 1: An ontology of repeated processes

2 Scripts for Process Representation

Scripts are a purely type-level encoding of events, subevents, and subevent ordering (in contrast to PSL where the encoding is at the instance level). This encoding of processes is very amenable to knowledge-suggestion rules. A vocabulary of type-level relations has been defined in the Cyc KB, and includes axioms for their interpretation at the instance level. The semantics of the instance level is provided by PSL or similar relations. For example, the subevent types of *RNASplicingProcess* are *SNRNPsBondToIntron* and *IntronExcisedFromPrimaryTranscript*:

```
(firstSubScriptedEventTypes RNASplicingProcess SNRNPsBondToIntron)
(lastSubScriptedEventTypes RNASplicingProcess
          IntronExcisedFromPrimaryTranscript)
```

2.1 Participants in Scripts

Type-level relations express statements such as *(some instance of) ObjectType plays* **Role** *in* **Scene**. For example, an *Intron-Molecule* is the *objectActedOn* in *RNASplicingProcess*. The problem of **identity** arises - the same Intron-Molecule may be involved in several subevents:

```
(sameInstancePlaysRole Intron-Molecule
   SNRNPsFuseEndsOfIntron IntronExcisedFromPrimaryTranscript
   objectActedOn outputs)
```

We have developed new vocabulary and axioms for participants and identity in Scripts.

2.2 Preconditions and Postconditions in Scripts

A novel type-level treatment of preconditions and postconditions of Scenes in Scripts has been proposed and refined. Again, instance-level relations, e.g. *preconditionFor-PropSit*, follow from the type-level relations,

```
PDDL description
Action:         (connect x y)
Preconditions:  (near x y) (shapeOfObject x HelixShaped)
Postconditions: (connectedTo x y)

Type-level description of conditions:
(preconditionOfScene Connect near
                     (TypeArgSpec-BinaryFn DNAMolecule RNAMolecule))
(postconditionOfScene Connect connectedTo
                     (TypeArgSpec-BinaryFn DNAMolecule doneBy))

Type-level description of identity:
(identityInConditionsOfScene-Arg1Arg1 Connect
             preconditionFor-PropSit near
             postconditionFor-PropSit connectedTo)
```

Figure 2: Action descriptions

e.g. *preconditionOfScene*, shown in Figure 2.

Translations from the Scene vocabulary to PDDL and to a CycL constraint formulation have been developed.

3 Process Representation and Planning

Planning is dependent on efficient encodings of action-related knowledge. The algorithms used may be general purpose, such as SAT and CSP, or specialised[1, 2, 3]. PDDL is a commonly-used input format for planning problems. The semantics of actions in PDDL are constraint-based, and can be characterised using our instance-level vocabulary as follows:
∀*x* ∀*y* ∃*z (isa z Connect)* ∧
(preconditionFor-PropSit (near x y) z) ∧
(postconditionFor-PropSit (connectedTo x y) z)
while the semantics of Scenes are:
∀*z* ∃*x* ∃*y (isa z Connect)* ∧
(preconditionFor-PropSit (near x y) z) ∧
(postconditionFor-PropSit (connectedTo x y) z)
therefore a transformation is required in order to construct both the PDDL definitions **and** the equivalent CycL constraint theory from the intial Scene definition. Once this transformation is performed, planning can be performed by an external planner which uses an equivalent representation of the problem as exists in Cyc. The results of planning–the plan–is an extension of the constraint theory and the world state theory. Figure 3 shows the sequence of transformation and planning processes.

In future work, the basic PDDL definitions of actions that we construct will be enhanced with type information. Additional ontology constraints including properties of relations will also be incorporated.

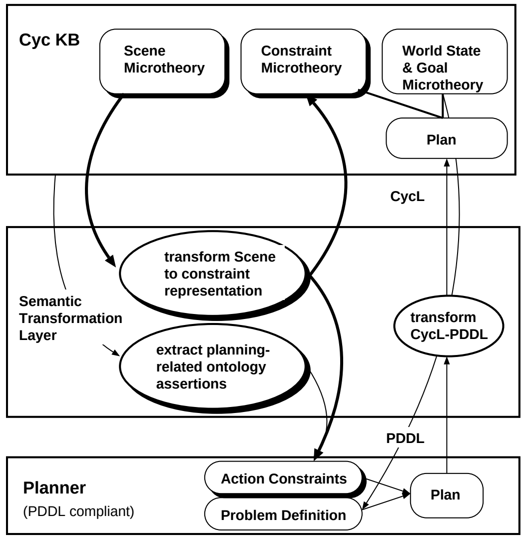

Figure 3: Integrating Cyc with an external Planner

Acknowledgements

This work is sponsored by the Defense Advanced Research Projects Agency (DARPA) under subcontract 00-C-0160-01 with Cycorp on BAA99-35. The U.S. Government is authorised to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either express or implied, of DARPA, Rome Laboratory or the U.S. Government.

References

[1] S. Kambhampati, E. Parker, and E. Lambrecht. Understanding and extending graphplan. In *Proceedings of the 4th European Conference on Planning*, 1997.

[2] H. Kautz and B. Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings of AAAI '96*, 1996.

[3] H. Kautz and B. Selman. Unifying SAT-based and Graph-based planning. In *Proceedings of*, 1999.

[4] D.B. Lenat. Leveraging cyc for hpkb intermediate-level knowledge and efficient reasoning, 1997. URL: `http://www.cyc.com/hpkb/proposal-summary-hpkb.html`.

[5] D.B. Lenat and R.V. Guha. *Building large knowledge-based systems. Representation and inference in the Cyc project*. Addison-Wesley, 1990.

[6] C. Schlenoff. The process specification language (psl) overview and version 1.0 specification, 2000. NIST Internal Report (NISTIR) 6459, http://www.mel.nist.gov/psl/.