Description Logics

What Are Description Logics?

- A family of logic based Knowledge Representation formalisms
  - Descendants of semantic networks and KL-ONE
  - Describe domain in terms of concepts (classes), roles (relationships) and individuals
- Distinguished by:
  - Formal semantics (typically model theoretic)
    - Decidable fragments of FOL
    - Closely related to Propositional Modal & Dynamic Logics
  - Provision of inference services
    - Sound and complete decision procedures for key problems
    - Implemented systems (highly optimized)

Description Logics

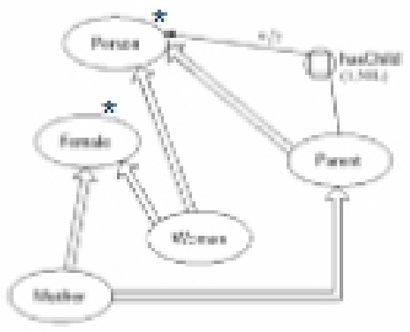
- Major focus of KR research in the 80's
  - Led by Ron Brachman – (AT&T Labs)
  - Grew out of early network-based KR systems like semantic networks and frames.
- Major systems and languages –
  - 80s: KL-ONE, NIKL, KANDOR, BACK, CLASSIC, LOOM
  - 90s: FACT, RACER,
  - 00s: DAML+OIL, OWL
- Used as the basis for the Semantic web languages DAML+OIL and OWL
- Some (one) commercial systems

Description Logics

- Thought to be well-suited for the representation of and reasoning about
  - ontologies
  - terminological knowledge
  - Configurations and configuration problems
  - database schemata
    - schema design, evolution, and query optimization
    - source integration in heterogeneous databases/data warehouses
    - conceptual modeling of multidimensional aggregation

Example of Network KR

- Person, Female, etc are concepts
- hasChild is a property of Person
  - hasChild relates Parent to Person
  - Nil means Infinity. A Parent is a Person with between 1 and infinity children
- Large arrows are "IS-A" links
  - A Mother is a (specialization of a) Parent
- Concepts are either primitive or definitions.
  - Primitive concepts have only necessary properties
  - Defined concepts have necessary and sufficient conditions.



Graphical notation introduced by KL-ONE

DL Paradigm

- A **Description Logic** is mainly characterized by a set of constructors that allow one to build complex descriptions or terms out of **concepts** and **roles** from atomic ones
  - **Concepts** correspond to classes
    - and are interpreted as sets of objects,
  - **Roles** correspond to relations
    - and are interpreted as binary relations on objects
- Set of axioms for asserting **facts** about concepts, roles and **individuals**

Basic Concepts of a DL

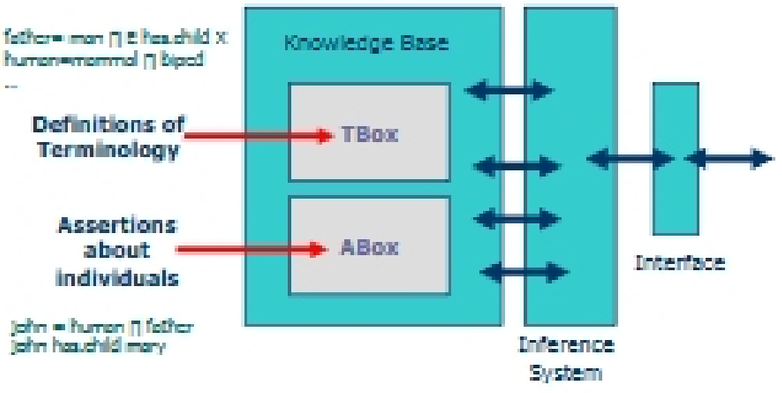
- Individuals are treated exactly the same as constants in FOL.
- Concepts are exactly the same as Unary Predicates in FOL.
- Roles are exactly the same as Binary Predicates in FOL.

Descriptions

- Just Like in FOL, what we are dealing with (ultimately) are sets of individuals and relations between individuals.
- The basic unit of semantic significance is the *Description*.
- *"We are **describing** sets of individuals"*
- Description logics differ in the operators they allow
- If a "happy father" is a man with both a son and a daughter and all of whose children are either rich or happy, then we can describe the concept in a typical DL as

  $\text{HappyFather} = \text{Man} \sqcap \exists \text{hasChild.Female} \sqcap \exists \text{hasChild.Male} \sqcap \forall \text{hasChild.}(\text{Rich} \sqcup \text{Happy})$

Typical Architecture



The division into TBox and ABox doesn't have a logical significance, but is made for conceptual and implementation convenience.

A family of languages

- The expressiveness of a description logic is determined by the operators that it uses.
- Add or eliminate certain operators (e.g., ¬, ∪), and the statements that can be expressed are increased/reduced in number.
- Higher expressiveness implies higher complexity
- AL or Attributive Language is the base and includes just a few operators
- Other DLs are described by the additional operators they include

AL: Attributive Language

Constructor	Syntax	Example
atomic concept	C	Human
atomic negation	~ C	~ Human
atomic role	R	hasChild
conjunction	C ^ D	Human ^ Male
value restrict.	∀R.C	Human ∀ hasChild.Blond
Existential rest. (lim)	∃R	Human ∃ hasChild
Top (univ. conc.)	T	T
bottom (null conc)	⊥	⊥

for concepts C and D and role R

ALC

ALC is the smallest DL that is propositionally closed (i.e., includes full negation and disjunction) and include booleans (and, or, not) and restrictions on role values

Constructor	Syntax	Example
atomic concept	C	Human
negation	**~ C**	**~ (Human V Ape)**
atomic role	R	hasChild
conjunction	C ^ D	Human ^ Male
disjunction	**C V D**	**Nice V Rich**
value restrict.	∀R.C	Human ∀ hasChild.Blond
Existential rest.	**∃R.C**	**Human ∃ hasChild.Male**
Top (univ. conc.)	T	T
bottom (null conc)	⊥	⊥