Knowledge representation and Reasoning

Robert Hoehndorf

Knowledge, Representation and Reasoning

Object-oriented representation of knowledge

- So far, most knowledge about some x is scattered across the KB
- Need to find a way to *organize* axioms better
- We see *objects* and *kinds* of objects in the world: use these as structuring principles
  - toys have a color, shape, weight, etc.
  - a class (situation) has: a room, teacher, day, time, seating arrangement, etc.

Frames

- Let's call the object structures *frames*
- Two types of frames:
  - Individual frame: represents single objects, like a single person
  - generic frame: represents categories of objects
- An individual frame is a named list of buckets called slots
- What goes in the bucket is called a filler of the slot
  - Individual frames: `jeddah`
  - slot names: `:Population` (with ":" at start)
  - generical frames: `SaudiCity`

Instances and specialization

- Individual frames have a special slot called `:INSTANCE-OF`, filled by a generic frame:

```
(jeddah
  <:INSTANCE-OF SaudiCity>
  <:Province makkah>
  <:Population 2.5M>
 ...
)
```

- Generic frames have a similar slot called `:IS-A`, filled by another generic frame:

```
(SaudiCity
  <:IS-A City>
  <:Province SaudiProvince>
  <:Country saudiArabia>
...
)
```

- We say that Jeddah is *an instance* of the frame SaudiCity, and that SaudiCity is *a specialization of* the frame City.

IS-A and inheritance

- Specialization relation implies that fillers of more general frames also apply for more specific ones
- Same holds for *procedures* (used to calculate/update slot fillers)

Reasoning with frames

- Basic reasoning: user instantiates a frame, i.e., declares that an object exists
- slot fillers are inherited, if possible
- slot values are calculated using procedures
- Examples: see book
- Here, we will not consider frame-based systems further

Description Logics

- used to formalize the terminology of an application domain
  - define important notions (concepts, relations) in the domain
  - state constraints in a way that these notions can be interpreted
  - infer consequences of these constraints

Following slides based on tuturial by Franz Baader (`http://lat.inf.tu-dresden.de/~baader/Talks/`)

Description Logics

- Derived from semantic networks and frames
- initially, used incomplete systems
- later, developed tableau-based systems, complexity results, first implementations
- then: highly expressive DLs, highly optimized tableau-based systems, relation to modal logics and (fragments of) FOL

Description Logics: overview

- TBox: defines the terminology of the domain
- ABox: states facts (assertions) about the world
- Reasoning: derive implicitly represented knowledge (e.g., subsumption)

Description Logic: ALC

Definition

Let N_C be a set of concept names and N_R be a set of relation names, $N_C \cap N_R = \emptyset$. $\mathcal{ALC}$ concept descriptions are inductively defined as:

- If $A \in N_C$, then A is an $\mathcal{ALC}$ concept description
- If C, D are $\mathcal{ALC}$ concept description, and $r \in N_R$, then the following are $\mathcal{ALC}$ concept descriptions:
  - $C \sqcap D$
  - $C \sqcup D$
  - $\neg C$
  - $\forall r.C$
  - $\exists r.C$

- We use $\bot$ as abbreviation of $A \sqcap \neg A$, $\top$ as abbreviation of $A \sqcup \neg A$

Description Logic: ALC

Definition

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty domain $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$:

- $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ for all $A \in N_C$,
- $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ for all $r \in N_R$

The interpretation function is extended to $\mathcal{ALC}$ concept descriptions as follows:

- $(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} := C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} := \Delta^{\mathcal{I}} - C^{\mathcal{I}}$
- $(\forall r.C)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} | \text{for all } e \in \Delta^{\mathcal{I}} : (d, e) \in r^{\mathcal{I}} \text{ implies } e \in C^{\mathcal{I}}\}$
- $(\exists r.C)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} | \text{there is } e \in \Delta^{\mathcal{I}} : (d, e) \in r^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}}\}$

Description Logic: ALC

- $\mathcal{ALC}$ can be seen as a fragment of FOL:
  - concept names are unary predicates, role names binary predicates
  - concept descriptions are formulas with one free variable
- the formulas resulting from transformation to FOL are known to be decidable (two-variable fragment)

Description Logic: extensions of ALC

- Description Logics are a family of logics
- Many different constructors
- Number restrictions: $\leq nr.C$, $\geq nr.C$ with semantics: $(\geq nr.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} | card(\{e | (d,e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}) \geq n\}$ etc
- More later!

Description Logic: terminologies

- A concept definition is of the form $A \equiv C$ where
  - A is a concept name
  - C is a concept description
- A TBox is a finite set of concept definitions such that it
  - does not contain multiple definitions,
  - does not contain cyclic definitions
- A *defined concept* occurs on the left-hand side of a definition
- A *primitive concept* does not occur on the left-hand side of a definition
- An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if it satisfies all its concept definitions: $A^{\mathcal{I}} = C^{\mathcal{I}}$ for all $A \equiv C \in \mathcal{T}$

Description Logic: terminologies

- Also possible to use more general constraints for the definition of concepts
- A *generalized concept inclusion* (GCI) is of the form $C \sqsubseteq D$ where C, D may be concept descriptions
- An interpretation $\mathcal{I}$ is a model of a set of GCIs $\mathcal{T}$ (a general TBox) if it satisfies all its concept inclusion axioms: $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all $C \sqsubseteq D \in \mathcal{T}$

Description Logic: assertions

- An assertion is of the form $C(a)$ (concept assertion) or $r(a, b)$ (role assertion), where C is a concept description, r is a role, a, b are individual names from a set N_I of such names
- An ABox is a finite set of assertions
- An interpretation $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ if it satisfies all its assertions:
  - $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all $C(a) \in \mathcal{A}$
  - $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$ for all $r(a, b) \in \mathcal{A}$

Description Logic: Reasoning

- Subsumption: Is C a subconcept of D?
  - $C \sqsubseteq_{\mathcal{T}} D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{T}$
- Satisfiability: Is the concept C non-contradictory?
  - C is satisfiable w.r.t. $\mathcal{T}$ iff $C^{\mathcal{I}} \neq \emptyset$ for some model $\mathcal{I}$ of $\mathcal{T}$
- Consistency: Is the ABox $\mathcal{A}$ non-contradictory?
  - $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$ iff it has a model that is also a model of $\mathcal{T}$
- Instantiation: Is e an instance of C?
  - $\mathcal{A} \models_{\mathcal{T}} C(e)$ iff $e^{\mathcal{I}} \in C^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$.

Description Logic: Reasoning

- Reducing subsumption to satisfiability: $C \sqsubseteq_{\mathcal{T}} D$ iff $C \sqcap \neg D$ is unsatisfiable w.r.t. $\mathcal{T}$
- Reducing satisfiability to subsumption: C is satisfiable w.r.t. $\mathcal{T}$ iff not $C \sqsubseteq_{\mathcal{T}} \bot$
- Reducing satisfiability to consistency: C is satisfiable w.r.t. $\mathcal{T}$ iff $\{C(a)\}$ is consistent w.r.t. $\mathcal{T}$
- Reducing instantiation to consistency: a is an instance of C w.r.t. $\mathcal{T}$ and $\mathcal{A}$ iff $\mathcal{A} \cup \{\neg C(a)\}$ is inconsistent w.r.t. $\mathcal{T}$
- Reducing consistency to instantiation: $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$ iff a is not an instance of $\bot$ w.r.t. $\mathcal{T}$ and $\mathcal{A}$

Description Logic: Reasoning

Definition

For a given TBox $\mathcal{T}$ and concept description C, the expansion $C^{\mathcal{T}}$ of C w.r.t. $\mathcal{T}$ if obtained from C by

- replacing defined concept by their definition,
- until no more defined concepts occur.

Description Logic: Reasoning

- TBox acyclic, therefore expansion always terminates
- Size of the expanded concept?
- C is satisfiable w.r.t. $\mathcal{T}$ iff $C^{\mathcal{T}}$ is satisfiable w.r.t. the empty TBox $\emptyset$
- $C \sqsubseteq_{\mathcal{T}} D$ iff $C^{\mathcal{T}} \sqsubseteq_{\emptyset} D^{\mathcal{T}}$
- same for consistency and instantiation

ALC reasoning

- We are looking for a *decision procedure* for reasoning in $\mathcal{ALC}$:
  - sound: positive answers are correct
  - complete: generate all answers, negative answers are correct
  - termination: answer in finite time
- efficient: best possible (worst-case) complexity
- Remember: FOL has no decision procedure

ALC reasoning

- Sufficient to decide consistency of ABox without TBox
  - TBox can be eliminated by concept expansion
  - all reasoning problems can be reduced to consistency
- a tableau-based consistency algorithm tries to generate a finite model for the input ABox $\mathcal{A}_0$
  - applies tableau rules to extend the ABox
  - checks for obvious contradictions
  - an ABox that is complete (no more rules can be applied) and open (no contradictions found) describes a model

ALC reasoning

$\mathcal{T}$: $GoodStudent \equiv Smart \sqcap Studious$

- Subsumption question:
  $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqsubseteq_?$
  $\exists attendedBy.GoodStudent$

ALC reasoning

$\mathcal{T}$: $GoodStudent \equiv Smart \sqcap Studious$

- Subsumption question: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqsubseteq_{?} \exists attendedBy.GoodStudent$
- Reduction to satisfiability: is the following concept unsatisfiable w.r.t. $\mathcal{T}$: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg \exists attendedBy.GoodStudent$

ALC reasoning

$\mathcal{T}$: $GoodStudent \equiv Smart \sqcap Studious$

- Subsumption question: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqsubseteq_? \exists attendedBy.GoodStudent$
- Reduction to satisfiability: is the following concept unsatisfiable w.r.t. $\mathcal{T}$: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.GoodStudent$
- Reduction to consistency: is the following ABox inconsistent w.r.t. $\mathcal{T}$: $\{(\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.GoodStudent)(a)\}$

ALC reasoning

$\mathcal{T}$: $GoodStudent \equiv Smart \sqcap Studious$

- Subsumption question: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqsubseteq_? \exists attendedBy.GoodStudent$
- Reduction to satisfiability: is the following concept unsatisfiable w.r.t. $\mathcal{T}$: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.GoodStudent$
- Reduction to consistency: is the following ABox inconsistent w.r.t. $\mathcal{T}$: $\{(\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.GoodStudent)(a)\}$
- Expansion: is the following ABox inconsistent: $\{(\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.(Smart \sqcap Studious))(a)\}$

ALC reasoning

$\mathcal{T}$: $GoodStudent \equiv Smart \sqcap Studious$

- Subsumption question: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqsubseteq_? \exists attendedBy.GoodStudent$
- Reduction to satisfiability: is the following concept unsatisfiable w.r.t. $\mathcal{T}$: $\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.GoodStudent$
- Reduction to consistency: is the following ABox inconsistent w.r.t. $\mathcal{T}$: $\{(\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.GoodStudent)(a)\}$
- Expansion: is the following ABox inconsistent: $\{(\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \neg\exists attendedBy.(Smart \sqcap Studious))(a)\}$
- Negation Normal Form: $\{(\exists attendedBy.Smart \sqcap \exists attendedBy.Studious \sqcap \forall attendedBy.(\neg Smart \sqcup \neg Studious))(a)\}$

Tableau algorithm

- Input: $\mathcal{ALC}$ ABox $\mathcal{A}_0$
- Output: "yes" if $\mathcal{A}_0$ is consistent, "no" otherwise
- Preprocessing: transform all concept descriptions in $\mathcal{A}_0$ in Negation Normal Form with the following rules:
  - $\neg(C \sqcap D) \Rightarrow \neg C \sqcup \neg D$
  - $\neg(C \sqcup D) \Rightarrow \neg C \sqcap \neg D$
  - $\neg\neg C \Rightarrow C$
  - $\neg(\exists r.C) \Rightarrow \forall r.\neg C$
  - $\neg(\forall r.C) \Rightarrow \exists r.\neg C$
- NNF can be computed in polynomial time, does not change the semantics of the concept

Tableau algorithm

- Datastructure: a set of ABoxes, start with $\{\mathcal{A}_0\}$
- Rule application: take one ABox from the set, replace with finite number of new ABoxes
- Termination: if no more rules apply to any of the ABoxes in the set
- complete ABox: no rule applies to it
- Answer "yes" if the set contains an open ABox, i.e., an ABox not containing a contradiction of the form $C(a)$ and $\neg C(a)$ for some individual name a
- Answer "no" if all ABoxes in the set are closed

Tableau rules

- The $\sqcap$-rule:
  - Condition: $\mathcal{A}$ contains $(C \sqcap D)(a)$ but not both $C(a)$ and $D(a)$
  - Action: $\mathcal{A}' = \mathcal{A} \cup \{C(a), D(a)\}$
- The $\sqcup$-rule:
  - Condition: $\mathcal{A}$ contains $(C \sqcup D)(a)$ but neither $C(a)$ nor $D(a)$
  - Action: $\mathcal{A}' = \mathcal{A} \cup \{C(a)\}$ and $\mathcal{A}'' = \mathcal{A} \cup \{D(a)\}$
- The $\exists$-rule:
  - Condition: $\mathcal{A}$ contains $(\exists r.C)(a)$ but there is no c with $\{r(a,c), C(c)\} \subseteq \mathcal{A}$
  - Action: $\mathcal{A}' = \mathcal{A} \cup \{r(a,b), C(b)\}$ where b is a new individual name
- The $\forall$-rule:
  - Condition: $\mathcal{A}$ contains $(\forall r.C)(a)$ and $r(a,b)$ but not $C(b)$
  - Action: $\mathcal{A}' = \mathcal{A} \cup \{C(b)\}$

Tableau algorithm

- Local correctness (rules preserve consistency)
- Termination (no infinite paths)
- Soundness (any complete and open ABox has a model)
- Completeness (closed ABoxes have no model)

Tableau algorithm: summary

- Starting with a finite ABox $\mathcal{A}_0$ in NNF, the algorithm always terminates with a finite set of complete ABoxes $\mathcal{A}_1, ..., \mathcal{A}_n$
- Local correctness: $\mathcal{A}_0$ is consistent iff one of $\mathcal{A}_1, ..., \mathcal{A}_n$ is consistent
- Answer "no": none of $\mathcal{A}_1, ..., \mathcal{A}_n$ is open, $\mathcal{A}_1, ..., \mathcal{A}_n$ are inconsistent $\Rightarrow$ $\mathcal{A}_0$ is inconsistent
- Answer "yes": one of $\mathcal{A}_1, ..., \mathcal{A}_n$ is open, therefore one of $\mathcal{A}_1, ..., \mathcal{A}_n$ consistent $\Rightarrow$ $\mathcal{A}_0$ is consistent

Adding number restrictions

Number restrictions $\leq nr.C$, $\geq nr.C$ with semantics:

- $(\geq nr.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} | card(\{e | (d,e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}) \geq n\}$
- $(\leq nr.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} | card(\{e | (d,e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}) \leq n\}$

NNF:

- $\neg(\geq (n+1)r.C) \Rightarrow (\leq nr.C)$
- $\neg(\geq 0r.C) \Rightarrow \perp$
- $\neg(\leq nr.C) \Rightarrow (\geq (n+1)r.C)$

New assertions:

- Inequality assertions of the form $x \neq y$ with semantics $x^{\mathcal{I}} \neq y^{\mathcal{I}}$ (symmetric)

Adding number restrictions

- The $\geq$ rule:
  - Condition: $\mathcal{A}$ contains $(\geq nr.C)(a)$ but there are no $c_1, ..., c_n$ with $\{r(a, c_1), C(c_1), ..., r(a, c_n), C(c_n)\} \cup \{c_i \neq c_j | 1 \leq i \leq n, 1 \leq j \leq n, i \neq j\} \subseteq \mathcal{A}$
  - Action: $\mathcal{A}' = \mathcal{A} \cup \{r(a, b_1), C(b_1), ..., r(a, b_n), C(b_n)\} \cup \{b_i \neq b_j | 1 \leq i \leq n, 1 \leq j \leq n, i \neq j\}$ where $b_1, ..., b_n$ are new individual names.
- The $\leq$ rule:
  - Condition: $\mathcal{A}$ contains $(\leq nr.C)(a)$, and there are $b_1, ..., b_{n+1}$ with $\{r(a, b_1), C(b_1), ..., r(a, b_n), C(b_n), r(a, b_{n+1}), C(b_{n+1})\} \subseteq \mathcal{A}$ but $\{b_i \neq b_j | 1 \leq i \leq n, 1 \leq j \leq n, i \neq j\} \not\subseteq \mathcal{A}$
  - for all $i \neq j$ with $b_i \neq b_j \notin \mathcal{A}$, $\mathcal{A}_{i,j} = \mathcal{A}[b_i/b_j]$ (replace b_i with b_j)

Adding number restrictions

- New contradictions:
  - $\mathcal{A}$ contains $a \neq a$ for some individual a
  - $\mathcal{A}$ contains $(\leq nr.C)(a)$, and there are $b_1, ..., b_{n+1}$ with $\{r(a, b_1), C(b_1), ..., r(a, b_{n+1}), C(b_{n+1})\} \subseteq \mathcal{A}$ and $\{b_i \neq b_j | i \neq j\} \subseteq \mathcal{A}$

Adding number restrictions

Is this a decision procedure?

- local correctness: rules preserve consistency

Adding number restrictions

Is this a decision procedure?

- local correctness: rules preserve consistency
- completeness: a closed ABox does not have a model

Adding number restrictions

Is this a decision procedure?

- local correctness: rules preserve consistency
- completeness: a closed ABox does not have a model
- soundness: a complete and open ABox has a model (no!)

Adding number restrictions

Is this a decision procedure?

- local correctness: rules preserve consistency
- completeness: a closed ABox does not have a model
- soundness: a complete and open ABox has a model (no!)
- termination: there is no infinite chain of rule applications (no!)

Adding number restrictions

- The choose rule:
  - Condition: $\mathcal{A}$ contains $(\leq nr.C)(a)$ and $r(a,b)$ but neither $C(b)$ nor $\overline{C(b)}$ ($\overline{C(b)}$ is NNF of $\neg C(b)$)
  - Action: $\mathcal{A}' = \mathcal{A} \cup \{C(b)\}$ and $\mathcal{A}'' = \mathcal{A} \cup \{\overline{C(b)}\}$

Adding number restrictions

- Apply generating rules ($\exists$, $\geq$) with lower priority

Adding GCIs

- A finite set of GCIs can be encoded in one GCI of the form $\top \sqsubseteq C$:
  - $\{C_1 \sqsubseteq D_1, ..., C_n \sqsubseteq D_n\} \Rightarrow \{\top \sqsubseteq (\neg C_1 \sqcup D_1) \sqcap ... \sqcap (\neg C_n \sqcup D_n)\}$
- Given a GCI of the form $\top \sqsubseteq C$ where C is in NNF:
  - Condition: $\mathcal{A}$ contains individual name a but not $C(a)$
  - Action: $\mathcal{A}' = \mathcal{A} \cup \{C(a)\}$

Adding GCIs

- Termination does not hold: test consistency of $\{P(a)\}$ with $\top \sqsubseteq \exists r.P$
- Solution: blocking
  - y is blocked by x iff $label(y) \subseteq label(x)$
  - generating rules are not applied to blocked individuals
  - to avoid cyclic blocking, fix an enumeration of individual names, and add to the blocking condition that y comes after x in the enumeration

Adding GCIs

- soundness must be reconsidered
  - because of blocking, ABox can be complete, although a generating rule applies
  - modication of canonical interpretation: the *r*-successors of a blocked individual are the *r*-successors of the least individual (in the enumeration) blocking it

Complexity of ALC reasoning

Concept satisfiability and ABox consistency for $\mathcal{ALC}$ are PSPACE-complete.

Proof (by reduction to Quantified Boolean Formula problem): Attributive concept descriptions with complements. Manfred Schmidt-Schauss and Gert Smolka. Artificial Intelligence, 48(1):1-26, 1991.

Extensions of ALC(Q)

Description Logic complexity navigator:
`http://www.cs.man.ac.uk/~ezolin/dl/`

- Role constructors: inverse, intersection, negation, union, role chains,...
- Role axioms: transitivity, reflexivity, role inclusions
- Concept constructors: functionality, nominals