

Semantic Web Chalk Talk

Amateur intro to description logics

- Talk plan and caveats
- History and motivations
- What is a description logic?
- Smorgasbord of constructors
- $KB = ABox + TBox$
- DL reasoning services and complexity
- Relation to RDF et al
- Application to databases and IT integration

Description logics

What and why?

- A logical formalism for representing information about individuals, classes of individuals and their description
- Structural or object-based (Frames etc)
- Well understood expressiveness/complexity tradeoffs
- Suited to describing data and schemas, hence used in data integration research
- Relevant to, and part of foundation of, RDF and RDFS/OIL/DAML

History – knowledge representation

Semantic networks (Quillian'66 etc)

- nodes + associations
- confusion between domain and language level associations, confused semantics

Frames (Minsky'81)

- concepts + slots (+ facets)
- object structure nice but not much better semantics
- confusion between assertions and descriptions

Various more formalized descendants

- KL-ONE (Brachman'78) – structured inheritance
- Krypton (Brachman et al '83) – Abox/Tbox separation
- DL formalisation – Classic (Borgida'89), Loom (MacGregor'87), Kris (Baader'91) etc

applications in expert systems, data modelling, NLP, software engineering ...

Outline of description logics

- The core is a *concept language*
 - use it for expressing factual assertions, intensional knowledge and queries
- Concepts – denote entities, classes
Example: Student $\{ x \mid \text{STUDENT}(x) \}$
- Roles – denote properties, relations
Example: Friend $\{ (x, y) \mid \text{FRIEND}(x,y) \}$
- Constructors for concept expressions
Example: Student \cap \exists Friend.Rich
 $\{ x \mid \text{STUDENT}(x) \wedge \exists y. \text{FRIEND}(x,y) \wedge \text{RICH}(y) \}$
- Individuals – instances of classes
Example: a27, Colin, Green ...

Outline of description logics

Things to note ...

- A subset of function-free, first order logic (FOL)
 - L_3 – at most three variable names
 - Propositional modal logic
- Predicates only, no explicit variables
 - Effectively using lambda expressions to map from FOL
 - $\text{Student} \equiv \lambda x. \text{STUDENT}(x)$
- These concept expressions are used to express terminology information and instance information
- Why not just use FOL directly?
 - greater structuring of the knowledge, can guide inference
 - more efficient (like decidable) inference procedures
- The choice of constructors \cap, \exists etc give expressivity/performance tradeoffs

Sets of constructors

<u>Construct</u>	<u>Syntax</u>	<u>Language</u>
Concept	A	FL
Role name	R	
conjunction	$C \cap D$	
value restriction	$\forall R.C$	
existensial quantification	$\exists R$	
top	\top	AL*
bottom	\perp	
negation (C)	$\neg A \neg C$	
disjunction (U)	$C \cup D$	
existential restriction (E)	$\exists R.C$	
number restrictions (N)	$(\geq n R) (\leq n R)$	
collection of individuals (O)	$\{a_1 \dots a_n\}$	
Role heirarchy	$R \subseteq S$	H
Inverse role	R^{-}	I
Qualified number restriction	$(\geq n R.C) (\leq n R.C)$	Q

with transitive roles also called
S and hence SHIN, SHIQ

Semantics

- Semantics defined by an *interpretation* (Δ^I, I)
- nonempty set, domain of discourse Δ^I
- an interpretation function I which maps:
 - every concept to a subset of Δ^I
 - every role to a subset of $\Delta^I \times \Delta^I$
- also require the unique name assumption
if $a \neq b$ then $a^I \neq b^I$
- a *model* for C is an interpretation where C^I is not empty
a concept is *satisfiable* if it has a model

Constructor semantics

<u>Construct</u>	<u>Syntax</u>	<u>Semantics</u>
Concept	A	$A^I \subseteq \Delta^I$
Role name	R	$R^I \subseteq \Delta^I \times \Delta^I$
conjunction	$C \sqcap D$	$C^I \cap D^I$
value restriction	$\forall R.C$	$\{x \in \Delta^I \mid \forall y.(x,y) \in R^I \Rightarrow y \in C^I\}$
existensial quantification	$\exists R$	$\{x \in \Delta^I \mid \exists y.(x,y) \in R^I\}$
top	\top	Δ^I
bottom	\perp	\emptyset
negation (C)	$\neg A \neg C$	$\Delta^I \setminus C^I$
disjunction (U)	$C \sqcup D$	$C^I \cup D^I$
existential restriction (E)	$\exists R.C$	$\{x \in \Delta^I \mid \exists y.(x,y) \in R^I \wedge y \in C^I\}$
number restrictions (N)	$(\geq n R) (\leq n R)$	$\{x \in \Delta^I \mid \#\{y \mid (x,y) \in R^I\} \geq n\}$
collection of individuals (O)	$\{a_1 \dots a_n\}$	$\{a_1^I \dots a_n^I\}$
Role heirarchy	$R \subseteq S$	$R^I \subseteq S^I$
Inverse role	R°	$\{(y,x) \mid (x,y) \in R^I\}$
Qualified number restriction	$(\geq n R.C) (\leq n R.C)$	$\{x \in \Delta^I \mid \#\{y \mid (x,y) \in R^I \wedge y \in C^I\} \geq n\}$

Simple examples of concept expressions

$\text{ANIMAL} \sqcap \neg \text{HERBIVORE}$

$\text{MALE} \sqcap \exists \text{ChildOf.MALE}$

$\text{CUSTOMER} \sqcap \forall \text{purchaseItem.EXPENSIVE} \sqcap (\geq 3 \text{ Child}) \sqcap \exists \text{creditRating}.\{\text{HIGH}, \text{VHIGH}\}$

Note on \exists and \forall , consider *creditRating*

- There could be an assertion $\text{creditRating}(c, \text{LOW})$ as well and still satisfy the concept expression
- If used $\forall \text{creditRating}.\{\text{HIGH}, \text{VHIGH}\}$ with no other cardinality restrictions then an individual with no creditRating assertion could satisfy the expression

KB, TBox and ABox

- Concept expressions on their own don't say much
- A key notion in DL work is to separate conceptual or terminological knowledge (TBox) from instance or assertional knowledge (ABox)
 - TBox: $\text{HAS-SON} = \exists \text{ChildOf.MALE}$
 - ABox: $\text{MALE}(\text{Dave}), \text{MALE}(\text{Colin}), \text{ChildOf}(\text{Dave}, \text{Colin})$
- Reasoning is then done using a "knowledge base" which is a pairing of the two
 - $\Sigma = \langle \text{TBox}, \text{ABox} \rangle$
- In original systems the ABox also includes assertional *rules* which make up the domain theory, rather than conceptual model
 - $(\geq 3 \text{ Child}) \cap \forall \text{purchaseItem.EXPENSIVE} \rightarrow \text{TARGET-CUSTOMER}$

TBox, intensional knowledge

- Typical TBox statements have form:

$$\left. \begin{array}{l} A \subseteq C \\ A = C \end{array} \right\} \text{ simple TBox}$$

$$\left. \begin{array}{l} C \subseteq D \\ C = D \end{array} \right\} \text{ free TBox (choice of semantics if recursive)}$$

- Reasoning services:

concept satisfiability: $\Sigma \not\models C \equiv \perp$

concept subsumption: $\Sigma \models C \subseteq D$ i.e. $\Sigma \not\models C \cap \neg D \equiv \perp$

consistency: $\Sigma \not\models$

- subsumption gives basis for classification, consistency checking, instance checking, retrieval ...

TBox reasoning

Structural algorithm

- For simple DLs structural algorithm to test subsumption:
 - Convert to conjunctive normal form
 - Test each term and recurse

- Conjunctive normal form (flatten, factorize \forall)

$$C \equiv \forall \text{child.Poor} \cap (\text{Person} \cap \exists \text{child}) \cap \forall \text{child.Student}$$

$$D \equiv (\text{Person} \cap \exists \text{child}) \cap \forall \text{child.Student}$$



$$C \equiv \text{Person} \cap \exists \text{child} \cap \forall \text{child.}(\text{Student} \cap \text{Poor})$$

$$D \equiv \text{Person} \cap \exists \text{child} \cap \forall \text{child.Student}$$

- Test C subsumes D if each term C_i satisfies:

C_i is form atomic or $\exists R$, then there is a D_j with $D_j = C_i$

C_i is form $\forall R.C'$, then there is a D_j with $D_j = \forall R.D'$ and C' subsumes D'

- linear complexity and *sound*
- only *complete* for FL (e.g. $A \cup \neg A$ subsumes everything)

$(\geq 2 (\text{child} \cap \text{son})) \cap (\geq 2 (\text{child} \cap \text{daughter})) \cap (\forall \text{son.Male}) \cap (\forall \text{daughter.}\neg \text{Male})$

$\subseteq (\geq 4 \text{child})$



TBox reasoning

Tableaux calculus and constraint systems

- Use model theoretic semantics
To check if a concept C is satisfiable try to build the most general possible model of the theory with the constraint that C is satisfiable – either succeed or get contradiction.
Model is represented by a set of *constraints*
Build model using *propagation rules*
- Constraints
 $x:C \quad xRy \quad x \neq y$
where x and y are general variables, not specific instances
- Propagation rules on constraint system S
 \cap : if $x:(C \cap D)$ in S then add $\{x:C, x:D\}$ to S
 \cup : if $x:(C \cup D)$ in S then add $\{x:E\}$ to S , E is one of C or D
 \forall : if $x:(\forall R.C)$ in S then add $\{y:C\}$ to S , y is an R -successor of x in S
 \exists : if $x:(\exists R.C)$ in S then add $\{xRy, y:C\}$ to S , if no R -successor of x in S
missed out lots of checks to ensure not repeating work already done
more complex when have role intersection
need blocking rules when have transitive roles

TBox reasoning

Tableaux calculus and constraint systems (2)

- Propagation rules generate an and/or search tree
- Branches terminate in contradictions:
 $\{x:\perp\}$ $\{x:A, x:\neg A\}$ etc
- OR nodes generated by nondeterministic choice rules (e.g. \cup)
Responsible for exponential growth in number of constraint systems
- AND nodes generated by checks of successors of variables
Responsible for exponential growth in size of single constraint system
- Lots of heuristic search techniques and short-cuts provide good typical-case performance – Horrocks, Patel-Schneider
- Several implementations available
E.g. FaCT (Corba interface)

TBox reasoning Complexity results

FL, AL	P
ALE, ALR, ALER, ALU, ALUN	NP
ALC[O] etc up to SI	PSpace
SHIN/Q (simple roles in restrictions)	PSpace
PDL (role composition, complement)	EXptime
SHIQ with transitive roles	NExptime?
KL-ONE	undecidable

ABox

- Assertions about individuals
 $C(a) \quad R(a,b)$
- Instance checking:
 $\Sigma \models C(a)$
Extra complexity from case reasoning.
 $\text{interesting} = (\text{plant} \cap (\geq 2 \text{ medialtem}).\text{visual})$
 $\cup (\text{animal} \cap \exists \text{medialtem})$
 $\text{plant} \cup \neg \text{animal} = \text{T}$
 $\text{picture} \subseteq \text{visual}$
 $\text{medialtem}(\text{grok}, p1) \quad \text{picture}(p1)$
 $\text{medialtem}(\text{grok}, p2) \quad \text{picture}(p2)$
 $\text{interesting}(\text{grok})?$
- Epistemic constructor K (use for KC or KR)
 $K\text{animal} \cap \exists \text{medialtem}$

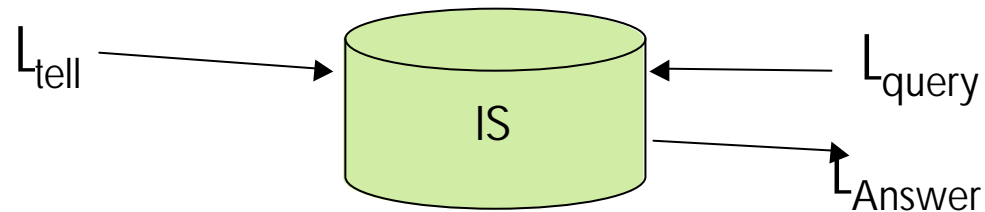
Relation to RDF stack?

- RDF good for ABox, assertions about the world
Close match concept `rdf:type`, role `rdfs:property`
- RDFS, DAML are TBox languages
OIL is SHIQ (? I think)
DAML+OIL also has nominals and concrete types, still decidable
but NExptime
- Use DL reasoners for:
 - ontology consistency checking
 - deductive queries (UNA?)
 - automatic concept hierarchies (user tool, for query generalization/refinement)
 - translating between DB structures
- [SmartGraph ~ KB = ABox + TBox]

DL and data modelling

- Can map ER model to reasonable DL
Relations -> concepts (reified) with n roles
- Reasonable mapping for OO models to DL
But just for the representation, not the behaviour
- Some advantages
 - ontological organisation (concept hierarchies etc)
 - consistency checking
 - schema refinement and inter-schema organization (discover equivalences)
 - greater expressive power (e.g. complement)

DLs and information systems



- Use DL in L_{tell}
 - allows us to insert indefinite information (≥ 2 medialtem)
 - yet still obtain answers using subsumption checking
- Use DL in L_{query}
 - query is just a concept C , find all a such that $C(a)$
 - query checking (unsatisfiable)
 - lattice of queries for generalization/iterative refinement
 - query result caching indexed by query lattice
- Use DL in L_{answer}
 - descriptive, abbreviated, intensional answers

Mapping DL to relational algebra

- There is a formal mapping from DLs like ALC to relational algebra
- Can use this to implement a DL ABox on an RDBMS or ORDMBS database so query-by-description is efficiently mapped into SQL (e.g. Calvanese'98, Borgida & Brachman'93)

Information system integration using logical views

- Several data sources, different but overlapping conceptual models, want to combine them
Virtual mediator
- Approach 1 “local as view”
 - build global conceptual model (global predicates)
 - define data sources as views into this model, but no guarantee of completeness
 - query expressed in global model
 - answer by finding all conjunctive queries over the views that are contained in the top query (subsumption again)
 - e.g. InformationManifold – uses a DL
 - actually uses Carin which has DL TBox but augments with horn clause expressions using DL predicates and Concept expressions

Information system integration using logical views (2)

- Approach 2 – “global as view”
 - mediator translates to intermediate concept objects (views)
 - could use DL or Datalog for this, though Tsimmis doesn't
 - formulate query directly in these intermediate objects
 - simple rewrite for query processing
 - only exports specific object collection, not whole logical model so legitimate queries that can't be satisfied

Side note

XML database query by DTD

- suggestive work using DL to describe XML documents (Calvanese'99)
- XML tree described as a set of DL assertions
- DTD described as set of DL concepts
- Makes it possible to build a DTD hierarchy
- and so query a collection of XML docs for all docs that satisfy an arbitrary DTD

Summary

- Well understood formalism with rich pile of inference machinery out there
- Compatible with RDF/DAML/OIL (somewhat)
- Well suited to integration of information sources and handling of semi-structured information (interest of semantic web group)