A description language

Computing satisfaction

We are interested whether KB $\models$ ($b \rightarrow e$), where b is a constant and e is a concept.

To find out if an individual satisfies a description, we need to propagate the information implied by what we know about other individuals before checking for subsumption. This can be done by a forward chaining procedure.

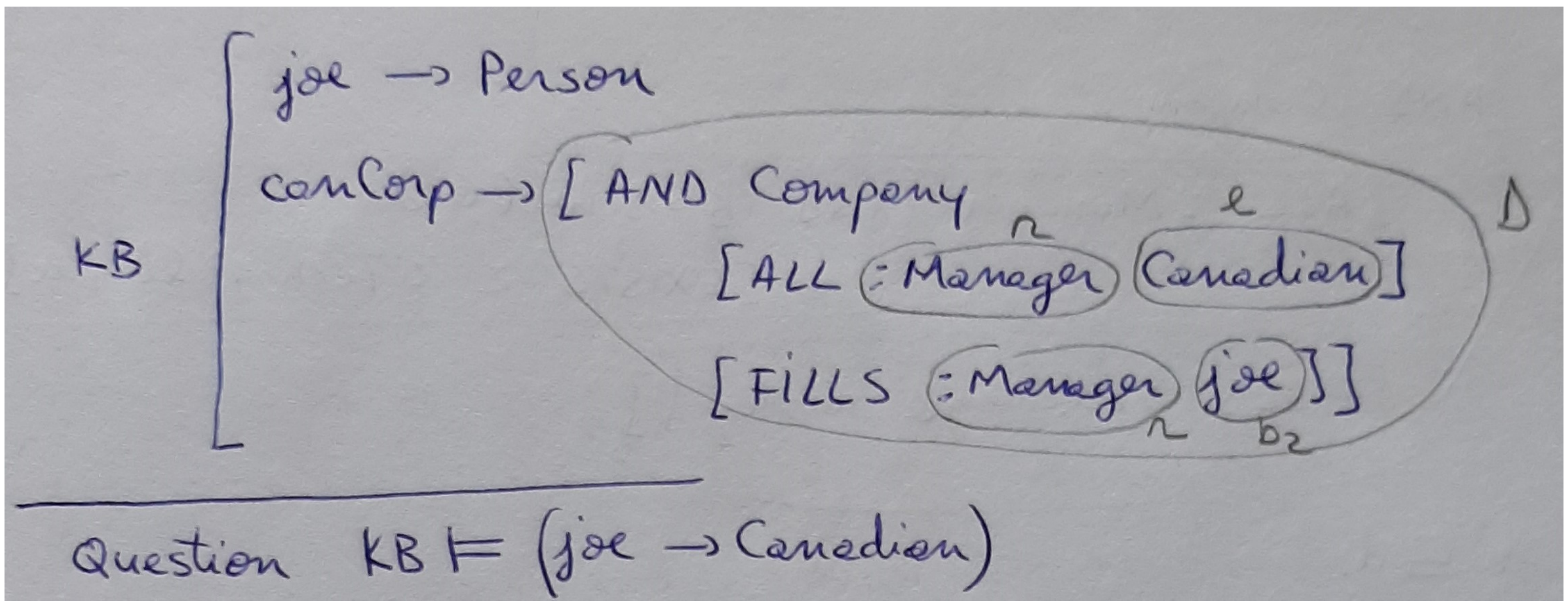
In the case where there are no EXISTS terms in any concept, the procedure is as following:

1. Construct S a list of pairs (b,d), where b is any constant mentioned in KB and d is the normalized version of the concept [AND d_1'…d_n'] for all d_i' such that ($b \rightarrow d_n'$)$\in$KB.
2. Find two constants b_1 and b_2 such that (b_1,d_1)$\in$S and (b_2,d_2)$\in$S, [FILLS r b_2] and [ALL r e] are both components of d_1 but KB $\not\models$ ($d_2 \sqsubseteq e$).
3. If no b_1 and b_2 can be found, then exit. Otherwise, replace the pair (b_2,d_2) in S by (b_2, d_2'), where d_2' is the normalized version of [AND d_2 e] and go to step 2.

A description language

The procedure computes for each constant b the most specific concept d such that KB ⊨ (b→d). Now, to test whether or not KB ⊨ (b→e), we need only to test whether or not KB ⊨ (d ⊑e).

Example 1



=>S={(joe,[AND Person Canadian]),(canCorp,D)}. Now, the procedure terminates because KB ⊨ ([AND Person Canadian] ⊑Canadian).

Because KB ⊨ ([AND Person Canadian] ⊑Canadian) it follows that

KB ⊨ (joe →Canadian).

A description language

In the case where there are EXISTS terms of the form [EXISTS 1 r], we will use role chains

$[\text{AND}\ldots[\text{ALL } r_1\ldots[\text{AND}\ldots[\text{ALL } r_k \text{ a}]\ldots]\ldots]\ldots]$

$\sigma=r_1\cdot\ldots\cdot r_k$ is called a role chain.

If b is a constant and r_1, r_2 roles, then $b\cdot r_1\cdot r_2$ represents an individual (perhaps unnamed) that is in relation r_2 with an individual that is in relation r_1 with b.

If σ is empty, then $b\cdot\sigma$ is b.

The forward chaining procedure extends by adding two steps:

A description language

in slide 1

1. Construct S a list of pairs (b,d), where b is any constant mentioned in KB and d is the normalized version of the concept [AND $d_1´…d_n´$] for all $d_i´$ such that $(b \rightarrow d_n´) \in KB$.
2. Find two constants b_1 and b_2 such that $(b_1,d_1) \in S$ and $(b_2,d_2) \in S$, [FILLS r b_2] and [ALL r e] are both components of d_1 but KB $\not\models$ $(d_2 \sqsubseteq e)$.
3. If no b_1 and b_2 can be found, then go to step 4. Otherwise, replace the pair (b_2,d_2) in S by $(b_2, d_2´)$, where $d_2´$ is the normalized version of [AND d_2 e] and go to step 2.

4. Find a constant b, a role chain σ (possibly empty) and a role r such that $(b \cdot \sigma, d_1) \in S$ and $(b \cdot \sigma \cdot r, d_2) \in S$ (if no such pair exists, take d_2 to be Thing), where [EXISTS 1 r] and [ALL r e] are components of d_1, but KB $\not\models$ $(d_2 \sqsubseteq e)$.
5. If these can be found, remove $(b \cdot \sigma \cdot r, d_2)$ from S (if applicable) and add the pair $(b \cdot \sigma \cdot r, d_2´)$, where $d_2´$ is the normalized version of [AND d_2 e]; then go to step 2. Otherwise exit.

We start with a property of the individual $b \cdot \sigma$ and conclude something new about the (unnamed) individual $b \cdot \sigma \cdot r$. Eventually, this can lead to new information about a named individual.

A description language

Example 4 Assume that we have in KB the sentence

b·σ σ is empty

r d1

ellen → [AND [EXISTS 1 :Child]

e

[ALL :Child [AND [FILLS :Peditrician marianne]

r

[ALL :Peditrician Scandinavian]]

]

]

Question: KB ⊨ (marianne→Scandinavian)

S={(b·σ,d_1)} and (b·σ·r,d_2)=(ellen :Child, Thing)

Because KB ⊭ (Thing⊑e), S becomes

S={(b·σ,d_1),(ellen :Child, [AND [FILLS :Peditrician marianne]
[ALL :Peditrician Scandinavian]])}

b·σ·r

From here, we conclude that (marianne→Scandinavian) (case with no EXISTS).

The case of terms of the form [EXISTS n r], n>1 is handled the same as for n=1. There is no need to create n different anonymous individuals because all of them would “produce” the same properties in the forward chaining.

A description language

Taxonomies and classification

Given a concept q, in DL it is common to ask for all of its instances, that is to find all c in KB so that KB $\models$ (c $\rightarrow$q).

Also, it is common to ask for all of the known categories that an individual satisfies. That is to say that given a constant c, we should find all concepts a so that KB $\models$ (c $\rightarrow$a).

When reasoning in DL, we should exploit the hierarchical organization of the concepts, with the most general ones at the top and the more specialized ones further down.

To represent sentences in KB, we use a taxonomy (a treelike data structure) that allows answering queries efficiently (time linear with the depth of the taxonomy, not with its size).

A description language

Obs. Subsumption is a partial order.

The taxonomy have atomic concepts as nodes and edges like

$$\begin{tikzcd} a_j \\ a_i \arrow[u] \end{tikzcd}$$

whenever $a_i \sqsubseteq a_j$ and there is no a_k such that $a_i \sqsubseteq a_k \sqsubseteq a_j$.

Each constant c in KB will be linked to the most specific atomic concept a_i such that KB $\models (c \rightarrow a_i)$.

Adding some new atomic concept or constant to a taxonomy corresponding to a KB is called classification. It involves creating a link from the new concept or constant to existing ones in the taxonomy.

This process exploits the structure of the taxonomy. We start with the concept Thing and then add incrementally new atomic concepts and constants.

A description language

Computing classification

I. Add a sentence ($anew \doteq d$) to the taxonomy, where anew is an atomic concept not appearing anywhere in the KB and d is any concept:

1. Compute S, the most specific subsumers of d

   $S=\{a$ – concept in the taxonomy$|$ KB $\models (d \sqsubseteq a)$, but $\nexists a' \neq a$ so that KB $\models (d \sqsubseteq a')$ and KB $\models (a' \sqsubseteq a)\}$

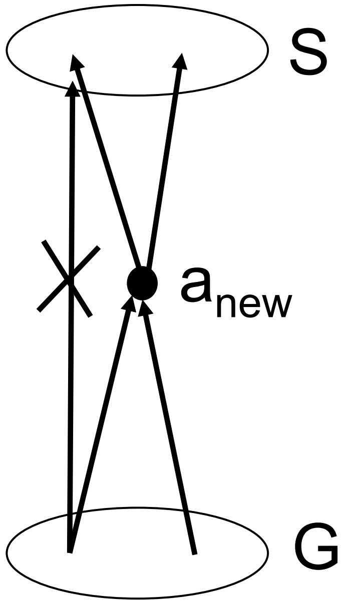
2. Compute G, the most general subsumees of d

   $G=\{a$ – concept in the taxonomy$|$ KB $\models (a \sqsubseteq d)$, but $\nexists a' \neq a$ so that KB $\models (a' \sqsubseteq d)$ and KB $\models (a \sqsubseteq a')\}$

3. If $\exists a \in S \cap G$ then a_{new} is already in the taxonomy under a different name – no action needed

4. Otherwise remove all links (if any) from concepts in G up to concepts in S

5. Add links from a_{new} up to each concept in S and links from each concept in G up to a_{new}



A description language

6. Handling constants

Compute C={c constant in taxonomy| $\forall a \in S$, KB $\models (c \rightarrow a)$ and $\nexists a' \in G$ such that KB $\models (c \rightarrow a')$}

Then for each $c \in C$ we test if KB $\models (c \rightarrow d)$ and if so, we remove the links from c to S and add a single link from c to a_{new}.

II. Add a sentence ($a_{new} \sqsubseteq d$) reduces to adding links from a_{new} to the most specific subsumers of d.

III. Add a sentence ($c_{new} \rightarrow d$) reduces to adding links from c_{new} to the most specific subsumers of d.

A description language

Compute S – the most specific subsumers of d

Start with S={Thing}

For all $a \in S$ if $\exists a'$ so that

$$\begin{tikzcd} a \\ a' \arrow[u] \end{tikzcd}$$

and $KB \models (d \sqsubseteq a')$ then remove a from S and add all a' in S.

Repeat until no element in S has a child that subsumes d.

Compute G – the most general subsumees of d

Start with G=S

If $\exists a \in G$ so that $KB \not\models (a \sqsubseteq d)$, then replace a with all its children (or delete it if it has no children).

Repeat until each element in G is subsumed by d.

Finally, we delete each $a \in G$ that has a parent subsumed by d.

A description language

Compute S – the most specific subsumers of d

Start with S={Thing}

For all $a \in S$ if $\exists a'$ so that $\begin{array}{c} a \\ \uparrow \\ a' \end{array}$

and KB $\models (d \sqsubseteq a')$ then remove a from S and add all a' in S.

Repeat until no element in S has a child that subsumes d.

Compute G – the most general subsumees of d

Start with G=S

If $\exists a \in G$ so that KB $\not\models (a \sqsubseteq d)$, then replace a with all its children (or delete it if it has no children).

Repeat until each element in G is subsumed by d.

Finally, we delete each $a \in G$ that has a parent subsumed by d.

Answering questions in DL

To find all constants c that satisfy a concept q, we should classify q and then collect all the constants at the fringe of the tree bellow q in the taxonomy.

To find all atomic concepts that are satisfied by a constant c, we go from c up in the taxonomy, collecting all the nodes that can be reached.

Defaults

Frames offer a simple form of default reasoning, where a slot has a certain value by inheritance unless another one is explicitly given.

Assuming that we have Dog(fido) in a KB in FOL, there are only two ways to reach the conclusion Carnivore(fido):

1. This is explicitly mentioned in KB;
2. In KB we have the universal form $\forall x.Dog(x) \supset Carnivore(x)$.

We are interested in expressing in FOL what we know about something in general and in particular using universals.

Defaults

For example, we may say that “Bikes have two wheels” but without stating that “All bikes have two wheels” because this would rule out a bike with three wheels.

A possible solution would be to say

“All bikes that are not P_1 or …or P_n have two wheels”,

where P_i represent the exceptional cases. The challenge here is to characterize these cases.

We would like to make a distinction between <u>universals</u> that hold for all instances and <u>generics</u> that hold in general.

Much of our knowledge about the world is generic, therefore it is important to formalize it.

Defaults

Default reasoning

In general, we know that dogs are carnivores. If Fido is a dog, what are the appropriate circumstances to infer that Fido is a carnivore? We can reason as following:

> Given that a P is generally a Q and knowing that P(a) is true, it is reasonable to conclude that Q(a) is true unless there is a good reason not to.

If all that we know about an individual is that it is an instance of P, then there is no reason not to conclude that it is an instance of Q.

For example, if we know that a polar bear has been playing in the mud, probably we do not want to conclude anything about its color. But if all we know is that it is a polar bear, then it is reasonable to conclude that it is white. The conclusion has no guarantee to be logically correct, it is only a reasonable default.

Defaults

This form of reasoning that involves general (not universal) knowledge about a particular individual, is called default reasoning.

Examples of situations when we want to conclude Q(a) given P(a):

-general statements: Children love playing.

Oranges are orange.

People in a long queue become impatient.

-lack of information to the contrary:

No country has a president taller than 2m.

Children learn easily foreign languages.

-conventions: The speed limit in a city.

The closest shop is five minute walk (by default assume that it is open).

-persistance: Marital status.

The size of objects.

The list of objects is not exhaustive, but it suggests the great variety of sources of default information. Our focus is to describe exactly when it is appropriate to draw a default conclusion, in the absence of universals.

Defaults

Nonmonotonicity

Ordinary deductive reasoning is monotonic, meaning that new facts produce only additional beliefs. If $KB_1 \models \alpha$ then $KB_2 \models \alpha$ for any KB_2 such that $KB_1 \subseteq KB_2$.

Default reasoning is nonmonotonic, meaning that sometimes new facts invalidate previous beliefs. For example, we believe by default that a bird flies, but if we find that the bird is an ostrich, we reconsider our belief.

Defaults

I. Closed-world reasoning

It is the simplest formalization of default reasoning. A finite vocabulary of predicate and constant symbols is used to represent facts about the world. But from all the valid atomic sentences, only a small fraction of them are expected to be true. The convention here is to explicitly represent the true atomic sentences and to assume that any unmentioned one is false.

Example:

DirectConnect(cleveland, toronto)

DirectConnect(toronto, chicago)

DirectConnect(cleveland, vancouver)

If a flight between two cities is not listed, then there is none. The closed-world assumption (CWA) is the following:

Unless an atomic sentence is known to be true, it can be assumed to be false.

Obs. A sentence assumed to be false can be later determined to be true.

Defaults

Def. $KB^+ = KB \cup \{\neg p \mid p \text{ is atomic and } KB \not\models p\}$. A new form of entailment is defined as following:

$$KB \models_c \alpha \text{ iff } KB^+ \models \alpha$$

In the previous example, KB^+ would include sentences of the form $\neg DirectConnect(c_1,c_2)$.

Consistency and completeness of knowledge

Def. A KB exhibits consistent knowledge iff there is no sentence α such that both α and ¬α are known.

Def. A KB exhibits complete knowledge iff for every sentence α, either α or ¬α is known.

Defaults

Knowledge can be incomplete. For example, if KB={(p∨q)}, neither p nor ¬p can be entailed from KB.

But with the CWA, the entailment relation is complete. For any sentence α, it holds that either KB $\models_c$ α or KB $\models_c$ ¬α (demonstration is by induction on the length of α).

Under CWA, whenever KB$\not\models$p then either KB $\models_c$ ¬p directly or ¬p is conceptually added to the KB. That means that we act as if KB represents completely knowledge.

Defaults

Query evaluation

The question KB $\models_c \alpha$ reduces to questions about the literals in α:

1. $KB \models (\alpha \wedge \beta)$ iff $KB \models \alpha$ and $KB \models \beta$
2. $KB \models \neg\neg\alpha$ iff $KB \models \alpha$
3. $KB \models \neg(\alpha \vee \beta)$ iff $KB \models \neg\alpha$ and $KB \models \neg\beta$

4. $KB \models_c (\alpha \vee \beta)$ iff $KB \models_c \alpha$ or $KB \models_c \beta$
5. $KB \models_c \neg(\alpha \vee \beta)$ iff $KB \models_c \neg\alpha$ or $KB \models_c \neg\beta$

6. If KB^+ is consistent then $KB \models_c \neg\alpha$ iff $KB \not\models_c \alpha$

For example, $KB \models_c ((p \wedge q) \vee \neg(r \wedge \neg s))$ reduces to either both $KB \models_c p$ and $KB \models_c q$, or $KB \models_c \neg r$, or $KB \models_c s$.

Defaults

Consistency and Generalized Closed-World Assumption (GCWA)

A consistent KB does not imply that KB^+ is also consistent. For example, if $KB=\{(p \vee q)\}$ then KB^+ contains $\{ (p \vee q), \neg p, \neg q\}$ because $KB \not\models p$ and $KB \not\models q$.

So, KB^+ is not consistent.

Obs. If a KB consists of just atomic sentences (e.g. DirectConnect) or conjunctions of atomic sentences (e.g. $p \wedge q$) or disjunctions of negative literals (e.g. $\neg p \vee \neg q$), then KB^+ is consistent.

One way to preserve consistency is to restrict the application of CWA only to atom that are "uncontroversial" (not like p and q in the example above).

Def. The generalized closed-world assumption is

$KB^* = KB \cup \{\neg p \mid$ for all collections of atoms $q_1, \ldots q_n$, if $KB \models (p \vee q_1 \vee \ldots \vee q_n)$ then $KB \models (q_1 \vee \ldots \vee q_n)\}$.

Defaults

The entailment in GCWA is defined as following:

$$KB \vDash_{GC} \alpha \text{ iff } KB^* \vDash_c \alpha$$

Under GCWA, we will not assume that p is false if there is an entailed disjunction of atoms including p that cannot be reduced to a smaller entailed disjunction not involving p.

For example, if $KB=\{(p \vee q)\}$ then $KB \vDash (p \vee q)$ but $KB \nvDash q$, so $\neg p \notin KB^*$ and similarly $\neg q \notin KB^*$.

If we consider an atom r, then $\neg r \in KB^*$ because $KB \vDash (r \vee p \vee q)$ and $KB \vDash (p \vee q)$.

Defaults

An example of interpretation: we know that there is a direct flight from Cleveland to Dallas or Houston. As a result, we know that there is a direct flight from Cleveland to Dallas or Houston or Austin.

But because there is a flight to one of the first two cities, under GCWA we will assume that there is no flight to Austin.

$KB \vDash (DirectConnect(cleveland,dallas) \vee DirectConnect(cleveland,houston))$

=>

$KB \vDash (DirectConnect(cleveland,dallas) \vee DirectConnect(cleveland,houston) \vee DirectConnect(cleveland,austin))$

=>

$\neg DirectConnect(cleveland,austin) \in KB^*$.

Defaults

Entailments in GCWA are a subset of those in CWA, that is if $\neg p \in KB^*$ then $\neg p \in KB^+$.

If KB has no disjunctive knowledge, then GCWA and CWA are in complete agreement.

Prop. If KB is consistent, then KB^* is consistent.

GCWA is a weaker version of CWA that agrees with CWA in the absence of disjunctions, but remains consistent in the presence of disjunctions.

Defaults

Quantifiers and Domain Closure

Let us assume that the representation language contains the predicate DirectConnect and the constants $c_1,\ldots,c_n$ and another smallTown.

If KB contains only atomic sentences of the form DirectConnect(c_i,c_j), then for any pair of constants c_i and c_j either DirectConnect(c_i,c_j) or ¬DirectConnect(c_i,c_j) is in KB^+.

Also, ¬DirectConnect(c_j,smallTown) is in KB^+ for every c_j

If we consider the query ¬∃xDirectConnect(x,smallTown), under CWA neither this query nor its negation is entailed. CWA excludes any of the named cities $c_1,\ldots,c_n$ flying to SmallTown, but it does not exclude other unnamed city doing so.

The easiest way to overcome this problem is to assume that the named constants are the only individuals of interest.

Defaults

Def. The closed-world assumption with domain-closure is

$$KB^{\diamond} = KB^{+} \cup \{\forall x.[x=c_1 \vee \ldots \vee x=c_n]\},$$

where $c_1,\ldots,c_n$ are all the constant symbols appearing in KB. The entailment in CWA with domain-closure is defined as following:

$$KB \vDash_{CD} \alpha \text{ iff } KB^{\diamond} \vDash \alpha \text{ .}$$

The main properties are:

$$KB \vDash_{CD} \forall x.\alpha \text{ iff } KB \vDash_{CD} \alpha_c^x \text{ for every constant c appearing in KB.}$$

$$KB \vDash_{CD} \exists x.\alpha \text{ iff } KB \vDash_{CD} \alpha_c^x \text{ for some constant c appearing in KB.}$$

Compared to CWA, in $KB^{\diamond}$ we make the additional assumption that no other objects exist apart from the named constants.

Now, under CWA with domain-closure, our query

$\neg\exists x DirectConnect(x,smallTown)$

is entailed.

Obs. It is the case that $KB \vDash_{CD} \alpha$ or $KB \vDash_{CD} \neg\alpha$ for any α (with or without quantifiers).