

Computational Logic

Description Logic: Querying Aboxes

Ralf Moeller

Hamburg Univ. of Technology

Acknowledgements

- Slides 5–12 by Michael Wessel (STS)

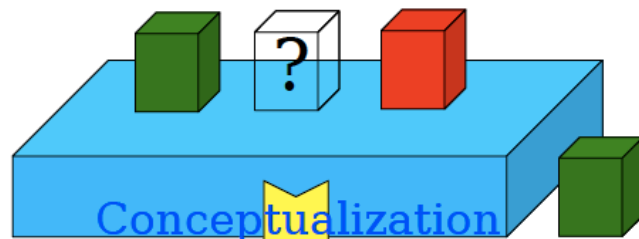
Problem Solving with DLs

- Conceptualization (abstraction)
 - ◆ Define basic notions (“ontology”)
- Formalization
 - ◆ Develop knowledge base
 - ◆ Knowledge might be indefinite
- Reduce application problem to formal decision problem(s)
 - ◆ Use DL reasoning system

Research @ STS

- Description logic inference systems
 - ◆ Racer (www.racer-systems.com)
- Tbox and Abox reasoning
- Expressive Abox query language
 - ◆ nRQL (new Racer Query Language)
 - (read “niracle” and hear “miracle”)
- Applications
 - ◆ Agent-based computing
 - ◆ High-level Computer Vision/Text understanding
 - ◆ Information Extraction / Retrieval

Querying Aboxes (1)

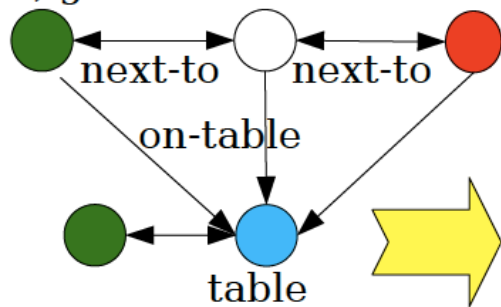


Conceptualization
(Abstraction)

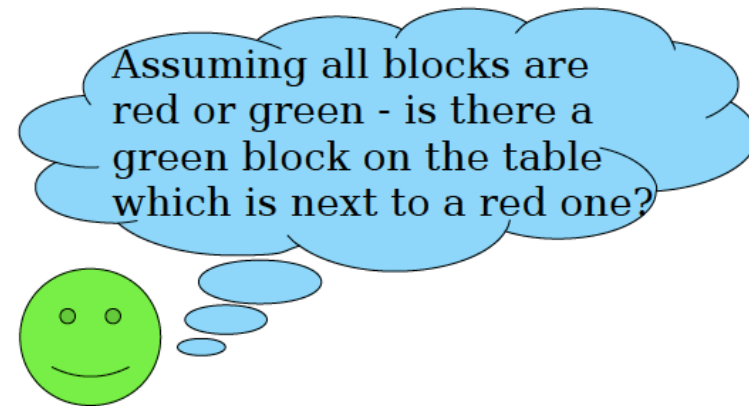
on-table, next-to, block, green, red, ...

Formalization

block, green



Problem Solving



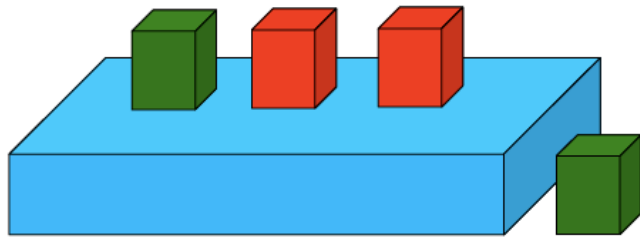
$$\{ block \sqsubseteq red \sqcup green, next_to \doteq next_to^{-1} \}$$

$$\{ t : table, lb : green \sqcap block, rb : red \sqcap block, \\ mb : block, ob : green \sqcap block, \\ (lb, t) : on_table, (ml, t) : on_table, (rb, t) : on_table, \\ (gb, ob) : next_to, (ob, rb) : next_to \}$$

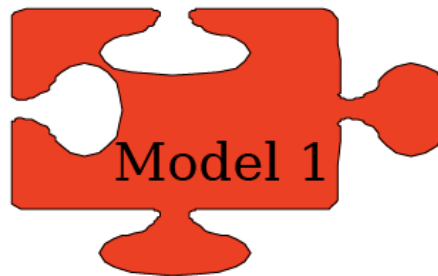
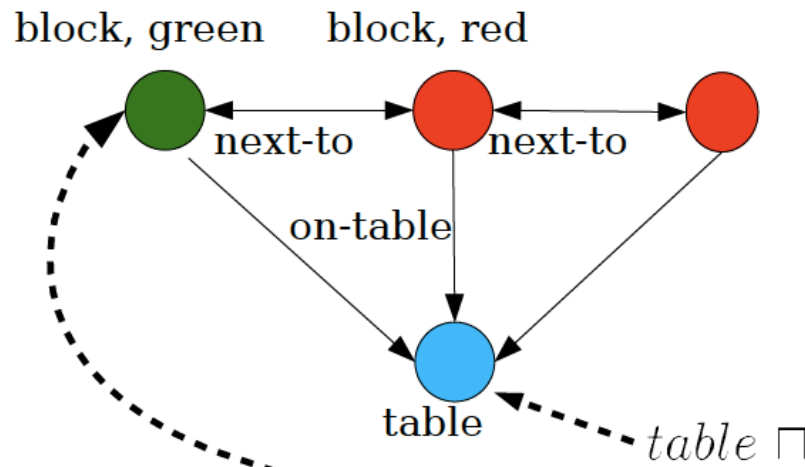
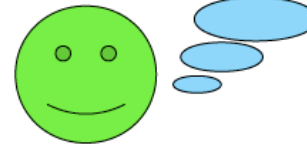
Ask for instances of the concept

$$table \sqcap \\ \exists on_table^{-1}. (block \sqcap green \sqcap \\ \exists next_to.(red \sqcap block))$$

Querying Aboxes (2)

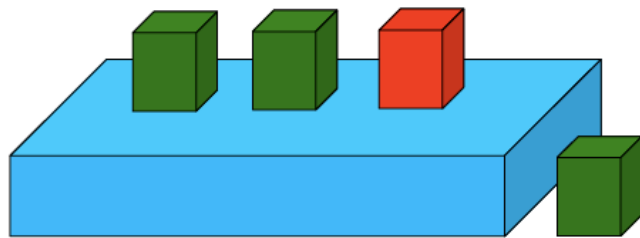


There are two possibilities. If the middle block is red, then the green left block is next to a red one. But...

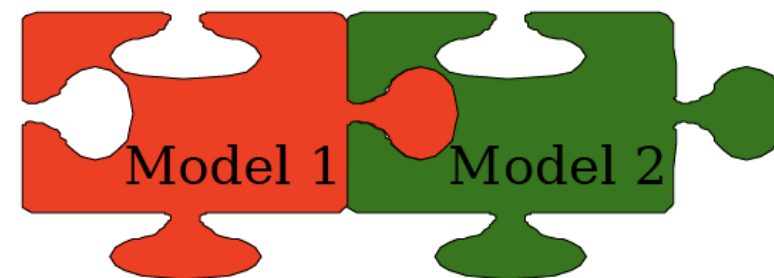
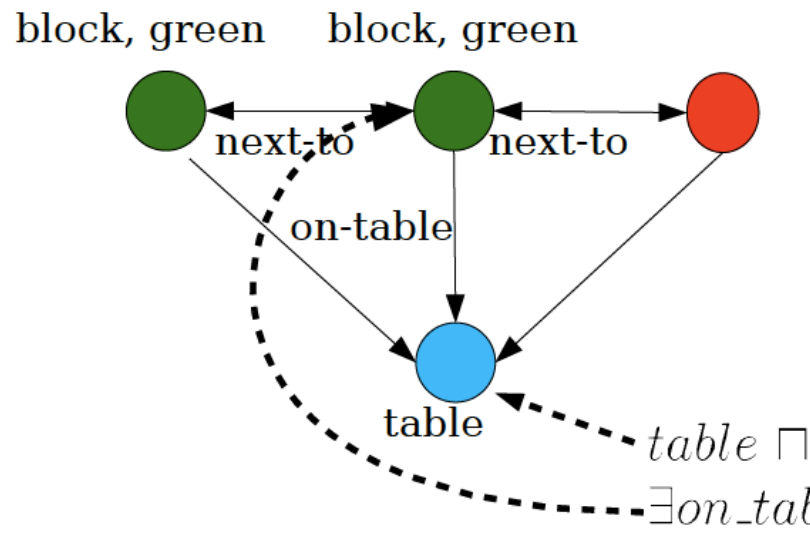
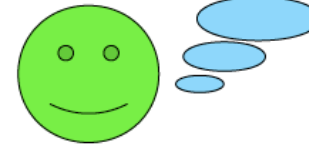


$$\exists on_table^{-1}. (block \sqcap green \sqcap \exists next_to.(red \sqcap block))$$

Querying Aboxes (3)

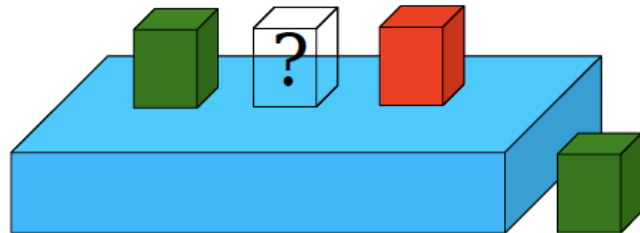


... if the middle block is green, then it is also next to the right Block, which is red. So, yes, there ALWAYS EXISTS such a block on the table!



$$\exists on_table^{-1}. (block \sqcap green \sqcap \exists next_to.(red \sqcap block))$$

Querying Aboxes (4)



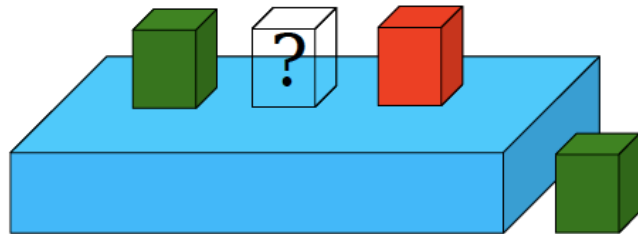
$\text{concept_instances}(\text{table} \sqcap \exists \text{on_table}^{-1}. (\text{block} \sqcap \text{green} \sqcap \exists \text{next_to}. (\text{red} \sqcap \text{block}))) = \{t\}$

However:

$\text{concept_instances}(\text{block} \sqcap \text{green} \sqcap \exists \text{next_to}. (\text{red} \sqcap \text{block}))) = \{\}$

- Unlike SQL, instance retr. queries can cope with
 - incomplete information (case analysis)
 - have to consider ALL models, not only one (rel.DB)
 - only the existence of such a block is entailed

Querying Aboxes (5)

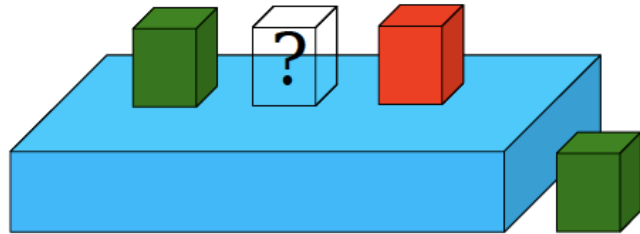


$ans(x) \leftarrow table(x), on_table(y, x),$
 $block(y), green(y),$
 $next_to(y, z), red(z), block(z).$

Answer: $x = t$

- However, no answer for head $ans(y) \leftarrow \dots$
 - distinguished variables in head: binding must hold in ALL models („certain answer“)
 - other variables: treated as existentially quantified

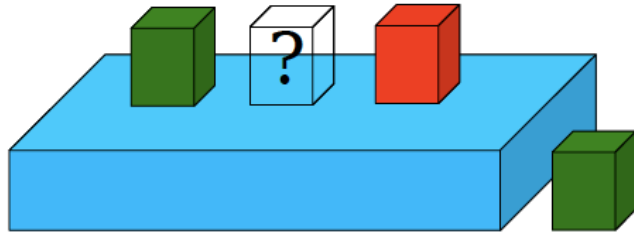
Querying Aboxes (6)


$$\begin{aligned} ans(x) \leftarrow & \text{table}(x), \text{on_table}(y, x), \\ & \text{block}(y), \text{green}(y), \\ & \text{next_to}(y, z), \text{red}(z), \text{block}(z). \end{aligned}$$

Gives no answer in nRQL!

- In grounded conjunctive queries
 - ALL variables are distinguished; a binding is only established iff it holds in ALL models
 - grounding: subst. variables \leftrightarrow entailed assertions

Querying Aboxes (7)



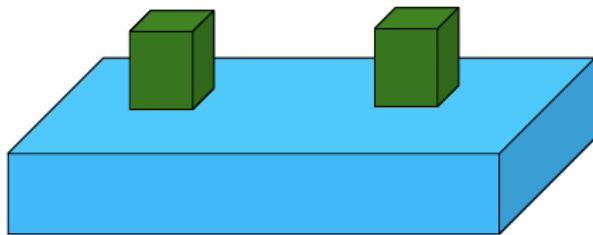
$$ans(x) \leftarrow (table \sqcap \exists on_table^{-1} \dots)(x)$$

This grounded CQ can be used instead of the full CQ (a simple instance retrieval query)

- This „rolling up“ into nested $\exists R. \exists S. \dots$ works only for non-cyclic queries
 - note that variables may introduce coreferences
 - no automatic rolling up in nRQL

Querying Aboxes (8)

- Open World Semantics



$ans(x) \leftarrow (\forall on_table.(block \sqcap green))(x)$

$ans(x) \leftarrow (\leq_2 on_table)(x)$

give no answers

- the model / world is not closed - models with additional red blocks or even non-blocks exist, but can be excluded: $t : \leq_2 on_table$
- the two blocks are the only ones \rightarrow all are green
- DB would conclude all blocks are green (CWA/NAF)

Assumptions

- Unique name assumption
 - ◆ Different individual names denote different domain objects
 - ◆ Usually NOT adopted in DL and first-order settings in general
- Domain closure assumption
 - ◆ The set of individuals is finite
 - ◆ NOT adopted in general
 - Reduces first-order to propositional case

Epistemic First-Order Queries

- Example: Find all blocks for which you cannot prove that they are red or green
 - ◆ $\text{ans}(X) \leftarrow \text{neg } ((\text{Red} \cup \text{Green})(X))$
- Different from
 - ◆ $\text{ans}(X) \leftarrow (\text{not } (\text{Red} \cup \text{Green}))(X)$
- Grounded semantics for neg:
 - ◆ Complement w.r.t. set of individuals

Deduction

- DLs use OWA
- Aboxes **describe** data
- Tboxes (axioms) used for query answering
(\rightarrow more answers)
 - ◆ Axioms (or KBs) “generate data”
- Query language
 - ◆ (Unions of) conjunctive queries
 - ◆ Epistemic first-order queries (grounded)
- Compute what is true (satisfied) in ALL models (\rightarrow Deduction)

Knowledge Bases Seen as Data Generators

- Ontology (in the narrow sense = KB):

*(define-role controlledBy
:domain MortgageLender :range Bank)*

(implies Bank MortgageLender)

*(implies MortgageLender
(at-least 1 controlledBy Bank))*

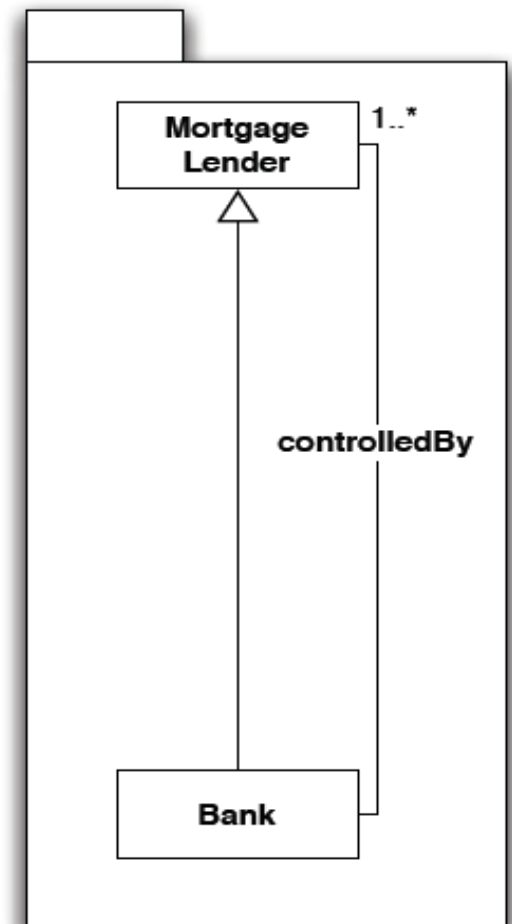
(related Halifax LeedsBS controlledBy)

(instance RBS Bank)

- Query:

`ans(X) <- controlledBy(X, Y)`

`X=Halifax, X=RBS, X=LeedsBS`



controlledBy	
Halifax	Leeds BS

Model Checking

- DBs use CWA/NAF (negation as failure)
- Data = Abox = Relational structure
- Query = Open FO formula (free variables existentially quantified)
- Given the relational structure, find bindings for variables such that query formula is satisfied in the structure
- Check whether structure is a model for the query formula \rightarrow Model Checking

Outlook

- Which approach is better?
 - ◆ Depends on the application
 - ◆ If indefinite knowledge is to be represented, use deduction
- How can we compare different problems?
 - ◆ Which is less computationally-expensive
 - ◆ See next semesters