Description logics

Kamal Lodaya

The Institute of Mathematical Sciences, Chennai

July 2017

Reference: Franz Baader and Carsten Lutz, Description logic, *Handbook of modal logic*, Elsevier, 2007:757–819.

Knowledge representation

- Older work: semantic networks (Quillian 1967), frames (Minsky 1981)
- (Minsky 1981) rejects the use of logic for representation
- (Brachmann 1978) suggests structured semantic networks
- (Hayes 1979) advocates thinking of inference mechanisms on frames as logics

Knowledge representation

- Older work: semantic networks (Quillian 1967), frames (Minsky 1981)
- (Minsky 1981) rejects the use of logic for representation
- (Brachmann 1978) suggests structured semantic networks
- (Hayes 1979) advocates thinking of inference mechanisms on frames as logics

Question: Should AI use logic for knowledge representation?

Logics

First-order logic	Description logics	Multi-modal logics
Unary predicates Binary predicates	Concepts Roles	Propositions Modal operators

Logics

First-order logic	Description logics	Multi-modal logics
Unary predicates Binary predicates	Concepts Roles	Propositions Modal operators

Description and modal logics have variable-free notation compared to first-order logic

$$\text{FOL:} \quad \begin{cases} Man(x) \wedge \\ \exists y(married(x, y) \wedge Doctor(y)) \wedge \\ \forall y(child(x, y) \supset Happy(y)) \end{cases}$$

$$\text{DL:} \quad Man \sqcap \exists married.Doctor \sqcap \forall child.Happy$$

$$\text{ML:} \quad Man \wedge \langle married \rangle Doctor \wedge [child] Happy$$

DL and ML notations are very close, noticed by (Schild 1991), borrowed ideas from ML into DL; DL ideas have also been borrowed into ML

Description logics AL to ALC

$$
\begin{aligned}
C \in AL &::= \top \mid \bot \mid P \mid \neg P \mid C \sqcap D \mid \forall r.C \mid \exists r.\top \\
C \in ALE &::= \top \mid \bot \mid P \mid \neg P \mid C \sqcap D \mid \forall r.C \mid \exists r.C \\
C \in ALU &::= \top \mid \bot \mid P \mid \neg P \mid C \sqcup D \mid C \sqcap D \mid \forall r.C \mid \exists r.\top \\
C \in ALC &::= \top \mid \bot \mid P \mid \neg C \mid C \sqcup D \mid C \sqcap D \mid \forall r.C \mid \exists r.C
\end{aligned}
$$

- AL is the core description logic, conjunctions $C \sqcap D$ and value restrictions $\forall r.C$ were thought to be basic for describing concepts
- ALE has existential restrictions $\exists r.C$, these are used for example in large medical ontologies
- ALU and ALC include full propositional logic, introduced by (Schmidt-Schauß and Smolka 1991)

Domain and interpretation

$$C \in ALC ::= \quad \top \mid \bot \mid P \mid \neg\, C \mid C \sqcup D \mid C \sqcap D \mid \forall r.C \mid \exists r.C$$

Domain $\mathcal{D}$ and interpretation map $\mathcal{I}$

- $(P)^{\mathcal{I}} \subseteq \mathcal{D}, \quad (\top)^{\mathcal{I}} = D, \quad (\bot)^{\mathcal{I}} = \emptyset$
- $(\neg C)^{\mathcal{I}} = \mathcal{D} \setminus (C)^{\mathcal{I}}$
- $(C \sqcap D)^{\mathcal{I}} = (C)^{\mathcal{I}} \cap (D)^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = (C)^{\mathcal{I}} \cup (D)^{\mathcal{I}}$

Domain and interpretation

$C \in ALC ::= \top \mid \bot \mid P \mid \neg\, C \mid C \sqcup D \mid C \sqcap D \mid \forall r.C \mid \exists r.C$

Domain $\mathcal{D}$ and interpretation map $\mathcal{I}$

- $(P)^{\mathcal{I}} \subseteq \mathcal{D}$, $(\top)^{\mathcal{I}} = D$, $(\bot)^{\mathcal{I}} = \emptyset$
- $(\neg C)^{\mathcal{I}} = \mathcal{D} \setminus (C)^{\mathcal{I}}$
- $(C \sqcap D)^{\mathcal{I}} = (C)^{\mathcal{I}} \cap (D)^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = (C)^{\mathcal{I}} \cup (D)^{\mathcal{I}}$
- $(r)^{\mathcal{I}} \subseteq \mathcal{D} \times \mathcal{D}$
- Value restriction
  $(\forall r.C)^{\mathcal{I}} = \{d \in \mathcal{D} \mid$ for all $e : (d, e) \in (r)^{\mathcal{I}}$ implies $e \in (C)^{\mathcal{I}}\}$
- Existential restriction
  $(\exists r.C)^{\mathcal{I}} = \{d \in \mathcal{D} \mid$ for some $e : (d, e) \in (r)^{\mathcal{I}}$ and $e \in (C)^{\mathcal{I}}\}$

Description logics ALCreg and ALCrvm

- Regular expressions: $R ::= \ r \mid R_1 \ \sqcup \ R_2 \mid R_1 \,;\, R_2 \mid R_1^*$
  - $Man \sqcap \exists child.Human \sqcap \forall(child; child^*)Happy$
  - $\sqcup$ of roles is different from disjunction of concepts
  - Universal role: $(r_1 \sqcup \ldots \sqcup r_k)^*$, where all the role names from the concept descriptions are used

Description logics ALCreg and ALCrvm

- Regular expressions: $R ::= \; r \mid R_1 \sqcup R_2 \mid R_1 ; R_2 \mid R_1^*$
  - $Man \sqcap \exists child.Human \sqcap \forall (child; child^*) Happy$
  - $\sqcup$ of roles is different from disjunction of concepts
  - Universal role: $(r_1 \sqcup \ldots \sqcup r_k)^*$, where all the role names from the concept descriptions are used
- Role value maps, e.g. $child \circ friend \subseteq known$
  $(R \subseteq S)^{\mathcal{I}} =$
  $\{d \in D \mid \forall e : (d, e) \in (R)^{\mathcal{I}} \text{ implies } (d, e) \in (S)^{\mathcal{I}}\}$
  - Allowing in full generality is very powerful
  - Restrict paths in role value maps to length one
  - Only allow maps $r \circ r \subseteq r$

Description logics S to SROIQ

S: Roles may be transitive, e.g. *part-of*

SR: Role value maps

Description logics S to SROIQ

S: Roles may be transitive, e.g. *part-of*

SR: Role value maps

SRO: Nominals (singleton concepts), e.g. *President-of-India*

Description logics S to SROIQ

- S: Roles may be transitive, e.g. *part-of*
- SR: Role value maps
- SRO: Nominals (singleton concepts), e.g. *President-of-India*
- SROI: Inverse roles, e.g. *has-part* $\equiv$ *part-of*$^{-}$

Description logics S to SROIQ

S: Roles may be transitive, e.g. *part-of*

SR: Role value maps

SRO: Nominals (singleton concepts), e.g. *President-of-India*

SROI: Inverse roles, e.g. $\mathit{has\text{-}part} \equiv \mathit{part\text{-}of}^{-}$

SROIN: Number restrictions, e.g. $\mathit{Father} \sqcap \leq 2 \mathit{child}.\top$
$(\leq nr.\top)^{\mathcal{I}} = \{d \in D \mid \#\{(d,e) \in (r)^{\mathcal{I}}\} \leq n\}$

SROIQ: Qualifying number restrictions, e.g.
$\mathit{Father} \sqcap \leq 1 \mathit{child}.\mathit{Female}$
$(\leq nr.C)^{\mathcal{I}} = \{d \in D \mid \#\{(d,e) \in (r)^{\mathcal{I}} \mid e \in (C)^{\mathcal{I}}\} \leq n\}$

Can also define other logics like ALUN, ALCN, ...

Terminological boxes

- TBox definitions: $P \equiv C$
  $Father \equiv Man \sqcap \exists child.Person$
  $Mother \equiv Woman \sqcap \exists child.Person$
  $Man \equiv Person \sqcap \neg Woman$
  $Woman \equiv Person \sqcap Female$
  $Parent \equiv Mother \sqcup Father$

Terminological boxes

- TBox definitions: $P \equiv C$
  $Father \equiv Man \sqcap \exists child.Person$
  $Mother \equiv Woman \sqcap \exists child.Person$
  $Man \equiv Person \sqcap \neg Woman$
  $Woman \equiv Person \sqcap Female$
  $Parent \equiv Mother \sqcup Father$
- Expanding a TBox to get a full interpretation:
  $Father \equiv Person \sqcap \neg(Person \sqcap Female) \sqcap \exists child.Person$

Terminological boxes

- TBox definitions: $P \equiv C$
  $Father \equiv Man \sqcap \exists child.Person$
  $Mother \equiv Woman \sqcap \exists child.Person$
  $Man \equiv Person \sqcap \neg Woman$
  $Woman \equiv Person \sqcap Female$
  $Parent \equiv Mother \sqcup Father$
- Expanding a TBox to get a full interpretation:
  $Father \equiv Person \sqcap \neg(Person \sqcap Female) \sqcap \exists child.Person$
- Problem of eliminating acyclic TBoxes:
  $C_1 \equiv \forall r.C_0 \sqcap \forall r.C_0,\ C_2 \equiv \forall r.C_1 \sqcap \forall r.C_1, \ldots$

Hierarchy and description logic SHROIQ

- SHROIQ (similar to OWL): $r \sqsubseteq s$, e.g. $Man \sqsubseteq Human$, only interpretations where $(r)^{\mathcal{I}} \subseteq (s)^{\mathcal{I}}$

Hierarchy and description logic SHROIQ

- SHROIQ (similar to OWL): $r \sqsubseteq s$, e.g. $Man \sqsubseteq Human$, only interpretations where $(r)^{\mathcal{I}} \subseteq (s)^{\mathcal{I}}$
- General concept inclusions: $C_1 \sqsubseteq C_2$
  $Person \sqcap \exists uncle.Father \sqsubseteq \exists cousin.Person$
- Assuming a universal role u: $(T)^{\mathcal{I}} = \forall u. \bigsqcap_{D \sqsubseteq E \in T} \neg D \sqcup E$
- Problem of eliminating GCIs

TBoxes which are not acyclic

$Human \equiv Adam \sqcup Eve \sqcup \exists parent.Human$
(assume $Adam$ and Eve are nominals)
or $Human \equiv \forall parent.Human$

- Let T a TBox with a primitive (not expanded) interpretation $\mathcal{J}$
- Let $Ext_{\mathcal{J}}$ be all extensions of $\mathcal{J}$
- Let $T_{\mathcal{J}} : Ext_{\mathcal{J}} \to Ext_{\mathcal{J}}$ map $\mathcal{I}$ to $T_{\mathcal{J}}(\mathcal{I})$ by
- $(D)^{T_{\mathcal{J}}(I)} = (T(D))^{\mathcal{I}}$ for each defined concept D
- $\mathcal{I}$ is a model of T iff $\mathcal{I}$ is a fixed point (that is, $T_{\mathcal{J}}(\mathcal{I}) = \mathcal{I}$), where $\mathcal{J}$ is $\mathcal{I}$ restricted to a primitive interpretation

Fixed points

- Least fixed point (lfp): smallest of all the fixed points under inclusion $\subseteq$
  $\mathit{Human} \equiv \mathit{Adam} \sqcup \mathit{Eve} \sqcup \exists \mathit{parent}.\mathit{Human}$
- Greatest fixed point (gfp): largest of all the fixed points under inclusion $\subseteq$
  $\mathit{Super\text{-}rich} \equiv \mathit{Rich} \sqcap \mathit{Famous} \sqcap \forall \mathit{works\text{-}with}.\mathit{Super\text{-}rich}$

ABoxes

- Concept assertion: *Logician*(*john*)
  $(C(a))^{\mathcal{I}}$ if $(a)^{\mathcal{I}} \in D$
- Names interpreted as singletons, $(a)^{\mathcal{I}} \in \mathcal{D}$
- Unique names assumption: $a \neq b$ implies $(a)^{\mathcal{I}} \neq (b)^{\mathcal{I}}$

ABoxes

- Concept assertion: $Logician(john)$
  $(C(a))^{\mathcal{I}}$ if $(a)^{\mathcal{I}} \in D$
- Names interpreted as singletons, $(a)^{\mathcal{I}} \in \mathcal{D}$
- Unique names assumption: $a \neq b$ implies $(a)^{\mathcal{I}} \neq (b)^{\mathcal{I}}$
- Role assertion: $(Man \sqcap \exists child.Woman)(john)$
  $(r(a,b))^{\mathcal{I}}$ if $((a)^{\mathcal{I}}, (b)^{\mathcal{I}}) \in (r)^{\mathcal{I}}$
- Interpretation $\mathcal{I}$ is a model of ABox A if it satisfies all assertions in A
- If nominals are available, assume for every name a there is a nominal a, and that u is fresh:
  $$A = \bigsqcap_{C(a) \in A} \exists u.(a \sqcap \mathcal{D}) \sqcap \bigsqcap_{r(a,b) \in A} \exists u.(a \sqcap \exists r.b)$$

More description logic

- Concrete domains: integers, rationals
  $Teenager \equiv Human \sqcap \geq_{10} (age) \sqcap \leq_{19} (age)$
- Aggregation: *sum*, *min*, *max*

Reasoning with concept descriptions (with TBoxes, but without ABoxes)

- C is subsumed by D with respect to T if $(C)^{\mathcal{I}} \subseteq (D)^{\mathcal{I}}$ for every model $\mathcal{I}$ of T
- C is satisfiable with respect to T if C and T have a common model

Reasoning with concept descriptions (with TBoxes, but without ABoxes)

- C is subsumed by D with respect to T if $(C)^{\mathcal{I}} \subseteq (D)^{\mathcal{I}}$ for every model $\mathcal{I}$ of T
- C is satisfiable with respect to T if C and T have a common model
- If bottom concept is available: C is satisfiable wrt T iff C is not subsumed by $\bot$ wrt T
- If negation is available: C is subsumed by D wrt T iff $C \sqcap \neg D$ is unsatisfiable wrt T

Reasoning with concept descriptions (with TBoxes, but without ABoxes)

- C is subsumed by D with respect to T if $(C)^{\mathcal{I}} \subseteq (D)^{\mathcal{I}}$ for every model $\mathcal{I}$ of T
- C is satisfiable with respect to T if C and T have a common model
- If bottom concept is available: C is satisfiable wrt T iff C is not subsumed by $\perp$ wrt T
- If negation is available: C is subsumed by D wrt T iff $C \sqcap \neg D$ is unsatisfiable wrt T

Subsumption algorithms:

ALN: polynomial time

ALE: NP (nondeterministic polynomial time)

ALU,ALUN: co-NP

ALEN,ALC,ALCN: polynomial space

Reasoning with TBoxes and ABoxes

- Name a in A is an instance of C with respect to T if for all models $\mathcal{I}$ of A and T, $(a)^{\mathcal{I}} \in (C)^{\mathcal{I}}$
- A is consistent with respect to T if A and T have a common model

Reasoning with TBoxes and ABoxes

- Name a in A is an instance of C with respect to T if for all models $\mathcal{I}$ of A and T, $(a)^{\mathcal{I}} \in (C)^{\mathcal{I}}$
- A is consistent with respect to T if A and T have a common model
- If negation is available: a in A is an instance of C wrt T iff $A \cup \{\neg C(a)\}$ is inconsistent wrt T
- If bottom is available: A is consistent wrt T iff there is some a in A which is an instance of $\bot$ wrt T

Compound inference problems

- Least common subsumer (lcs) of two concepts
- Most specific concept (msc) of every individual
- Hierarchy: Compute the concept hierarchy
  Algorithm: Incremental, start with $\bot \sqsubseteq \top$, then do top and bottom searches for direct subsumers
- Classification: Given T, compute subsumption relation $\sqsubseteq$ of concept names used in T
  Helps in organization of KB
  Algorithm: multiple invocation of subsumption wrt T, naively $O(n^2)$ for n concept names in T, instead compute concept hierarchy and proceed along it

Compound inference problems

- Realization: Given A, T, a, compute set of concept names C used in T satisfying $C(a)$ which are minimal with respect to subsumption in T
  Helps in browsing and understanding of KB
  Algorithm: multiple invocation of instance checking and subsumption
- Retrieval: Given A, T, C, compute set of individual names a used in A such that $C(a)$ in T
  Used in querying KBs, some of which have huge number of names
  Algorithm: multiple invocation of instance checking

Nonstandard inference problems

- Rewriting to a shorter description, which may be a good approximation:
  $\mathit{Person} \sqcap \forall \mathit{child}.\mathit{Female} \sqcap \exists \mathit{child}.\top \sqcap \forall \mathit{child}.\mathit{Person}$
  $\rightarrow \mathit{Parent} \sqcap \forall \mathit{child}.\mathit{Woman}$
- Matching patterns:
  $\mathit{Man} \sqcap \exists \mathit{child}.(\mathit{Man} \sqcap X) \sqcap \exists \mathit{spouse}.(\mathit{Woman} \sqcap X)$
  is matched by
  $\mathit{Man} \sqcap \exists \mathit{child}.(\mathit{Man} \sqcap \mathit{Tall}) \sqcap \exists \mathit{spouse}.(\mathit{Woman} \sqcap \mathit{Tall})$

Nonstandard inference problems

- Unification:
  $\forall child.\forall child.Rich \sqcap \forall child.Rmr$ and
  $Acr \sqcap \forall child.Acr \sqcap \forall child.\forall spouse.Rich$
  are unified by
  $Rmr \equiv Rich \sqcap \forall spouse.Rich,\ Acr \equiv \forall child.Rich$ to the equivalent descriptions:
  $\forall child.\forall child.Rich \sqcap \forall child.(Rich \sqcap \forall spouse.Rich)$
  and
  $\forall child.Rich \sqcap \forall child.\forall child.Rich \sqcap \forall child.\forall spouse.Rich$

Trade-offs in terminological reasoning

- Would like highly expressive description language
- Also would like efficient implementation of inference algorithms which have acceptable run times on realistic inputs coming from applications
- Algorithms should be sound (only make valid inferences), complete (should make all valid inferences) and terminating on all inputs

Trade-offs in terminological reasoning

- Would like highly expressive description language
- Also would like efficient implementation of inference algorithms which have acceptable run times on realistic inputs coming from applications
- Algorithms should be sound (only make valid inferences), complete (should make all valid inferences) and terminating on all inputs

Theorem (Turing 1936)

There is no reasoning algorithm for FOL (subsumption) which is sound, complete and terminating, even with one binary predicate symbol.

Trade-offs in terminological reasoning

- Would like highly expressive description language
- Also would like efficient implementation of inference algorithms which have acceptable run times on realistic inputs coming from applications
- Algorithms should be sound (only make valid inferences), complete (should make all valid inferences) and terminating on all inputs

Theorem (Turing 1936)

There is no reasoning algorithm for FOL (subsumption) which is sound, complete and terminating, even with one binary predicate symbol.

Theorem (Bonatti 2003)

Neither for SHOIQ with terminological cycles.

Practical TBox reasoning

Theorem (Cook-Karp-Levin 1970s)

There is a reasoning algorithm for propositional logic which is sound, complete and terminating in polynomial time for all inputs if and only if there are such algorithms for thousands of other problems, such as colourability, bin packing, travelling salesperson . . .

- It is conjectured that there are no such algorithms

Practical TBox reasoning

Theorem (Cook-Karp-Levin 1970s)

There is a reasoning algorithm for propositional logic which is sound, complete and terminating in polynomial time for all inputs if and only if there are such algorithms for thousands of other problems, such as colourability, bin packing, travelling salesperson . . .

- It is conjectured that there are no such algorithms
- Reasoning algorithms for description logic inference algorithms which are sound and complete typically take exponential time in the worst case (between polynomial space and nondeterministic exponential time)
- ML suggested use of (optimized) tableau algorithms (Horrocks 1997)
- (Haarslev and Möller 2001) give examples of practical success

Structural subsumption algorithms

Let us first work with only conjunction $C \sqcap D$ and value restriction $\forall r.C$

- Every description is satisfiable, so we look at computing subsumption $C \sqsubseteq D$

Structural subsumption algorithms

Let us first work with only conjunction $C \sqcap D$ and value restriction $\forall r.C$

- Every description is satisfiable, so we look at computing subsumption $C \sqsubseteq D$
- Convert formula to a structural subsumption normal form:
  $C \equiv P_1 \sqcap \ldots \sqcap P_m \sqcap \forall r_1.C_1 \sqcap \ldots \sqcap \forall r_n.C_n$
  where the P_i are distinct, the r_j are distinct and the C_j are recursively in normal form

Structural subsumption algorithms

Let us first work with only conjunction $C \sqcap D$ and value restriction $\forall r.C$

- Every description is satisfiable, so we look at computing subsumption $C \sqsubseteq D$
- Convert formula to a structural subsumption normal form: $C \equiv P_1 \sqcap \ldots \sqcap P_m \sqcap \forall r_1.C_1 \sqcap \ldots \sqcap \forall r_n.C_n$ where the P_i are distinct, the r_j are distinct and the C_j are recursively in normal form
- Proof: Use associativity, commutativity and idempotence of $\sqcap$ and convert $\forall r.C_1 \sqcap \forall r.C_2$ to $\forall r.(C_1 \sqcap C_2)$

Structural subsumption algorithms

Let us first work with only conjunction $C \sqcap D$ and value restriction $\forall r.C$

- Every description is satisfiable, so we look at computing subsumption $C \sqsubseteq D$
- Convert formula to a structural subsumption normal form: $C \equiv P_1 \sqcap \ldots \sqcap P_m \sqcap \forall r_1.C_1 \sqcap \ldots \sqcap \forall r_n.C_n$ where the P_i are distinct, the r_j are distinct and the C_j are recursively in normal form
- Proof: Use associativity, commutativity and idempotence of $\sqcap$ and convert $\forall r.C_1 \sqcap \forall r.C_2$ to $\forall r.(C_1 \sqcap C_2)$
- Let $D \equiv Q_1 \sqcap \ldots \sqcap Q_k \sqcap \forall s_1.D_1 \sqcap \ldots \sqcap \forall s_l.D_l$
- To check $C \sqsubseteq D$: every P_i equals some Q_j, and every r_i equals some s_j, with $C_i \sqsubseteq D_j$
- Distinctness of roles means at most one recursive call per C_i, so polynomial time

Allowing bottom

Now we allow $\bot$, so satisfiability is not trivial

- In the definition of normal forms, we allow $\bot$ as a normal form
- If any of the P_i is $\bot$, the whole conjunction is rewritten to $\bot$
- Checking subsumption: observe that $\bot$ is subsumed by any description

Allowing negated atoms

Now we allow negated atomic concepts $\neg P$

- Treat them as concept names, except that when P and $\neg P$ occur as conjuncts they rewrite to $\bot$
- $\forall r.\neg P \sqcap P \sqcap \forall r.(P \sqcap \forall r.Q)$
  $\to P \sqcap \forall r.(\neg P \sqcap P \sqcap \forall r.Q)$
  $\to P \sqcap \forall r.(\bot \sqcap \forall r.Q)$
  $\to P \sqcap \forall r.\bot$

Allowing number restrictions

Now let us allow number restrictions

- They may be true, e.g. $\geq 10r \sqsubseteq \geq 5r$
- They may conflict, e.g. $\geq 2r \sqcap \leq 1r$
- They may conflict with value restrictions, e.g. $\geq nr \sqcap \forall r.\bot$
- Again we rewrite conflicts to $\bot$ and proceed to normalize

Allowing TBoxes

Now we allow TBoxes, first acyclic:

- Using associativity, commutativity and idempotence of $\sqcap$ and converting $\forall r.(C_1 \sqcap C_2)$ to $\forall r.C_1 \sqcap \forall r.C_2$, we get concept-centred normal form:
  Conjunctions of $\forall r_1. \ldots . \forall r_n.P$ for $n \geq 0$, with distinct r_i

Allowing TBoxes

Now we allow TBoxes, first acyclic:

- Using associativity, commutativity and idempotence of $\sqcap$ and converting $\forall r.(C_1 \sqcap C_2)$ to $\forall r.C_1 \sqcap \forall r.C_2$, we get concept-centred normal form:
  Conjunctions of $\forall r_1.\ldots.\forall r_n.P$ for $n \geq 0$, with distinct r_i
- More generally $\forall L.P$, where L is a finite set of words over the roles in T (conventionally $\forall\emptyset.P = T$)
- L is of polynomial size
- With $C \equiv \forall L_1.P_1 \sqcap \ldots \forall L_k.P_k$ and $D \equiv \forall L'_1.P_1 \sqcap \ldots \forall L'_k.P_k$, $C \sqsubseteq D$ iff $L_i \subseteq L'_i$, for $i = 1, k$
- Each inclusion can be checked in polynomial time and k is polynomial in the input descriptions

Allowing cyclic TBoxes

Now we allow cyclic TBoxes

- Represent the normal forms using finite automata over the alphabet of role names (the transitions are labelled by words) in T
- $D \equiv \forall r.D \sqcap \forall s.C, \quad B \equiv \forall r.\forall s.C, \quad C \equiv \forall s.C \sqcap P$
- The language of paths from D to P in the automaton represents all value restrictions to be satisfied by instances of concept D
- Hence subsumption of cyclic TBoxes, with greatest fixed point solutions, reduces to language inclusion of finite automata, which can be done using a polynomial space algorithm
- There is also a polynomial space algorithm for cyclic TBoxes with least fixed point solutions

Allowing top and existential restriction

Instead of AL we can work with conjunction $C \sqcap D$, top concept $\top$ and existential restrictions $\exists r.C$

- Normal form: $C \equiv P_1 \sqcap \ldots \sqcap P_m \sqcap \exists r_1.B_1 \sqcap \ldots \sqcap \exists r_l.B_l$, where P_i and r_j are distinct and B_j are recursively in normal form
- Description graph G_T: Node C labelled with $P_1, \ldots, P_m$, r_j-labelled edges to nodes B_j
- Checking $C \sqsubseteq D$: find a simulation from G_T to G_T relating (D, C)
- With greatest fixed point solutions of cyclic TBoxes, simulations can be computed in polynomial time
- Also polynomial time for least fixed point soultions

Can allow bottom concept $\bot$, nominals and GCI retaining polynomial time

Conclusion

- Description logics provide a wide set of features which allow repesentation of diverse situations found in applications
- Algorithms have been developed for a rich set of reasoning problems
- Other languages like OWL have been built on top of description logics in the web setting
- Many description logics have low processing complexity which allows successful development of software tools using them
- There are restricted description logics which have efficient reasoning algorithms
- Trade-offs between adequate expressiveness and fast implementations