

# DL Reasoning

Stasinos Konstantopoulos

IIT, NCSR 'Demokritos'

10-3-2006

# Overview

## Description Logics

- Definitions

- Some Family Members

## Reasoning Algorithms

- Introduction

- Resolution Calculus

- Tableau Calculus

## Concluding Remarks

# The DL Family

## Definition

- ▶ A family of (mostly) First-Order Dyadic Logics with counting quantifiers, i.e. (mostly) fragments of  $\mathcal{C}^2$
- ▶ Some include features exceed  $\mathcal{C}^2$ , e.g. relation composition.

# Some Family Members

## Some Simple DLs

The following are PSPACE-complete with an acyclic TBox and EXPTIME-complete with a general TBox:

- ▶  $\mathcal{ALC}$ : all the usual connectives, both quantifiers
- ▶  $\mathcal{ALCOR}^+$  (a.k.a.  $\mathcal{SO}$ ): with nominals and transitivity axioms
- ▶  $\mathcal{SON}$  and  $\mathcal{SOQ}$ : with unqualified ( $\leq nR$ ) and qualified ( $\leq nR.C$ ) cardinality restrictions, resp.

## Some Family Members

Some more complex DLs

EXPTIME-complete:

- ▶  $SHOQ$  adds role hierarchies
- ▶  $SHIQ$  takes away nominals and adds role inversal
- ▶  $SHIF$  takes away cardinality restrictions and only allows for functionality axioms, i.e.  $\leq 1R$

Note, however, that:

- ▶  $SHOIQ$  is NEXPTIME-complete
- ▶ in  $-Q$  and  $-N$  only simple roles (neither transitive nor have transitive subroles) are admitted; otherwise even  $SHN$  is undecidable.

# Some Family Members

## Some Implemented DLs

- ▶  $SHIQ^-$  (RacerPro)
  - ▶ number restrictions only for roles without sub-roles
- ▶  $SHIQ$  (KAON2)
- ▶  $SHOIN$  (Pellet)
- ▶  $SHOIQ$  (FaCT++)

## Some Family Members

Implemented DLs with datatype reasoning

- ▶  $SHIQ^-(D^-)$  (RacerPro)
  - ▶ Integers: min/max restrictions
  - ▶ Reals: linear polynomial (in-)equations over the reals or cardinals with order relations
  - ▶ Strings: equalities and inequalities
  - ▶ User-defined: none
  - ▶ no feature chains on concrete domains
- ▶  $SHOIN(D)$  (Pellet)
- ▶  $SHOIQ(D)$  (FaCT++)

OWL DL is  $SHOIN(D)$  and OWL Lite is  $SHIF(D)$

# Deductive Inference

Entailment (semantics) is realised through (syntactic) inference operators.

Inference operators form a semantically empty logical calculus, which we want to be:

- ▶ Sound (yield correct results)
- ▶ Complete (yield *all* correct results)
- ▶ Decidable (yield results in *finite* time)

# Open World vs. Closed World

## Closed World

- ▶ binary predicates: either true or false;
- ▶ failure to prove is considered proof of falsity;
- ▶ negation-as-failure;
- ▶ model is presupposed;
- ▶ reasoner says: 'Give me a model and I'll give you an answer.'

# Open World vs. Closed World

## Open World

- ▶ ternary predicates: true, false, dunno;
- ▶ failure to prove is distinct from proof of falsity;
- ▶ explicit negation;
- ▶ results valid on all models;
- ▶ reasoner says: 'Here's your answer, you figure out what it means in your model.'

# Robinson Rule

## Definition

Robinson (1965):

$R$  is resolved from clauses  $C_1$  and  $C_2$  iff there are literals  $l_1 \in C_1$  and  $l_2 \in C_2$  and substitution  $\theta$  such that:

$$l_1\theta = \neg l_2\theta$$

$$R = (C_1 - \{l_1\})\theta \cup (C_2 - \{l_2\})\theta$$

# Robinson Rule

## Characteristics and Application

- ▶ A general, sound and complete deduction rule.
- ▶ Non-deterministic AND non-deterministically applied.
- ▶ Linear resolution: restriction of the non-deterministic application of the rule. A fixed clause keeps being transformed by resolving it against other clauses in a given set.

## SLD Resolution

- ▶ SLD Resolution: Selection Linear Definite Resolution. Proposed by Kowalski (1979).
- ▶ Linear resolution over definite clauses, using a selection function:
  1. the fixed clause is the query
  2. clauses in the set are definite
- ▶ An oracular function selects which atom in the query to resolve on and which definite clause in the query set to resolve against.

# Warren Abstract Machine

## Top-Down Leftmost Resolution

WAM( $Q$ ) resolves query  $Q = \{q_1, q_2 \dots q_n\}$ :

1. Get the left-most literal  $q$ .
2. For each clause  $p_i$  of predicate  $P$  with head  $q$ , starting with the one provided first:
  - 2.1 Unify with the head of the predicate:
    - 2.1.1 if unification fails, re-iterate.
    - 2.1.2 if unification succeeds, continue.
  - 2.2 Replace  $q$  with the body of  $p_i$  and perform all applicable unifications.
    - 2.2.1 if unification fails, re-iterate.
    - 2.2.2 if unification succeeds, continue.
3. Assign the result to  $Q'$ .
4. If  $Q'$  is empty, succeed. Else WAM( $Q'$ ).

# Warren Abstract Machine

## Characteristics

- ▶ Left-to-right approximates the non-deterministic selection function of SLD Resolution
- ▶ Non-deterministic disjunction is deterministicised by considering the order rules were provided in.
- ▶ Effectively, WAM implements a depth-first, left-to-right traversal of the SLD tree.
- ▶ Semi-complete: can get stuck in infinite recursions.

# Warren Abstract Machine

## PROLOG

- ▶ implements WAM: definite clauses, plus one Horn query.
- ▶ SLDNF-Resolution: Normal Programs (with neg literals) under Closed-World Assumption, after finitely many steps.
- ▶ Cardinality restrictions can be coded using extra-logical predicates:
  - ▶ `findall/3` will retrieve all admissible instantiations
  - ▶ but, a very expensive way to impose cardinality restrictions.
  - ▶ `i.d.b.` can be used for more efficient solutions, but there is no general mechanism provided by the compiler.
- ▶ No concrete domains.

# SLG Resolution

## General bottom-up resolution

SLG Resolution: Selection Linear resolution for General LP, proposed by Chen and Warren (1996).

- ▶ **Tabling:** bottom-up resolution. Allows for infinite loop detection.
- ▶ **Goal delaying:** unground negative literals are delayed. If delay list cannot be closed, a conditional answer is returned (CWA not necessary).
- ▶ The mechanism behind DATALOG, DATALOG<sup>V</sup>, and other deductive database languages.

# SLG Resolution

## LRD-stratified Resolution

LRD-stratified Resolution, proposed by Sagonas and Swift (1998):

- ▶ Left-to-right selection, allows delays.
- ▶ Restricted to stratified Normal-Clause Programmes (terminates).
- ▶ Implementation: SLG-WAM abstract machine, XSB compiler.
- ▶ Datalog systems:
  - ▶ SLG systems
  - ▶ complex terms are not supported, unification is strcmp() up to variable renaming
  - ▶ DES (Prolog), DATALOG++ (XSB)
- ▶ O-Telos:
  - ▶ derived from Datalog and Telos (CML-like, SLD language).
  - ▶ Implemented by CONCEPTBASE (Prolog/Java)

# SLG Resolution

## KAON2

KAON2:

- ▶ Implements  $SHIQ^-$ 
  - ▶ by reduction to  $DATALOG^V$  (Hustadt et al., 2004)
  - ▶  $SHIQ^-(D)$  coming soon
  - ▶ why  $DATALOG^V$  and not  $DATALOG$ ?
- ▶ In: OWL and (a deviation from) SWRL
- ▶ Out: SPARQL

# Analytic Tableaux

## Definition

$$\alpha \frac{X, A \wedge B}{X, A, B}$$

$$\gamma \frac{X, \forall x A}{X, A_x^a}$$

$$\beta \frac{X, A \vee B}{X, A \quad X, B}$$

$$\delta \frac{X, \exists x A}{X, A_x^a}$$

- ▶  $a$  in  $\delta$ -rules must be a new symbol.
- ▶ a branch is *closed* if it contains  $F$  and  $\neg F$
- ▶ a tableau is *closed* if all its branches are closed.

# Analytic Tableaux

## Comments

- ▶ The ABox can be incorporated in  $X$ , but optimised reasoners will query some form of instance DB.
- ▶ Adaptations for typed variables modify the  $\gamma$  and  $\delta$  rules.
- ▶ How to use them:
  - ▶ iff  $X$  is valid, then the tableau for  $\neg X$  closes
  - ▶ why not construct the tableau for  $X$ ?

# Implementations

## FACT++

- ▶ Implements *SHOIQ(D)*
- ▶ Implemented in C++
- ▶ In: KRSS, DIG
- ▶ Out: DIG

# Implementations

## PELLET

- ▶ Implements *SHOIN*(D):
  - ▶ *SHOIN* with tableaux
  - ▶ datatypes with a separate XSD reasoner
- ▶ Implemented in Java
- ▶ In: OWL, DIG
- ▶ Out: SPARQL, DIG

# Implementations

## RACERPRO

- ▶ Implements  $ALCQHI_{\mathcal{R}^+}(D^-)$ 
  - ▶ plus a an incomplete bit of  $-\mathcal{O}$ : ABox checking, but no TBox inference
- ▶ In: KRSS, RDF, OWL, DIG
- ▶ Out: nRQL, OWL-QL, DIG

# Implementations

## JENA

Not really a reasoner, more of an RDFS validator: no abstract inference, only instance-based.

- ▶ Almost implements  $\mathcal{FL}^{-}\mathcal{R}^{+}\mathcal{HIF}$ :  $\forall R.C$  only checks membership, cannot infer membership.
- ▶ Implemented in Java
- ▶ In: RDFS, OWL, DAML+OIL
- ▶ Out: RDQL

# Implementations

## Other

- ▶ HOOLET: translates *SHIF* to first-order axioms and uses the VAMPIRE first-order prover.
- ▶ Bell Labs CLASSIC (Lisp) and NEOCLASSIC (C++): the *FL<sup>-</sup>ON* reasoners at the core of the PROSE and QUESTAR configurators for AT&T products.
- ▶ BACK, LOOM

## Concluding Remarks


- ▶ Usual trade-off between expressivity and tractability
- ▶ Massive gains in expressive power:
  - ▶ more than 2 variables
  - ▶ cardinality restrictions
- ▶ Real-life problems not to be confused with theoretical complexity results
  - ▶ we perform and complete  $\text{NEXPTIME}$ -complete and undecidable calculations every day
  - ▶ implementations are heavily optimised and handle the hard bits extra-logically
  - ▶ but might as well avoid them, if we can

Weidong Chen and David S. Warren. Tabled evaluation with delaying for general logic programs. *Journal of the ACM*, 43(1): 20–74, January 1996.

Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reducing SHIQ Description Logic to Disjunctive Datalog programs. In *Proc. of the 9th International Conference on Knowledge Representation and Reasoning (KR2004)*, pages 152–162, 2004.

Robert A. Kowalski. *Logic for Problem Solving*, volume 7 of *Artificial Intelligence Series*. North Holland, New York, 1979. URL <http://www.doc.ic.ac.uk/~rak/>.

John A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

Konstantinos Sagonas and Terrance Swift. An abstract machine for tabled execution of fixed-order stratified logic programs. 

*ACM Transactions on Programming Languages and Systems*, 20  
(3):586–634, May 1998.

Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya  
Kalyanpur, and Yarden Katz. Pellet: a practical OWL-DL  
reasoner. URL  
<http://www.mindswap.org/papers/PelletJWS.pdf>.  
Submitted for publication to *Journal of Web Semantics*, 2006.