Logic-based Ontology Engineering

Henrik Tscherny

18. Juli 2022

Inhaltsverzeichnis

1 Description Logic Attribute Language with Concepts (ALC)

Description logics (DLs) are a *decidable* subset of first-order logic. They are restricted to *unary and binary predicates* .

1.1 Vocabulary

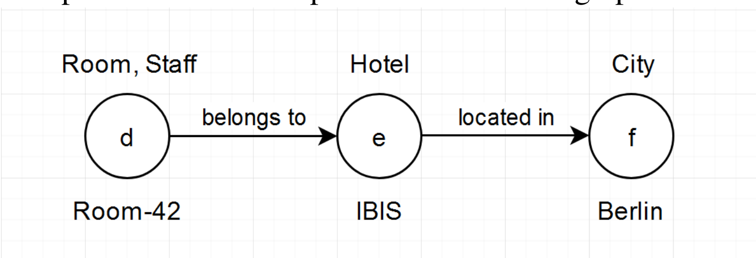
- C: Concepts/Classes/Categories $\rightarrow$ Person(x) is the set of all elements which are a person
  - Concepts can be defined inductively, if C and D are concepts then:
Name:	top	bottom	conjunction	disjunction	negation
Syntax:	$\top$	$\bot$	$C \sqcap D$	$C \sqcup D$	$\neg C$
Semantics:	$\Delta^{\mathcal{I}}$	$\emptyset$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - Example: Plant $\sqcap$ Tree $\rightarrow$ set of objects being a plant and a tree
  - if C is a concepts and r is a role the following are also concepts (describes *outgoing role connections*):
Name:	existential restriction	value restriction
Syntax:	$\exists r.C$	$\forall r.C$
Semantics:	$\{d \mid \exists e.(d,w) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}$	$\{d \mid \forall e.(d,e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$
  - Example: $\exists\, isSpicy.Food \;=\; \exists y.(isSpicy(x,y) \wedge Food(y))$
    $\rightarrow$ spicy food exists
  - Example 2: $\forall\, isPlant.Tree \;=\; \forall y.(isPlant(x,y) \rightarrow Tree(y))$
    $\rightarrow$ all trees are plants
  - *size* of a concept:
    * size(A) = size($\top$) = size($\bot$) = 1
    * size($C \sqcap D$) = size($C \sqcup D$) = 1 + size(C) + size(D)
    * size($\neg C$) = size($\exists r.C$) = size($\forall r.C$) = 1 + size(C)
      $\rightarrow$ **the number of $\exists$, $\forall$, $\neg$, $\sqcap$, $\sqcup$, $\top$, $\bot$ + 'basic' concepts**
    * Example: size($\exists r.(\exists s.A \sqcap \exists r.\exists s.\top)$) = 7
  - *role depth* of a concept:
    * rd(A) = rd($\top$) = rd($\bot$) = 0
    * rd($\neg C$) = rd(C)
    * rd($C \sqcap D$) = rd($C \sqcup D$) = $max\{rd(C), rd(D)\}$
    * rd($\exists r.C$) = rd($\forall r.C$) = $1 + rd(C)$
      $\rightarrow$ **maximal nesting depth of role restrictions in the concept**
    * Example: rd($\exists r.(\exists s.A \sqcap \exists r.\exists s.\top)$) = 3
  - if $C \sqsubseteq_{\mathcal{O}} D$ then C is more specific than D, D is more general than C

- R: Roles/Relations/Properties/Attributes $\rightarrow$ likes(x,y) is the set of all pairs, where x likes y
  - only unary and binary relations $\rightarrow$ n-ary relations can be transformed into unary and binary relations with reification
- I: Objects/Individuals $\rightarrow$ represents a constant in first-order logic, like a TOM, a specific person

1.2 Interpretations

A DL *interpretation* is a tuple $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where:

- $\Delta^{\mathcal{I}}$ is the **domain** of $\mathcal{I}$ ($\Delta^{\mathcal{I}} \neq \emptyset$)
- $\cdot^{\mathcal{I}}$ is the **interpretation function**
  - each $A \in C$ is interpreted as a set $A^{\mathbb{I}} \subseteq \Delta^{\mathcal{I}}$
  - each $r \in R$ is interpreted as a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
  - each $a \in I$ is interpreted a an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
- Interpretations can be represented as labeled graphs like:



- axioms can be used to restrict the set of interpretations
- every interpretation is a model of $\emptyset$

1.3 Axioms

- *general concept inclusion (CGI)*
  - Syntax: $C \sqsubseteq D$
  - Semantics: $\mathcal{I} \models C \sqsubseteq D \Leftrightarrow C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - Lecture $\sqsubseteq$ Course
- *equivalence axiom*

- $C \equiv D \Leftrightarrow C \sqsubseteq D \sqcap D \sqsubseteq C$

- *concept definitions*
  - $A \equiv D$, where A is a concept name
  - Example: Tutor $\equiv$ Person

- *Assertions/Facts*
Name:	concept assertion	role assertion
Syntax:	$a : C$	$(a, b) : r$
Semantics:	$a^{\mathcal{I}} \in C^{\mathcal{I}}$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$
  - Example concept assertion: Room-42:Room
  - Example role assertion: (Room-42, IBIS):belongsTo

- *closure axiom*
  - define the range/scope of a relation
  - setting the scope of a description under the open world assumption
  - Example: Cow $\sqsubseteq \forall$ eats.(Grass $\sqcup$ Grain)

- *covering axiom*
  - (partial) definition with a disjunction on the right
  - can be used to cover not explicitly stated informations
  - Example: Person $\equiv$ Man $\sqcup$ Woman $\sqcup$ Diverse

- *disjointness axiom*
  - can be used to separate to concepts from each other
  - Example: Animal $\sqsubseteq \neg$ Plant

1.4 Ontologies

An ontology is a set $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, where:

- ABox $\mathcal{A}$, finite set of assertion/facs (ABox hold the 'data')

- TBox $\mathcal{T}$, finite set of general concept inclusions (TBox hold the 'knowledge')

- Example:

- $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$
- $\mathcal{A} = \{$Room-42:Room, (Room-42,IBIS) :belongsTo, (IBIS,Berlin) :locatedIn$\}$
- $\mathcal{T} = \{$Room$\sqsubseteq \neg$City,Hotel$\sqsubseteq \neg$Room$\}$
- $C(\mathcal{O}) = \{$Room,Hotel,City$\}$
  $R(\mathcal{O}) = \{$belongsTo,locatedIn$\}$
  $I(\mathcal{O}) = \{$Room-42,IBIS,Berlin$\}$

- an ontology is **inconsistent** iff $\mathcal{O} \models \top \sqsubseteq \bot$
- if a ontology is consistent the ABox is irrelevant for checking satisfiability and subsumption
- a TBox $\mathcal{T}$ is called acyclic if:
  - each concept name A has at most one definition $A \equiv C_A \in \mathcal{T}$
  - the *depends on* relation is acyclic $\rightarrow$ A *depends on* B if B occurs in the definition C_A of A in $\mathcal{T}$
- the expansion of C wrt. to an acyclic TBox $\mathcal{T}$ is to exhaustively replace all defined concept names in C by their definitions

1.5 Reasoning

Reasoning allows to entail information from an ontology

- reasoning for acyclic ALC TBoxes is in PSpace, general ALC reasoning is in ExpTime
- $\mathcal{O}$ entails an axiom $\alpha(\mathcal{O} \models \alpha)$ if every model of $\mathcal{O}$ is also a model of α
C is subsumed by D	$\mathcal{O} \models C \sqsubseteq D$	$C \sqsubseteq_{\mathcal{O}} D$
C is equivalent to D	$\mathcal{O} \models C \equiv D$	$C \equiv_{\mathcal{O}} D$
C is strictly subsumed by D	$C \sqsubseteq_{\mathcal{O}} \wedge C \not\equiv_{\mathcal{O}} D$	$C \sqsubset_{\mathcal{O}} D$
C and D are disjoint	$\mathcal{O} \models C \sqcap D \sqsubseteq \bot$	$-$

    (everything wrt. to $\mathcal{O}$)
- $\mathcal{O} \models a : C$, a is an instance of C
- $\mathcal{O} \not\models C \sqsubseteq \bot$, C is satisfiable
- If all concept names in $\mathcal{O}$ are satisfiable, then $\mathcal{O}$ in coherent

- classification is the task of computing all entailments of the form $\mathcal{O} \models A \sqsubseteq B,\ A, B \in C$

- materialization is the task of computing all entailments of the form $\mathcal{O} \models a : A$ and $\mathcal{O} \models (a, b) : r$, with $(a, b) \in I, A \in C, r \in R$

- A tautology is an axiom that is satisfied by *all* interpretations
  - if α is a tautology, then α is entailed by the empty ontology $\emptyset$ and also entailed by all $\mathcal{ALC}$ ontologies
  - the following are tautologies

$$
\begin{array}{llll}
C \sqcap D \equiv D \sqcap C & (C \sqcap D) \sqcap E \equiv C \sqcap (D \sqcap E) & C \sqcap C \equiv C & C \sqcap \top \equiv C \\
C \sqcup D \equiv D \sqcup C & (C \sqcup D) \sqcup E \equiv C \sqcup (D \sqcup E) & C \sqcup C \equiv C & C \sqcup \bot \equiv C
\end{array}
$$

$$
\begin{array}{cccc}
\neg\neg C \equiv C & \neg(C \sqcap D) \equiv \neg C \sqcup \neg D & & \neg(C \sqcup D) \equiv \neg C \sqcap \neg D \\
\neg\top \equiv \bot & \neg\bot \equiv \top & \neg\exists r.C \equiv \forall r.\neg C & \neg\forall r.C \equiv \exists r.\neg C
\end{array}
$$

$$
C \sqcap D \sqsubseteq C \qquad C \sqsubseteq \top \qquad C \sqsubseteq C \sqcup D \qquad \bot \sqsubseteq C
$$

$$
(\exists r.C) \sqcap (\forall r.D) \sqsubseteq \exists r.(C \sqcap D)
$$

$$
\begin{array}{lll}
\forall r.(C \sqcap D) \equiv (\forall r.C) \sqcap (\forall r.D) & \forall r.\top \equiv \top & (\forall r.C) \sqcup (\forall r.D) \sqsubseteq \forall r.(C \sqcup D) \\
\exists r.(C \sqcup D) \equiv (\exists r.C) \sqcup (\exists r.D) & \exists r.\bot \equiv \bot & \exists r.(C \sqcap D) \sqsubseteq (\exists r.C) \sqcap (\exists r.D)
\end{array}
$$

- in negation normal form (NNF) negation ($\neg$) is only directory in front of concept names

- every concept can be expressed in NNF

- *entailment rules*:
  - $C \sqsubseteq_{\mathcal{O}} D \land D \sqsubseteq_{\mathcal{O}} E \rightarrow C \sqsubseteq_{\mathcal{O}} E$ (transitivity)
  - $C \sqsubseteq_{\mathcal{O}} D \rightarrow \exists r.C \sqsubseteq_{\mathcal{O}} \exists r.D, \forall r.D, C \sqcap E \sqsubseteq_{\mathcal{O}} D \sqcap E \land C \lor R \sqsubseteq_{\mathcal{O}} D \sqcup E$ (substitution)
  - $C \sqsubseteq_{\mathcal{O}} D \sqcap E \leftrightarrow C \sqsubseteq_{\mathcal{O}} D \land C \sqsubseteq_{\mathcal{O}} E$
  - $C \sqcup D \sqsubseteq_{\mathcal{O}} E \leftrightarrow C \sqsubseteq_{\mathcal{O}} E \land D \sqsubseteq_{\mathcal{O}} E$
  - $C \sqcap D \sqsubseteq_{\mathcal{O}} E \leftrightarrow C \sqsubseteq_{\mathcal{O}} \neg D \sqcup E$
  - $C \sqsubseteq_{\mathcal{O}} D \leftrightarrow \neg D \sqsubseteq_{\mathcal{O}} \neg C$ (contraposition)
  - $C \sqcap \neg C \sqsubseteq_{\emptyset} \bot$ (contradiction)

- *reasoning with ABoxes*

- $\mathcal{O} \models a : C \wedge C \sqsubseteq_{\mathcal{O}} D \rightarrow \mathcal{O} \models a : D$
- $\mathcal{O} \models a : \top$
- $\mathcal{O} \models a : \bot \rightarrow \top \sqsubseteq_{\mathcal{O}} \bot$ ($\mathcal{O}$ is inconsistent)
- $\mathcal{O} \models a : (C \sqcap D) \leftrightarrow \mathcal{O} \models a : C \wedge \mathcal{O} \models a : D$
- $\mathcal{O} \models a : C \vee \mathcal{O} \models a : D \rightarrow \mathcal{O} \models a : (C \sqcup D)$
- $\mathcal{O} \models a : C \wedge \mathcal{O}$ is consistent $\rightarrow \mathcal{O} \not\models a : \neg C$
- $\mathcal{O} \models (a, b) : r \wedge \mathcal{O} \models b : C \rightarrow \mathcal{O} \models a : (\exists r.C)$
- $\mathcal{O} \models (a, b) : r \wedge \mathcal{O} \models a : (\forall r.C) \rightarrow \mathcal{O} \models b : C$

(1-4) hold in both directions

1.6 Open World vs. Closed World

- in an open-world assumption, unknown $\neq$ false
  $\rightarrow$ you have to explicitly state that sth. is false
- in a closed-world assumption, unknown $=$ false
  $\rightarrow$ you have to explicitly state that sth. is true

2 SROIQ(D)

More expressive DL then ALC, that is still decidable (reasoning is 2-NExpTime-complete)
SROIQ(D) stands for:

- transitive roles (S)
- complex role axioms (R)
- nominals (O)

Name	nominal	local reflexivity
Syntax	$\{a\}$	∃r.Self
Semantics	$\{a^{\mathcal{I}}\}$	$\{x \mid (x, x) \in r^{\mathcal{I}}\}$
Example	∃employedBy.{TUDresden}	∃loves.Self

- Inverse roles (I)

Name	inverse role
Syntax	r^{-}
Semantics	$(r^{-})^{\mathcal{I}} = \{(d, e) \mid (e, d) \in r^{\mathcal{I}}\}$
Example	belongsTo^{-}

- qualified number restrictions (Q)

  -

Name	at-least restriction	at-most restriction
Syntax	$\geq nr.C$	$\leq nr.C$
Semantics	$\{d \mid \#\{e \in C^{\mathcal{I}} \mid (d,e) \in r^{\mathcal{I}}\} \geq n\}$	$\{d \mid \#\{e \in C^{\mathcal{I}} \mid (d,e) \in r^{\mathcal{I}}\} \leq n\}$
Example	Student $\sqcap$ ($\geq$ 2 attends.Lecture)	$\leq$ 1 belongsTo.$\top$

  (qualifiers e.g. $\leq$ and $\geq$ are optional and only numbers can be used)

- concrete domains (D)

2.1 Structure

- In SROIQ(D) an ontology consists of three parts $\mathcal{O} = \mathcal{A} \cup \mathcal{TR}$
- *Assertions*

  -

Name	equality	inequality	negated role assertion
Syntax	$a \approx b$	$a \not\approx b$	$(a,b) : \neg r$
Semantics	$a^{\mathcal{I}}$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin r^{\mathcal{I}}$
Example	DD $\approx$ Dresden	$DD \not\approx APB$	(Ernie, Bert): $\neg$ hasBrother

  - all assertions are only syntactic sugar:
    * $a : C \Leftrightarrow \{a\} \sqsubseteq C$
    * $a \approx b \Leftrightarrow \{a\} \sqsubseteq \{b\}$
    * $a \not\approx b \Leftrightarrow \{a\} \sqsubseteq \neg\{b\}$
    * $(a,b) : r \Leftrightarrow \{a\} \sqsubseteq \exists r.\{b\}$
    * $(a,b) : \neg r \Leftrightarrow \{a\} \sqsubseteq \forall r.\neg\{b\}$

- *Role Axioms* $\rightarrow$ RBox
- a RBox $\mathcal{R}$ is regular if there is a strict partial order $<$ on $R^{-}(\mathcal{O})$ s.d:
  - $r < s \Leftrightarrow r^{-} < s, \forall r, s \in R^{-}(\mathcal{O})$
  - every role inclusion is of the form:
    * $r \circ r \sqsubseteq r$
    * $r^{-} \sqsubseteq r$
    * $r_1 \circ \cdots \circ r_n \sqsubseteq r$
    * $r \circ r_1 \circ \cdots \circ r_n \sqsubseteq r$
    * $r_1 \circ \cdots \circ r_n \circ r \sqsubseteq r$

- role axioms:
  -

Name	*Syntax*	*Meaning*
Domain	$Dom(r) \sqsubseteq C$	$\exists r.\top \sqsubseteq C$
Range	$Ran(r) \sqsubseteq C$	$\exists r^{-}.\top \sqsubseteq C,\ \top \sqsubseteq \forall r.C$
Functionality	$Fun(r)$	$\top \sqsubseteq \leq 1r.\top$
Reflexivity	$Ref(r)$	$\top \sqsubseteq \exists r.Self$

  -

Name	role inclusion	complex role inclusion
Syntax	$r \sqsubseteq s$	$s_1 \circ \cdots \circ s_n \sqsubseteq r$
Semantics	$s_1^{\mathcal{I}} \circ \cdots \circ s_n^{\mathcal{I}} \subseteq r^{\mathcal{I}}$	

  -

Name	role disjointness	global reflexivity
Syntax	$Dis(r, s)$	$Ref(r)$
Semantics	$r^{\mathcal{I}} \cup s^{\mathcal{I}} = \emptyset$	$\{(x, x) \mid x \in \Delta^{\mathcal{I}}\} \subseteq r^{\mathcal{I}}$

  - *Additional Axioms*:

Name	*Syntax*	*Defined as*
disjointness	$Dis(C, D)$	$C \sqsubseteq \neg D \vee D \sqsubseteq \neg C \vee C \sqcap D \sqsubseteq \bot$
role equivalence	$r \equiv s$	$r \sqsubseteq s, s \sqsubseteq r$
domain restriction	$Dom(r) \sqsubseteq C$	$\top \sqsubseteq \forall r^{-}.c \vee \exists r.\top \sqsubseteq C$
range restriction	$Ran(r) \sqsubseteq C$	$\top \sqsubseteq \forall r.C \vee \exists r^{-}.\top \sqsubseteq C$
role irreflexivity	$Irr(r)$	$\exists r.Self \sqsubseteq \bot$
role functionality	$Fun(r)$	$\top \sqsubseteq \leq 1r$
role symmetry	$Sym(r)$	$r \sqsubseteq r^{-}$
role asymmetry	$Asy(r)$	$Dis(r, r^{-})$
role transitivity	$Tra(r)$	$r \circ r \sqsubseteq r$

- *Concrete Domains* $\rightarrow \Delta^D = \mathbb{Z}$, One$^D = 1$, Even$^D = \{..., -2, 0, 2, ...\}$
- *Attributes*
  - Let R_C be an infinite set of concrete role names (also called attribute names)
  - extend the definitions of an Interpretation $\mathcal{I}$ to assign each $u \in R_C$ a binary relation $u^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
  - Example: (APB/E005, 30):hasSize, Fun(hasSize)
- *Concrete Role Restrictions*
  -

Name	(concrete) existential restriction
Syntax	$\exists u.P$
Semantics	$\{d \mid \exists v.(d, v) \in u^{\mathcal{I}} \wedge v \in P^D\}$
Example	∃hasSize.Positive, ∃hasPrice.≥ 1000 EUR

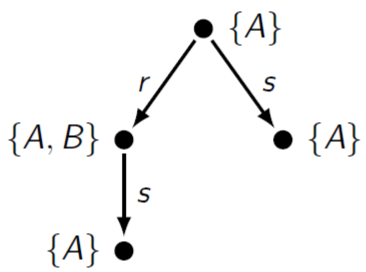
3 EL

Much simpler DL with much lower complexity, EL can be reasoned in PTime

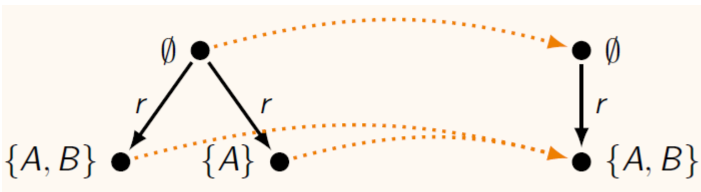
- $\mathcal{EL}$ is the sublogic of $\mathcal{ALC}$ that only allows $\top, \sqcap, \exists r.C$
- all $\mathcal{EL}$ ontologies are consistent and all concepts within are satisfiable
- an Atom is a concept name or an existential restriction
  $(At(C) := \{C_1, ..., C_n\})$
- every concept C is equivalent to a concept of the form $C_1 \sqcap \cdots \sqcap C_n$
  ($C = \top$ is the empty conjunction with $At(\top) = \emptyset$)
- in $\mathcal{EL}$ concepts are expressed as description trees

3.1 Description Trees

- description Tree $T = (V, E, v_0, l)$ with:
  - root node: v_0
  - nodes $v \in V$
  - edged $e \in E$
  - labeling function $l(v) \subseteq C$ which assigns each node a finite set of concepts and $l(e) \in R$ assigns each edge role names
- the root node v_0 is labeled by $l(v) := At(C) \cap C$
- for every $\exists r.D \in At(C), \exists(v_0, v_{\exists r.D}) : l(v_0, v_{\exists r.D}) := r$ where $v_{\exists r.D}$ is the root of a copy of the description tree T_D of D
- Example: $A \sqcap \exists r.(A \sqcap (\exists s.A) \sqcap B) \sqcap Es.A$



- the tree T_C can be seen as an interpretation whose root satisfies C
  this interpretation is called the canonical models of C

- *Proof* (by induction on the role depth of C):
  - base case ($rd(C) = 0$):
    * C is a conjunction of concept names and $\top$
    * for every $A \in At(C)$, $A \in l(v_0)$ and thus $v_0 \in A^{\mathcal{I}_C}$
    * consequently, $v_0 \in \left(\bigsqcap_{A \in At(C)} A\right)^{\mathcal{I}_C} = C^{\mathcal{I}_C}$
  - step (rd(C) > 0):
    * For $D = A \in At(C) \cap C$ the case is as for $rd(C) = 0$
    * Assume $D = \exists r.E \in At(C)$, then $(v_0, v_{\exists r.E}) \in E$, where $l(v_0, v_{\exists r.E}) = r$ and $v_0, v_{\exists r.E}$ is the root in the description three of E
    * $rd(E) < rd(C)$ and by IH, $v_{\exists r.E} \in E^{\mathcal{I}_C}$
    * $(v_0, v_{\exists r.E}) \in r^{\mathcal{I}_C}$, and thus $v_0 \in (\exists r.E)^{\mathcal{I}_C}$
    * We obtain $v_0 \in \left(\bigsqcap_{D \in At(C)} D\right)^{\mathcal{I}_C} = C^{\mathcal{I}_C}$
- homomorphism between description trees $T_1 = (V_1, E_1, v_1, l_1)$, $T_2 = (V_2, E_2, v_2, l_2)$
  - $h : V_1 \to V_2$
  - $h(v_1) = v_2$
  - $l_1(v) \subseteq l_2(h(v)), \forall v \in V_1$
  - $\forall (v, w) \in E_1\ \exists (h(v), h(w)) \in E_2$ each edge with the same label
  - 

  - for two $\mathcal{EL}$ concepts C, D, the axiom $C \sqsubseteq D$ is a tautology iff there is a homomorphism from T_D to T_C
- for two $\mathcal{EL}$ concepts C, D, the axiom $C \sqsubseteq D$ is a tautology iff $\forall D' \in At(D)\ \exists C' \in At(C) : (C', D' \in C \wedge C' = D') \vee$ $(C' = \exists r.C'',\ D' = \exists r.D'' \wedge C'' \sqsubseteq D'')$is a tautology
- for two $\mathcal{EL}$ concepts $C \equiv D \Leftrightarrow T_C \cong T_D$

- every $\mathcal{EL}$ concept can be reduced by applying the following rules exhaustively:
  - $C \sqcap \top \to C$
  - $C \sqcap C \to C$
  - $\exists r.C \sqcap \exists r.D \to \exists r.C$ if $C \sqsubseteq D$ is a tautology,

3.2 Reasoning

- *Normal Form of EL TBoxes*
  - A TBox $\mathcal{T}$ is in normal form if all CGIs have the form:
    * $A_1 \sqcap \cdots A_n \sqsubseteq B$
    * $A \sqsubseteq \exists r.B$
    * $\exists r.A \sqsubseteq B$
  - if a TBox is not in normal form you may apply those normalization rules;
    * $C \sqsubseteq D \sqcap E \to C \sqsubseteq D,\ C \sqsubseteq E$
    * $C \sqsubseteq \exists r.\hat{D} \to C \sqsubseteq \exists r.A_{\hat{D}},\ A_{\hat{D}} \sqsubseteq \hat{D}$
    * $\hat{C} \sqsubseteq \hat{D} \to \hat{C} \sqsubseteq A_{\hat{D}},\ A_{\hat{D}} \sqsubseteq \hat{D}$
    * $C \sqcap \hat{D} \sqsubseteq E \to \hat{D} \sqsubseteq A_{\hat{D}},\ C \sqcap A_{\hat{D}} \sqsubseteq E$
    * $\exists r.\hat{C} \sqsubseteq D \to \hat{C} \sqsubseteq A_{\hat{C}},\ \exists r.A_{\hat{C}} \sqsubseteq D$
    * $\hat{C}, \hat{D}$ are neither concept names nor $\top$ and $A_{\hat{C}}, A_{\hat{D}}$ is a fresh concept name
- with classification one can decide several concept subsumptions at once (Determine all $A \sqsubseteq_{\mathcal{T}} B$ where $A, B \in C$)
  - *classification rules*:
    * $\dfrac{}{A \sqsubseteq A}$ (CR1)
    * $\dfrac{}{A \sqsubseteq \top}$ (CR2)
    * $\dfrac{A \sqsubseteq A_1 \cdots A \sqsubseteq A_n \quad A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B}{A \sqsubseteq B}$ (CR3)
    * $\dfrac{A \sqsubseteq \exists r.B \quad B \sqsubseteq C \quad \exists r.C \sqsubseteq D}{A \sqsubseteq D}$ (CR4)
    * if the premise is in $\mathcal{T}$ but the conclusion is not, then add the conclusion to $\mathcal{T}$

- every rule adds a new axiom $A \sqsubseteq B \Rightarrow$ at most quadratic many inferences
- the classification terminates in PTime in the size of $\mathcal{T}$
- classification can also be use to check complex subsumptions $C \sqsubseteq_{\mathcal{T}} D$, where C, D are not just concept names
  * add the CGIs $A_C \sqsubseteq C,\ D \sqsubseteq B_D$ to $\mathcal{T}$, where A_C, B_D are fresh concept names
  * normalize the extended TBox
  * check whether $A_C \sqsubseteq B_D$ is entailed from it

4 Ontology Learning

Extraction knowledge from a source (text, database) and transform it into an ontology

- *Learning Problem*:
  - $\mathcal{O}$ a consistent ontology
  - $A \in C$ be the target concept name
  - $E^+ \subseteq I(\mathcal{O})$ be a set of positive examples for A
  - $E^- \subseteq I(\mathcal{O})$ be a set of negative examples for A
  - concept learning problem:
    * $\mathcal{O} \models a : C_A \forall a \in E^+$
    * $\mathcal{O} \not\models a : C_A \forall a \in E^-$
    * $\rightarrow$ the ontology models positive but not negative examples
- to avoid overfitting we can restrict C_A to be an $\mathcal{ALC}$ concept
  $\rightarrow$a exact solution then may not exist but an approximation does
- *finding approximate solutions*
  - generate candidates for C_A called hypotheses
  - evaluate the hypotheses
    * $fn(C_A) := \#\{a \in E^+ \mid \mathcal{O} \not\models a : C_A\}$ false negatives
    * $fp(C_A) := \#\{a \in E^- \mid \mathcal{O} \models a : C_A\}$ false positives
    * $acc(C_A) := 1 - \dfrac{fn(C_A) + fp(C_A)}{\#E^+ + \#E^-}$ accuracy

* $score(C_A) := acc(C_A) - \beta \cdot size(C_A)$ ($\beta \in [0,1]$ is a parameter)
* choose the best C_A hypotheses based on score
* add new concept definition $A \equiv C_A$ to $\mathcal{O}$

- *how to find a good hypothesis*:
  * Start with $C_A = \top$ which has all positive and negative examples as instances and refine concept iterativly
  * downward refinement operator ρ
    - $\rho(C) \subseteq \{D \mid D \sqsubseteq_{\mathcal{O}} C\}$
    - each $D \in \rho(C)$ is formulated using the signature of $\mathcal{O}$
    - write $C \to_\rho D$ if $D \in \rho(C)$
    - $\to_\rho^*$ is the reflexive transitive closure of $\to_\rho$
    - D can be reach from C via $\to_\rho$ if $C \to_\rho^* D$
  * *properties of refinement operator*
    - *(locally) finite*: if $\rho(C)$ is finite for all concepts C
    - *proper*: if $C \to_\rho D \Rightarrow D \sqsubset_{\mathcal{O}} C$
    - *complete*: if $D \sqsubset_{\mathcal{O}} \Rightarrow C \to_\rho^* E$ for some concept $E \equiv_{\mathcal{O}} D$
    - *ideal*: if it is finite proper and complete
      (in each step only finitely many new hypothesis are created which are not equivalent to a previous one and also from which all more specific concepts can be reached from)
  * every $\mathcal{ALC}$ ontology has a complete and finite refinement operator but there is no ideal refinement operator

- *refining concepts*
  * define $\downarrow(A)$ as the lower neighbors and $\uparrow(A)$ as the upper neighbors of A
  * define the atomic domain $ADom(r)$ as the unique minimal $A \in C \cup \{\top, \bot\}$ s.d. $\mathcal{O} \models Dom(r) \sqsubseteq A$
  * define the atomic range $ARan(r)$ similarly
  * instead of $\rho_\top$ (special case of $B = \top$ we define ρ_B relative to a context $B \in C \cup \{\top, \bot\}$
    - $\rho_B(C) := \rho'_B(C) \cup \{\bot, C \sqcap \top\}$
    - $\rho'_B(\top) := \{C_1 \sqcup \cdots \sqcup C_n \mid C_1, \cdots, C_n$ are of the form (a)-(c)$\}$
    - $\rho'_B(\bot) := \emptyset$
    - $\rho'_B(A) := \{E \mid E \in \downarrow(A),\ B \sqcap E \not\sqsubseteq_o nt\bot\}$
    - $\rho'_B(\neg A) := \{\neg E \mid E \in \uparrow(A),\ B \sqcap \neg E \not\sqsubseteq_{\mathcal{O}} \bot\}$

· $\rho'_B(\exists r.D) := \{\exists r.E \mid E \in \rho_{ARan(r)}(D)\}$
· $\rho'_B(\forall r.D) := \{\forall r.E \mid E \in \rho_{ARan(r)}(D)\}$
· $\rho'_B(C_1 \sqcap C_2) := \{C_1 \sqcap E \mid E \in \rho_B(C_2)\} \cup \{E \sqcap C_2 \mid E \in \rho_B(C_1)\}$
· $\rho'_B(C_1 \sqcup C_2) := \{C_1 \sqcup E \mid E \in \rho_B(C_2)\} \cup \{E \sqcup C_2 \mid E \in \rho_B(C_1\}$

  * $\rho_\top$ is complete for $\mathcal{ALC}$
  * $\rho_\top$ is not proper for $\mathcal{ALC}$
    ($\rho_\top(\exists r.D)$ contains $\exists r.D \sqcap \top$ which is equivalent to $\exists r.D$)
- $\rho_\top^{\sqsubseteq}$ is complete and proper but not finite

- *Algorithm*
  - **Input**: Ontology $\mathcal{O}$, concept names $A, E^+, E^- \subseteq I(\mathcal{O})$, parameter β
  - **Output**: A list of candidates for C_A

1. initialize search tree with a singe node labeled by $(\top, 0)$
2. **while** true **do**:
   - choose a node labeled by (C, n) with maximal $acc(C) - \beta \cdot n$
   - **for** all $D \in \rho_\top^{\sqsubseteq}(C)$ with $size(D) = n + 1$ and $D \not\equiv_\mathcal{O} \bot$ **do**:
     * create a child node with label (D, n)
   - change the label of the current node from (C, n) to $(C, n + 1)$
   - stop at any time
3. **return** all concepts in the search tree (ranked by score)

- the algorithm can be improved further by the use of a heuristic

5 Matching/Aligning Ontologies

Given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, an alignment $\mathfrak{A}$ is a third ontology that shares the vocabulary of $\mathcal{O}_1$ and $\mathcal{O}_2$. $\mathfrak{A}$ contains bridge axioms that relate one ore more entities of the ontologies

- *correspondence* (between $\mathcal{O}_1$ and $\mathcal{O}_2$)
  - (e_1, r, e_2, c)
    * e_1 is in $\mathcal{O}_1$ and e_2 is in $\mathcal{O}_2$
    * either $e_1, e_2 \in C,\ e_1, e_2 \in R,$ or $e_1, e_2 \in I$

        * $r \in \{\equiv, \sqsubseteq, \sqsupseteq, \perp, \between\}$
        * $c \in [0, 1]$ is a confidence value
    - Example: (ont1:Arm,≡,ont2:Arm,0.95)

- similarity can be measured using a similarity measure $\sigma M_1 \times M_2 \to [0, 1]$ (the closer the value is to 1 to more similar are m_1 and m_2)

- concept name similarity can be derived from other similarity measures

    - given $f_1 : N_1 \to M_1$ and $f_2 :\to M_2$, the similarity measure induced by σ, f_1, and f_2 is
      $\sigma' : N_1 \times N_2 \to [0, 1]$, where $\sigma'(n_1, n_2) := \sigma(f_1(n_1), f_2(n_2))$
    - Example: σ'(C20480, GO_0009987) := σ(Cellular Process", cellular process") = 1

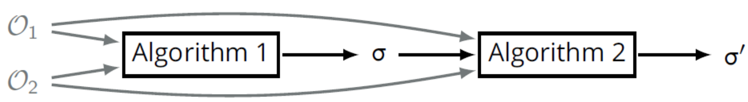
- the distance measure δ induced by σ is $\delta : M_1 \times M_2 \to [0, 1]$, defined by $\delta(m_1, m_2) := 1 - \sigma(m_1, m_2)$
  (similarity and distance measure contain the same information, its only for convenience)

- *other distance/similarity measures*:

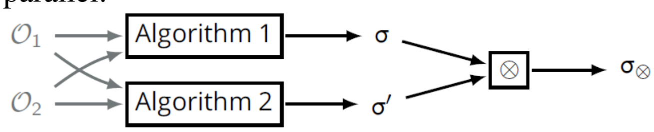
    - hamming distance: $\delta(v, w) := \frac{1}{m} \cdot |\{i \in \{1, ..., n\} \mid v_i \neq w_i\}|$
    - substring similarity: $\sigma(v, w) := \frac{|u|}{m}$, where u is the longest common substring of v and w
    - **n-gram similarity**: $\sigma(v, w) := \frac{|n - gram(v) \cap n - gram(w)|}{m - n + 1}$,
      where $n - gram(v)$ is the set of n-letter substrings of v
    - levenshtein (edit) distance: minimal number of (insert, delete, replace) operations to get from w to v (divided by m)

- similarity measures can be combined

    - sequentially:



    - parallel:



- similarity measures can be aggregated (by an aggregation operator $\otimes$)
  - weighted sum
  - weighted product
  - triangular norm (t-norm) $\otimes : [0,1] \times [0,1] \to [0,1]$ that is associative, commutative, monotone and has neutral element 1
  - triangular conorm (t-conorm) like t-norm but with neutral element 0
- similarity can also be defined for concepts
  - directed similarity between to Concepts C, D
    * $= \sigma_d(C, D) :=$
    $$\begin{cases} \sigma'(C, D), & C, D \in C \\ \sigma'(r, s) \cdot (\beta + (1 - \beta \cdot \sigma_d(E, F)), & C = \exists r.E \wedge D = \exists s.F \\ \min\limits_{D' \in At(D)} \max\limits_{C' \in At(C)} \sigma_d(C', D'), & |At(C)| > 1 \vee |At(D)| > 1 \\ 1, & At(D) = \emptyset \\ 0, & \text{otherwise} \end{cases}$$
    * $\delta_d(C, D)$ measures how C is subsumed by D
  - undirected similarity
    * $\sigma_u(C, D) := \sigma_d(C, D) \otimes \sigma_d(D, C)$
    * $\otimes$ is a commutative aggregation operator
- A similarity should be equivalence invariant: $\sigma(\exists r.B, \exists s.D)$ as a function $\sigma'(B, D)$ if $B \equiv_{\mathcal{O}} C$, then $\sigma(\exists r.B, \exists s.D)$ should be equal to $\sigma(\exists r.C, \exists s.D)$
- *from similarity to alignment*
  - given a similarity measure, each concept may be similar to multiple concepts but we only want the best ones
  - *Threshold-based methods* ($\tau \in [0, 1]$)
    * hard threshold: $\sigma(A, B) \geq \tau$
    * delta threshold: $\sigma(A, B) \geq \max\limits_{\sigma} - \tau$ with $\max_\delta := \max_{A', B'}\{\sigma(A', B')\}$
    * proportional threshold: $\sigma(A, B) \geq \tau \cdot \max_\sigma$
    * percentage threshold: it is among the $\tau \cdot |C(\mathcal{O}_1)| \cdot |C(\mathcal{O}_2)|$ correspondences with the highest similarity

- ∗ normalized threshold: $\dfrac{\sigma(A,B)}{\max\{\sigma(A,B')\}} \geq \tau$ and $\dfrac{\sigma(A,B)}{\max\{\sigma(A',B)\}} \geq \tau$
- sometimes a bijective alignment is needed, where one concept name has exactly one correspondence
  - ∗ greedy algorithm
  - ∗ maximum weight graph matching

6 Ontology Maintenance

- *Justification*
  - process of finding axioms responsible for erroneous entailment
  - a justification $\mathfrak{J} \subseteq \mathcal{O}$ for an axiom α is:
    - ∗ $\mathfrak{J} \models \alpha$
    - ∗ $\mathfrak{J}$ is a minimal set with this property $\rightarrow \forall \mathfrak{J}' \subset \mathfrak{J} : \mathfrak{J}' \not\models \alpha$
  - a justification provides an explanation for the error α
  - computing a single justification is not enough to fix the error because the error may be caused by other clauses for the entailment of α
  - *How to compute justifications*
    - ∗ Black-box algorithms
      - · Use reasoner to decide $\mathcal{O} \models \alpha$ and construct justification as a series of calls to the reasoner
    - ∗ Glass-box algorithms
      - · Extend an existing reasoning algorithm for checking $\mathcal{O} \models \alpha$ to trace the axioms from $\mathcal{O}$ that are used to derive α $\rightarrow$ faster but requires more knowledge about the reasoner (harder to implement)
  - to compute all Justification we need an algorithm to compute a single justification *SingleJustification* (black box algorithm):
    - ∗ **Input**: Ontologies $\mathcal{O}_f, \mathcal{O}$, axiom α with $\mathcal{O}_f \not\models \alpha$ and $\mathcal{O}_f \cup \mathcal{O} \models \alpha$
    - ∗ **Output**: A minimal subset $\hat{\mathcal{O}} \subseteq \mathcal{O} : \mathcal{O}_f \cup \hat{\mathcal{O}} \models \alpha$
    1. **if** $|\mathcal{O}| = 1$ **then return** $\mathcal{O}$
    2. split $\mathcal{O}$ int two halves $\mathcal{O}_1$ and $\mathcal{O}_2$

3. **if** $\mathcal{O}_f \cup \mathcal{O}_1 \models \alpha$ **then return** SingleJustification($\mathcal{O}_f$, ont_1, α)
4. **if** $\mathcal{O}_f \cup \mathcal{O}_2 \models \alpha$ **then return** SingleJustification($\mathcal{O}_f$, ont_2, α)
   · minimize $\mathcal{O}_1$ while fixing $\mathcal{O}_2$, and vice versa
5. $\mathcal{O}_1'$:= **SingleJustification**($\mathcal{O}_f \cup \mathcal{O}_2$,$\mathcal{O}_1$,$\alpha$)
6. $\mathcal{O}_2'$:= **SingleJustification**($\mathcal{O}_f \cup \mathcal{O}_1'$,$\mathcal{O}_2$,$\alpha$)
7. **return** $\mathcal{O}_1' \cup \mathcal{O}_2'$

∗ to compute a single justification for α in $\mathcal{O}$ we can call **SingleJustification($\emptyset$, $\mathcal{O}$, α)**

– glass box algorithm (pinpointing)

∗ idea: Assign unique labels to axioms in $\mathcal{O}$, a formula over the labels describes then a subset of $\mathcal{O}$

∗ pinpointing algorithm for $\mathbb{EL}$

· $\dfrac{}{(A \sqsubseteq A)^{\text{true}}}$ (CR1)

· $\dfrac{}{(A \sqsubseteq \top)^{\text{true}}}$ (CR2)

· $\dfrac{(A \sqsubseteq A_1)^{\varphi_1} \cdots (A \sqsubseteq A_n)^{\varphi_n} \quad (A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B)^{\varphi}}{(A \sqsubseteq B)^{\varphi_1 \wedge \cdots \wedge \varphi_n \wedge \varphi}}$ (CR3)

· $\dfrac{(A \sqsubseteq \exists r.B)^{\varphi_1} \quad (B \sqsubseteq C)^{\varphi_2} \quad (\exists r.C \sqsubseteq D)^{\varphi_3}}{(A \sqsubseteq D)^{\varphi_1 \wedge \varphi_2 \wedge \varphi_3}}$ (CR4)