Integrating Descriptions and Classification into a Predicate Calculus Framework

Robert MacGregor
USC/Information Sciences Institute
macgregor@isi.edu

Abstract

The PowerLoom system under development at USC/ISI implements both a general-purpose theorem prover, for reasoning with predicate calculus expressions, and a description classifier. The PowerLoom classifier borrows some ideas from its precedessor, the Loom classifier, but it also implements some novel classification techniques. These new techniques enable it to handle the increased expressive power of the logic, and they facilitate the implementation of "lazy" classification.

1 Introduction

Propositions and atomic terms form the two principal building blocks of a typical deductive system based on the predicate calculus. The PowerLoom system [PowerLoom URL] uses descriptions as a third building block in a predicate calculus framework. Below we briefly discuss how descriptions are used by the PowerLoom theorem prover, and how a description classifier operates within the deductive system.

PowerLoom reasons with expressions phrased in a higher-order variant of KIF. A description in PowerLoom is represented by the KIF **kappa** operator: **(kappa (?x_1…?x_k) Px_1…x_k)**. Description terms appear in three places within PowerLoom:

- they denote relations and classes (this is traditional);
- they are used to represent queries;
- they provide an alternate representation for material implications (universally-quantified propositions).

Descriptions may or may not have names. A description with name R denotes the relation R. PowerLoom distinguishes between classes and unary relations, but both categories are represented as unary descriptions. PowerLoom uses the same representation and the same inference machinery to evaluate queries and descriptions—a query **(retrieve (?x_1…?x_k) Px_1…x_k)** is internalized as **(kappa (?x_1…?x_k) Px_1…x_k)**. The first time that a description is evaluated, an optimizer orders its clauses, and prunes out any redundant clauses. The optimization is cached, so that subsequent evaluations use the already-optimized query.

Material implications of the form

(forall (?x_1…?x_k) (=> Px_1…x_k Qx_1…x_k))

may be rewritten as

```
(subset-of (kappa (?x1…?xk) Px1…xk)
           (kappa (?x1…?xk) Qx1…xk))
```

This representation was first tried out in an experimental predicate calculus classifier described in [MacGregor 94]. PowerLoom elects to perform this transformation in several contexts; the transformation is particularly useful in the case that **(kappa (?x_1…?x_k) Qx_1…x_k)** is equivalent to an existing named relation.

Example 1:

```
(forall (?x) (=> (Person ?x) (Animal ?x)))
```

becomes

```
(subset-of Person Animal)
```

Example 2:

```
(forall (?x ?y)
    (<=> (Daughter ?x ?y)
         (and (Child ?x ?y)(Female ?y))))
```

becomes

```
(and
  (subset-of
    Daughter
    (kappa (?x ?y)
           (and (Child ?x ?y)(Female ?y))))
  (subset-of
    (kappa (?x ?y)
           (and (Child ?x ?y)(Female ?y)))
     Daughter))
```

PowerLoom uses 'subset-of' propositions to represent subsumption links—within a hierarchy of descriptions/relations, it represents the sentence "D1 subsumes D2" by the proposition "(subset-of D2 D1)". Specialized algorithms and indices are applied to the taxonomy defined by the 'subset-of' relation (the Cyc relation 'genls' is equivalent to the PowerLoom 'subset-of').

The PowerLoom backtracker backchains by tracing backwards across 'subset-of' propositions: Given a goal (G ?x1 … ?xk), it finds subgoals/antecedents of G by retrieving the first arguments of all 'subset-of' propositions having G as their second argument. Benefits of this architecture include:

- Recursive optimization of subgoals—every description encountered during backtracking is optimized (upon first use);
- Consolidation of logical reasoning based on implication ('**=>**') with sets/subset reasoning based on 'subset-of';
- Straightforward integration of a description classifier—in this framework, the job of a classifier is to compute all 'subset-of' relations.

2 The Classifier

Here we briefly discuss some features of the PowerLoom classifier.

Subsumption relations are computed using universal introduction. To prove "(subset-of P Q)" we do the following:

> Fork a context; introduce one skolem per kappa variable; inherit P onto the skolem(s); trigger forward chaining within the context; bind each of Q's kappa variables onto the corresponding skolem; and test whether Q is true in the forked context.

There is no explicit notion of *definition* in PowerLoom. The closest analog is a biconditional implication (e.g., the biconditional applied to the relation 'Daughter' in Example 2). We call a subsumption semantics which takes into account all axioms (including material implications, biconditionals, and ground facts) *extensional* subsumption, while we call the traditional description logic notion of subsumption *intensional*. Our experience with [Loom URL] (which supports only intensional subsumption computations) suggests that most users have trouble distinguishing between intensional and extensional subsumption, and also that of the two, extensional is the more useful. Hence, for PowerLoom, we have elected to ignore intensional subsumption. However, it would quite easy to add the notions of 'definition' and 'intensional subsumption' to the architecture.

PowerLoom divides the traditional notion of description classification into two phases, "upclassification" and "downclassification". Upclassification of a description D1 involves computing all subsumers of D1, i.e., all D2s such that (subset-of D1 D2). Downclassification involves computing all subsumees of D1, i.e., all D2s such that (subset-of D2 D1). During upclassification of a description D1, it is not necessary to perform a complete universal introduction computation between D1 and each potential subsumer. Instead, we fork a context, introduce skolem(s), inherit D1 onto the skolem(s), and forward propagate one time only. The subsumption test between D1 and each D2 consists only of binding D2's kappa variables onto the skolems and evaluating D2 (using backtracking deduction).

In PowerLoom, downclassification of a description D1 is performed by upclassifying each potential subsumee of D1. Thus, the timesaving "trick" described in the previous paragraph applies to all subsumption computations. Unlike the Loom classifier, which aggressively classifies all concepts/descriptions introduced into the system, the PowerLoom classifier is lazy—it is normally triggered only by a call to the procedure 'find-direct-supers-and-subs'. Calling that procedure on a (named or unnamed) description D causes D to be both upclassified and downclassified. Most of the complexity of the PowerLoom classifier involves minimizing the number of upclassifications triggered by one of these calls.

The division of the traditional classification algorithm into upclassify/downclassify components yields benefits in both batch and incremental uses of the classifier. Because the opportunities for pruning are less during downclassification than for upclassification, we prefer to avoid downclassification whenever possible. If our objective is to classify all descriptions (i.e., batch classification), we can accomplish our objective by upclassifying each individual description; downclassification is avoided entirely. In many situations (perhaps the majority) users want to compute direct supers of a description D, but don't care about subs. In this case, it is sufficient to upclassify D (again avoiding downclassification).

The motivation for this split sprang from some work we did in developing a parallel classifier. There, we noted that it is easier to minimize data transfers and to modularize processing if the downclassify step is eliminated.

Acknowledgments

This research was funded under DARPA contracts N00014-94-C-0245 and F30602-97-1-0194.

References

[Loom URL] `www.isi.edu/isd/LOOM`.

[MacGregor 94] Robert M. MacGregor, A Description Classifier for the Predicate Calculus,in *Proceedings of the Twelfth National Conference onArtificial Intelligence, (AAAI 94)*, pp. 213-220, 1994.

[PowerLoom URL]
`www.isi.edu/isd/LOOM/PowerLoom`..