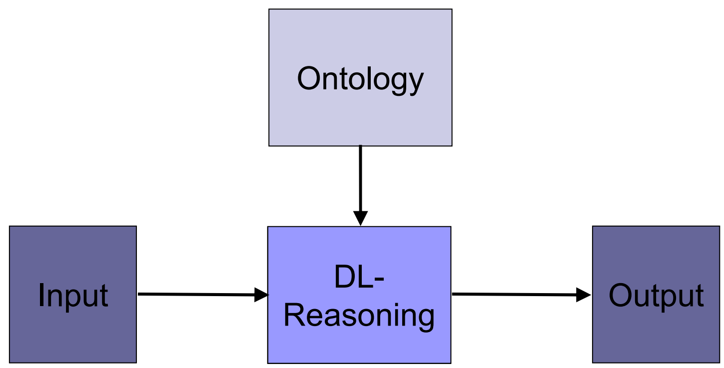
Part D
Approximate reasoning on tableaux

Holger Wache

Overview of the Course

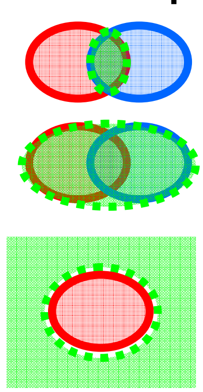
A. Semantic Web in general and OWL syntax
B. OWL Semantics (DLs) and tableaux reasoning
C. Approximate reasoning in general
D. Approximate reasoning on tableaux
E. Approximate resolution for OWL

Semantic Web Systems in General



Description Logics

- **C** ⊓ **D**
- **C** ⊔ **D**
- ¬ **C**
- ∃ R.C
- …



C := **D**

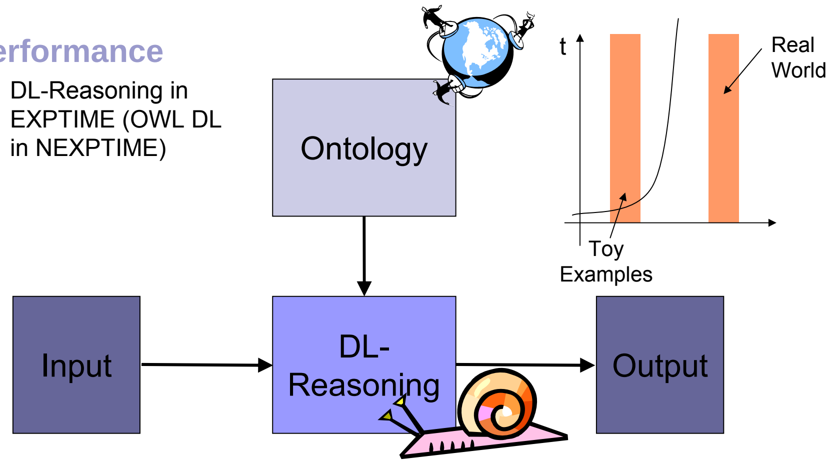
- Human := Man ⊔ Woman
- Man := ¬ Woman
- Father := Man ⊓ ∃ hasChild.Human

C ⊑ **D**

- Father ⊑ Human
- Father ⊑ Woman

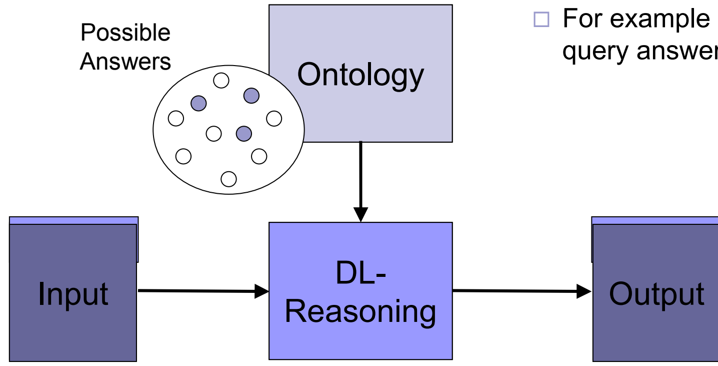
Problem

- **Performance**
  - DL-Reasoning in EXPTIME (OWL DL in NEXPTIME)

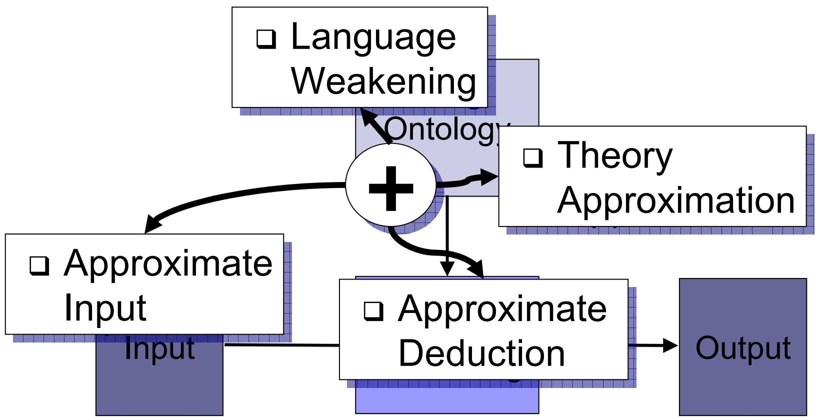


Problem

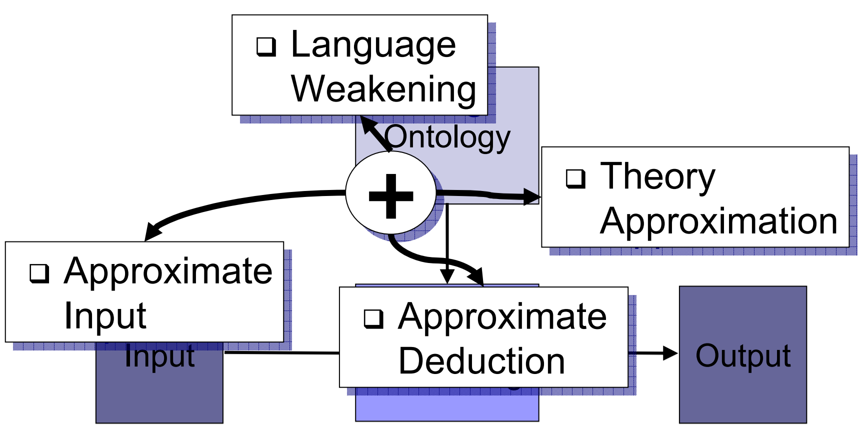
- **Robustness**
  - For example during query answering



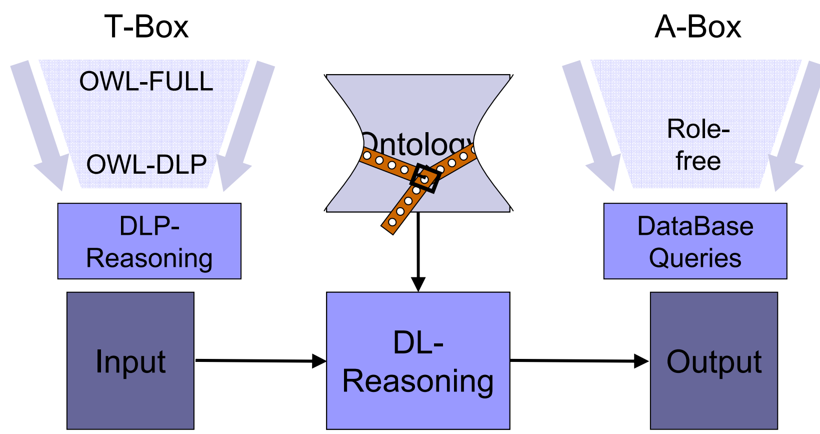
Approximation Approaches



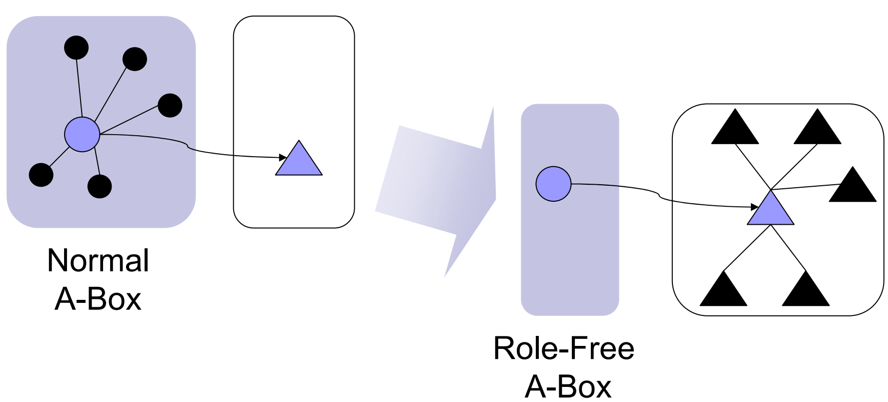
Approximation Approaches



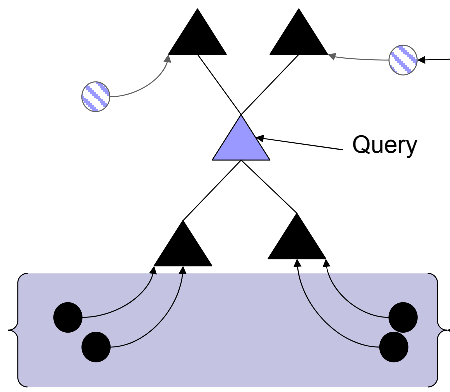
Approximation through Language Weakening



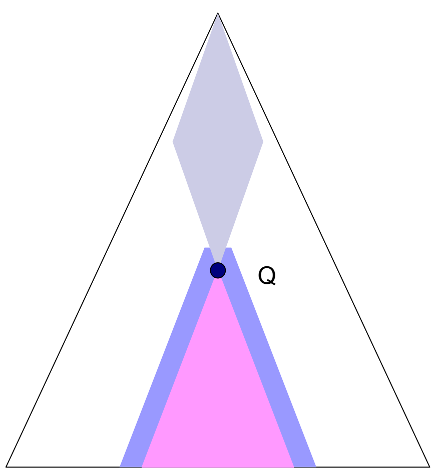
Approximation on A-Boxes



Querying in Role-Free A-Boxes

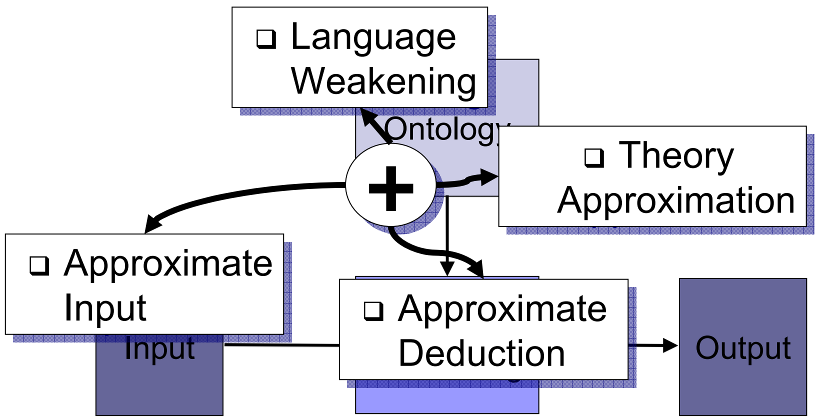


Instance retrieval



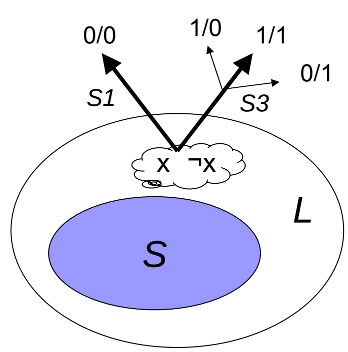
- For role-free A-Boxes
  $a{:}Q \leftrightarrow I_a \sqsubseteq Q$; I_a concept description of instance *a*
- Try to replace DL reasoning as much as possible with (scalable) DB retrieval
  - Classify *Q*
    [DL REASONING]
  - For all $C \sqsubseteq Q$: $a{:}C \rightarrow a{:}Q$
    [DATABASE RETRIEVAL]
  - If needed check border instances
    [DL REASONING]

Approximation Approaches

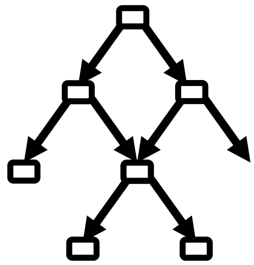


Approximate Entailment by Cadoli-Schaerf

- *S-1*-entailment: interpret everything outside of *S* as *false*
- *S-3*-entailment: interpret everything outside of *S* as *true* (or normal)



Cadoli-Schaerf for DL



$\models^S$

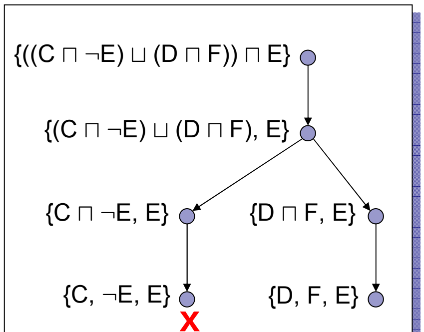
- Description Logic is subset of First Order Logic
- Cadoli Schaerf can directly be applied
- Needs to implement a specialized reasoner ….

Approximate Reasoning

Description Logic (DL) + Tableaux Reasoning

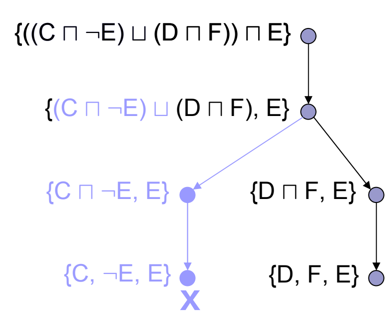
Same effect when query is modified

Approximate Reasoning through modification of the query



Approximating Inference Rule

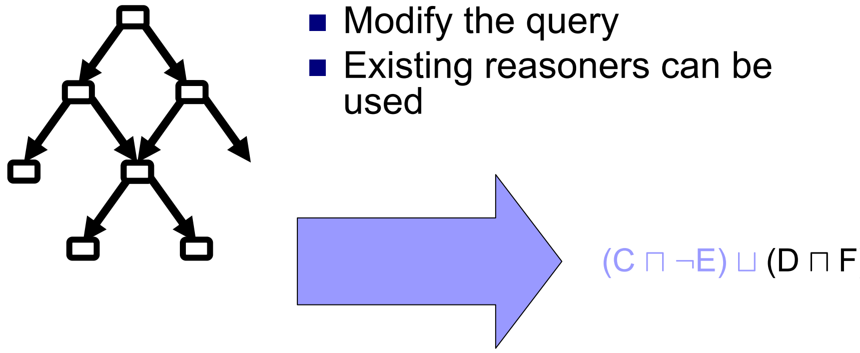
OR



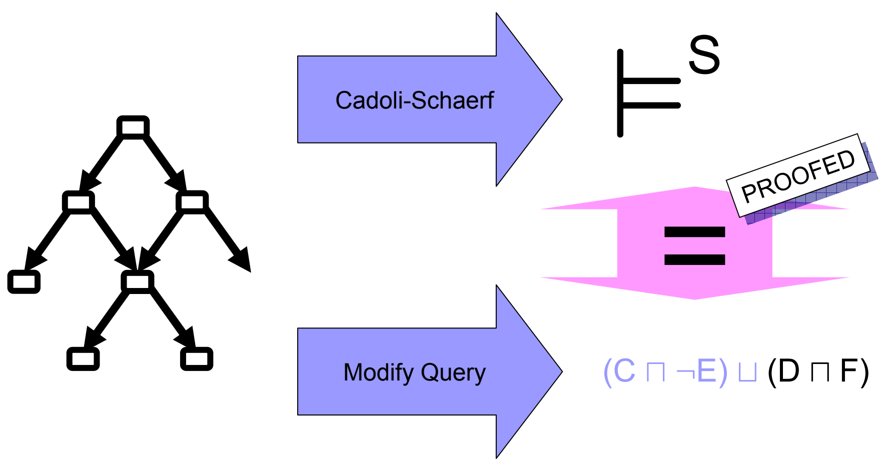
Approximating Query

Cadoli-Schaerf for DL

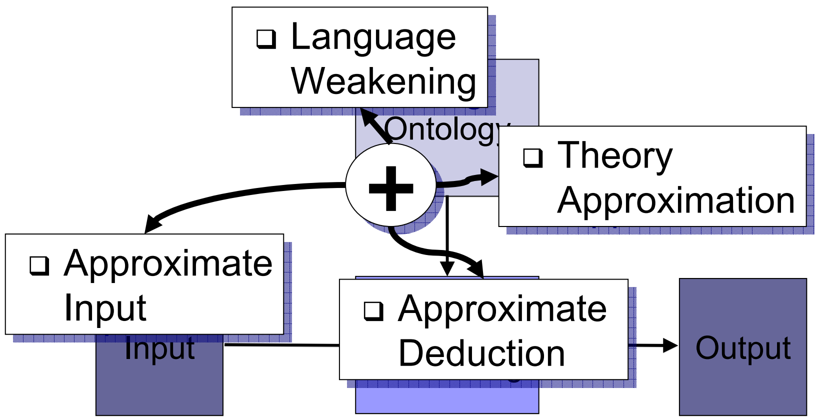
- Instead of using a different entailment
- Modify the query
- Existing reasoners can be used



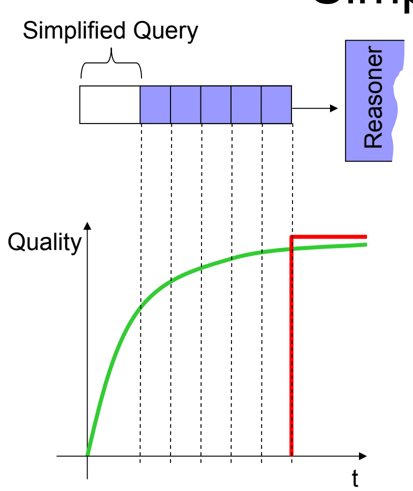
Cadoli-Schaerf for DL



Approximation Approaches



Approximate Deduction through Simplification



- Simplify query
- Simple query ⇒ fast query answering
- Simple query ⇒ approximated answers
- Continuously complete query
- Anytime behavior

How to simplify?

First Idea:
Omit some parts (e.g. Φ, Ψ)

$Q^I \overset{?}{\longleftrightarrow} Q^I$

$Q^I \subseteq Q^I$

Query = … $\sqcap$ $\cancel{\Phi}$ $\sqcap$ … $\sqcap$ (…$\sqcup$ $\cancel{\Psi}$ $\sqcup$ …)

$Q^I \subseteq Q^I$

How to simplify? (II)

Second Idea:
Rewrite some parts (e.g. Φ, Ψ)

$Q^I \subseteq Q^I$

$Q^I \subseteq Q^I$ $Q^I \subseteq Q^I$

Query = … $\sqcap$ $\top$ $\sqcap$ … $\sqcap$ (…$\sqcup$ $\top$ $\sqcup$ …)

$$\phi \mapsto \psi$$

Cadoli-Schaerf Approximation for DLs

$$
\begin{array}{l}
C_i^{\top} : \exists R.C \mapsto \top \\
C_i^{\perp} : \exists R.C \mapsto \perp
\end{array}
$$

- Well-founded theory with (theoretical) nice results
- BUT:How to increase the complexity?

S3-Approximation for Description Logics (ALE)

$$(\exists \text{friend.tall}) \sqcap \forall \text{friend.}((\forall \text{friend.doctor}) \sqcap \exists \text{friend.}\neg\text{doctor}).$$

- Levels i = nested quantifiers

L_1 … $\exists$friend.¬doctor

L_0 ($\exists$friend.tall) …).

- $C_i^{\top}$ = Omm… …ater or equal level i

$C^{\top}_{Level}$ is unsatisfiable

$\Rightarrow$ C is unsatisfiable

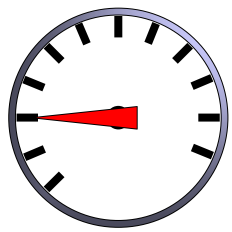
$\exists R.C \mapsto \top$

$C_0^{\top}$ $\top$ … friend.doctor) $\sqcap \top$)

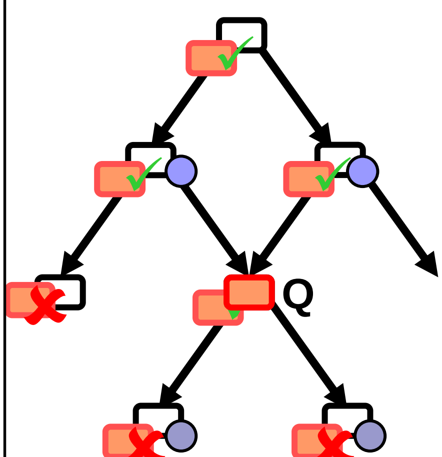
$C_1^{\top}$ $(\exists \text{friend.tall}) \sqcap \forall \text{friend.}((\forall \text{friend.doctor}) \sqcap \top)$

$C_2^{\top}$ $(\exists \text{friend.tall}) \sqcap \forall \text{friend.}((\forall \text{friend.doctor}) \sqcap \exists \text{friend.}\neg\text{doctor}).$

Application

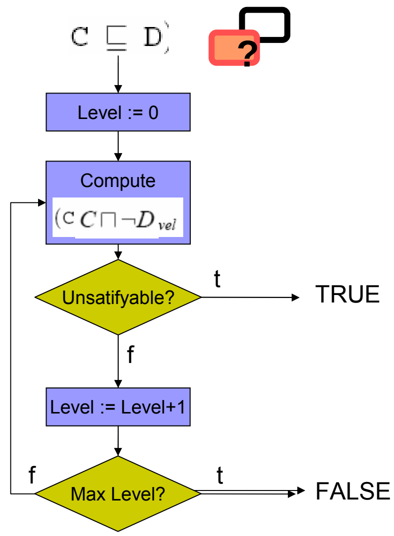


Application: Classification



- Central process
  Classify Query Q
- Contained in
  - Generating the subsumption hierarchy
  - Instance Retrieval
- Generates a sequence of subsumption tests

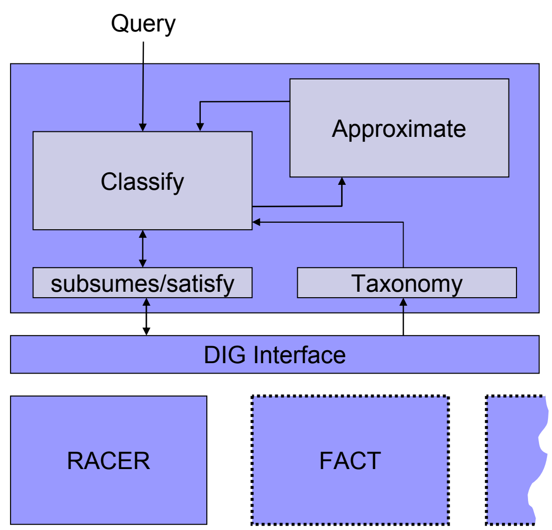
Approximating the Classification



- *C is subsumed by D* $\Leftrightarrow$
  $C \sqcap \neg D$ *is unsatisfiable*
- Cadoli-Schaerf ensures:

$$(C \sqcap \neg D)^{\top}_{Level} \text{ is unsatisfiable}$$
$$\Rightarrow (C \sqcap \neg D) \text{ is unsatisfiable}$$

Implementation



Mixed Results: Classifying in TAMBIS

- Application: Classification of Concepts
  $\Rightarrow$ sequence of subsumption test: $C \sqsubseteq D$

≈0 ≈24 ≈279 ≈0

		normal			$C_i^\perp$			$C_i^\top$			$C_i^\perp \& C_i^\top$	
		true	false		true	false		true	false		true	false
Tambis (16)				$C_0^\perp$	157	32	$C_0^\top$	8	181	$C_0^\perp$	157	32
				$C_1^\perp$			$C_1^\top$			$C_0^\top$	8	149
	N	24	279	N			N			N		

$(C \not\sqsubseteq D)_i^\perp \;\Rightarrow\; C \not\sqsubseteq D$

$(C \sqsubseteq D)_i^\top \;\Rightarrow\; C \sqsubseteq D$

$(C \sqcap \neg D)_i^\perp$ is satisfiable
$\Rightarrow (C \sqcap \neg D)$ is satisfiable

$(C \sqcap \neg D)_i^\top$ is unsatisfiable
$\Rightarrow (C \sqcap \neg D)$ is unsatisfiable

Further Results

		normal		$C_i^\perp$		$C_i^\top$		$C_i^\perp \& C_i^\top$	
		true	false	true	false	true	false	true	false
Dolce (10)	$C_0^\perp$	-	-	0	0	-	-	0	0
	$C_0^\top$	-	-	-	-	0	0	0	0
	normal	10	113	10	113	10	113	10	113
Galen (10)	$C_0^\perp$	-	-	0	0	-	-	0	0
	$C_0^\top$	-	-	-	-	0	0	0	0
	normal	10	12190	10	12190	10	12190	10	12190
Monet (10)	$C_0^\perp$	-	-	0	0	-	-	0	0
	$C_0^\top$	-	-	-	-	0	0	0	0
	normal	20	656	20	656	20	656	20	656
MadCow (10)	$C_0^\perp$	-	-	145	0	-	-	145	0
	$C_0^\top$	-	-	-	-	5	140	5	140
	normal	66	152	66	152	61	152	61	152
Wine (10)	$C_0^\perp$	-	-	228	1	-	-	228	1
	$C_0^\top$	-	-	-	-	6	223	6	222
	normal	33	252	33	251	27	252	27	251

Problem: Term Collapsing

$C \sqsubseteq D \;\leftrightarrow C \sqcap \neg D$ is unsatisfiable

Queries have very often this structure

$$\text{Query} = \underbrace{\ldots \sqcap \Phi \sqcap \ldots}_{} \sqcap \underbrace{(\ldots \sqcup \Psi \sqcup \ldots)}_{}$$

- Term C
  - to be classified;
  - very often conjunction of subterms
  - e.g. conjunctive queries
- Term D
  - From the subsumption hierarchy
  - Very often also conjunction of subterms

Problem: Term Collapsing

$$\text{Query} = \underbrace{\ldots \sqcap \top \sqcap \ldots}_{}$$

- Term C
  - to be classified;
  - very often conjunction of subterms
  - e.g. conjunctive queries

Problem: Term Collapsing

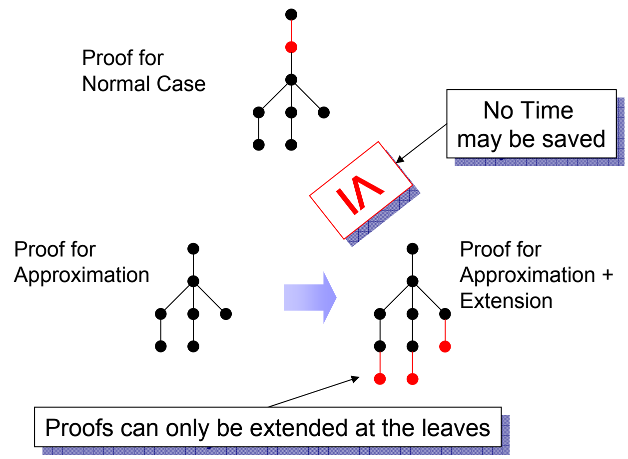
Query = $\perp$

Classifying in TAMBIS (IV)

	normal			$C_i^\perp$			$C_i^\top$			$C_i^\perp \& C_i^\top$		
		true	**false**		**true**	**false**		**true**	**false**		**true**	**false**
Tambis (16)				$C_0^\perp$	157	32	$C_0^\top$	8	181	$C_0^\perp$	157	32
				$C_1^\perp$	0	0	$C_1^\top$	0	0	$C_0^\top$	8	149
	N	24	279	N	24	247	N	16	279	N	16	247

Term Collapsing: 157 = 100% 65 = 35,9% 190 = 62,1%

Approximation and Proof Reuse



Lessons learned

$$\phi \longmapsto \psi$$

- Avoid Term Collapsing
  - Replace ψ with something else than $\top$ or $\bot$
- Find better places to rewrite
  - Ontology-adapted ϕ?
  - Heuristic to select the rewriting places?

Special Case: Instance Retrieval

Focused Case: Instance Retrieval

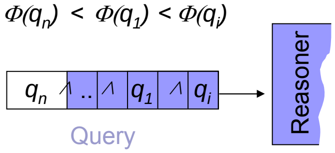
- Find all instances *a* which belongs to a query *Q*: *a:Q*
- Tool *InstanceStore:*
  - Try to replace DL reasoning as much as possible with (scalable) DB retrieval
  - Only applyable to role-free A-Boxes
    $a{:}Q \leftrightarrow I_a \sqsubseteq Q$; I_a concept description of instance a
- Boolean Conjunctive Queries
  - $q_1 \wedge \ldots \wedge q_n$, where $q_1,\ldots,q_n$ are of the form $x{:}C$ or $\langle x,y\rangle{:}R$
  - Restrict to those which can be converted to a concept expression *C*

New Approximation Method: Heuristic Ordering of Conjuncts

$q_1 \wedge .. \wedge q_i \wedge .. \wedge q_n$

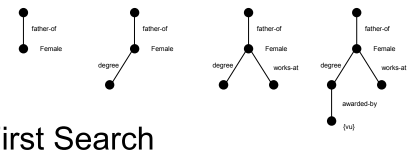
$\Phi(q_1)$ $\Phi(q_i)$ $\Phi(q_n)$

$\Phi(q_n) < \Phi(q_1) < \Phi(q_i)$



- Compute a ranking value for each conjunct
  $$\Phi(q_i) : C \mapsto \mathbb{R}$$
- Order the conjuncts $q_1, \wedge, q_n$ according to their value
- Complete approximated query with more and more expensive conjuncts

How to order conjuncts?



- Recall
  - Node expansion
  - Depth-First and Breadth-First Search
- Furthermore
  - According to the needed computation time for each conjunction
    - Estimate the computation time a priori
  - According to the possible search space reduction
    - Prefer conjuncts which eliminate a lot of instances

How to estimate the computation costs

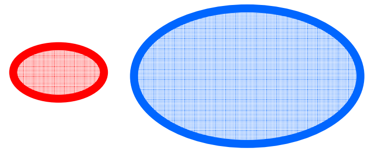
$$
\begin{aligned}
\Phi(A) &= 1\\
\Phi(\neg A) &= 0\\
\Phi(C \sqcap D) &= 2 + \Phi(C) + \Phi(D)\\
\Phi(C \sqcup D) &= \phi + 2 + \Phi(C) + \Phi(D)\\
\Phi(\exists\, R.C) &= 2 + \Phi(C)\\
\Phi(\forall\, R.C) &= n + n \cdot \Phi(C)
\end{aligned}
$$

where ϕ is the current value of $\Phi(E)$

where n is the number of existential quantifiers in E

Effects of Cadoli-Schaerf for Subsumption

$C \sqsubseteq D$



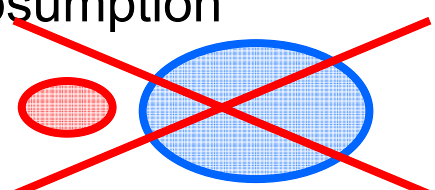
Semantics

$(C \sqsubseteq D)^{\perp}$

$$C \sqsubseteq D \quad \leftrightarrow \quad \not\models \; C \sqcap \neg D$$

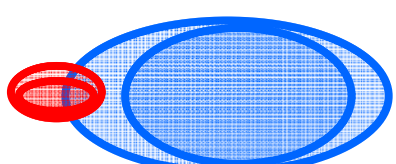
Effects of Cadoli-Schaerf for Subsumption

$C \not\sqsubseteq D$



Semantics

$(C \not\sqsubseteq D)^{\perp}$

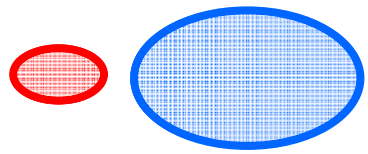


$$(C \sqsubseteq D)^{\perp} \leftrightarrow \not\models (C \sqcap \neg D)^{\perp}$$

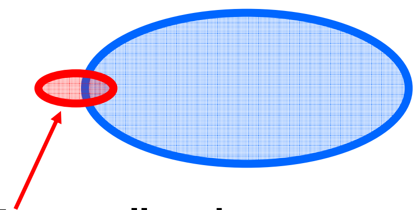
$$\downarrow C^{\perp} \quad \neg D^{\perp} \updownarrow$$

Effects of CS for Subsumption: Term Collapsing

$C \not\sqsubseteq D$

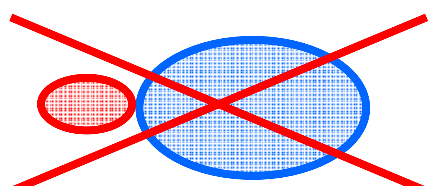


$(C \not\sqsubseteq D)^{\perp}$



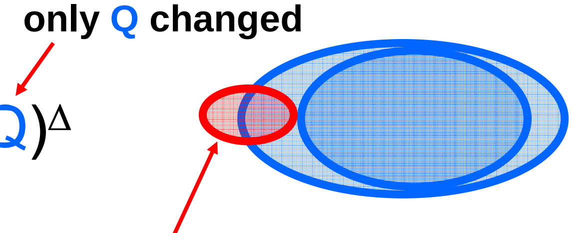
Effects of new Approximation

$(C_a \not\sqsubseteq Q)$

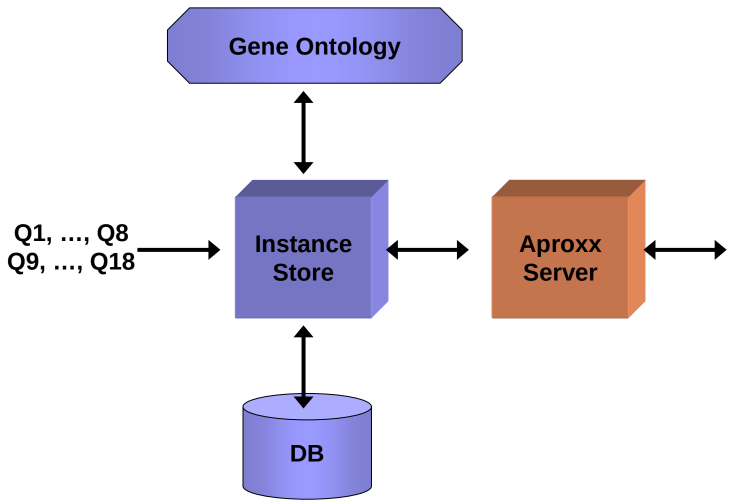


only Q changed

$(C_a \not\sqsubseteq Q)^{\Delta}$



Evaluation Scenario



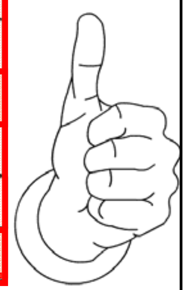
Results: Subsumption tests

More Levels

	normal			$c^{\top}$						c^{Δ}		
		true	false		true	false		true	false		true	false
Q2										$L0$	20	0
										$L1$	20	0
				$L0$	0	19	$L0$	19	0	$L2$	9	11
	normal	9	11	normal	9	11	normal	9	11	normal	9	0
Q8										$L0$	607	0
				$L0$	0	606	$L0$	606	0	$L1$	10	597
	normal	10	597	normal	10	597	normal	10	597	normal	10	0
Q12				$L0$	0	7871	$L0$	7871	0	$L0$	15	7856
	normal	15	7856	normal	15	7856	normal	15	7856	normal	15	0
Q14										$L0$	408	0
										$L1$	5	403
				$L0$	0	407	$L0$	407	0	$L2$	5	0
	normal	5	403	normal	5	403	normal	5	403	normal	5	0
Q15				$L0$	0	6693	$L0$	6693	0	$L0$	6693	0
	normal	46	6647	normal	46	6647	normal	46	6647	normal	46	6647
Q17				$L0$	0	7873	$L0$	7873	0	$L0$	1	7872
	normal	1	7872	normal	1	7872	normal	1	7872	normal	1	0

Results: Time

	normal	$c^{\top}$	$c^{\perp}$	c^{Δ}
Q2	175	348	299	547
Q8	5373	8383	7753	9912
Q12	[illegible]	[illegible]	[illegible]	56478
Q14	[illegible]	[illegible]	[illegible]	7391
Q15	61560	90847	83714	114162
Q17	113289	[illegible]	[illegible]	93074

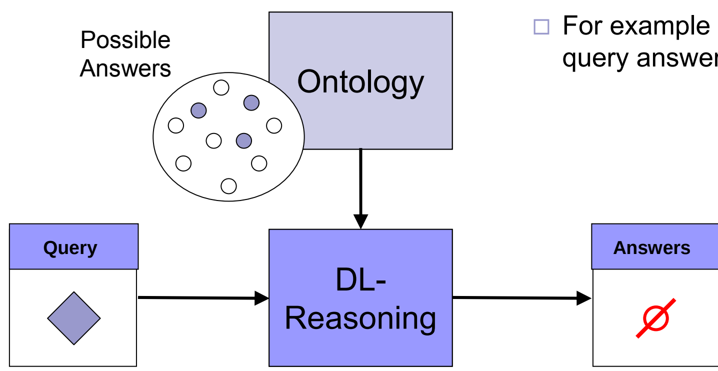


Summary

- Achieving scalability through approximation and modularization/distribution
- S1&S3 approximation seems not to work in practical situations
- Heuristic approximation seems to be better
  - Performance improvements (only) in restricted cases?!
- What's about Robustness?

Problems tackled in KWEB

- **Robustness**
  - For example during query answering



WP1.1 Use-Case 1 “Human Resources” and WP2.1

Job request:
Semantic Web professional with 10 years business experiences, solid background in Trust and Security, finished studying with a excellent grade, not married, and not older than 21 years

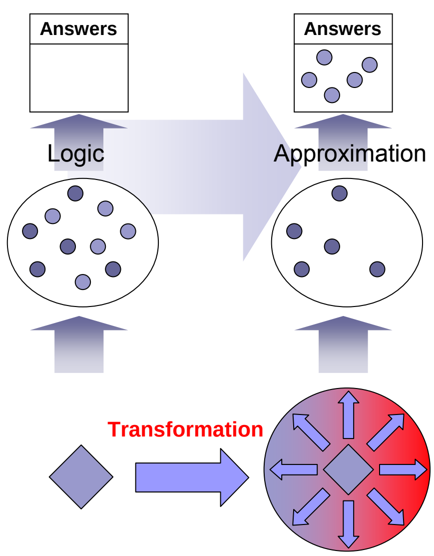
- Collaborative Answering for supporting users
- Approximation as a basic technology
- Feature: Distributed reasoning possible
- Feature: Matching of different ontologies

L3S Use-Case “E-learning” (KWEB portal: REASE)

Ressource request (somehow completed):
I’m looking for a lecture with subject “approximation for the Semantic Web” hold in Slovak. I have experiences in propositional logics and I followed a course about Timbuktu.

Ressource:
Lecture with “approximation for the Semantic Web” in the description (only a part of that talks) hold in Danish. It needs experiences in description logics and OWL.

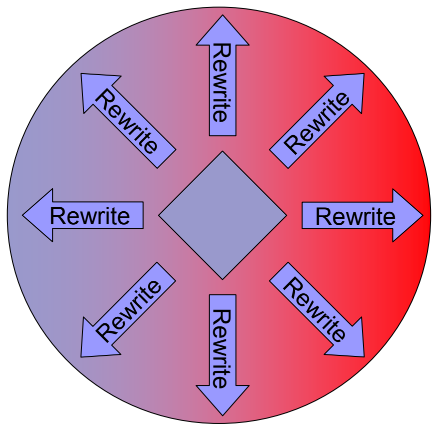
Collaborative Query Answering



- **Assisting users in formulating their query**
- **Technically: Query reformulation**
- **How to transform?**
- **Technologically similar to approximation**

Idea: Rewriting rules

Domain-Centric



User-Centric

Rewrite Rule

Pattern

Replaced-By

Condition

- Three parts
- Invocation:
  1. If “Pattern” matches (part-of) query, …
  2. … replace matched pattern with “replaced-by” …
  3. … if “Condition” is satisfied

Rewriting rules: Example

```
PATTERN
  {Resource} subject {Subject}
  WHERE
  Subject Like Value^^xsd:string,
```

```
REPLACE-BY
  {Resource} subject {Subject},
  {Resource} title {TMPTitle}
  WHERE
  TMPTitle Like NEWTitle
```

```
WITH
  NEWTitle = concat("*",concat(Value,"*"))
```

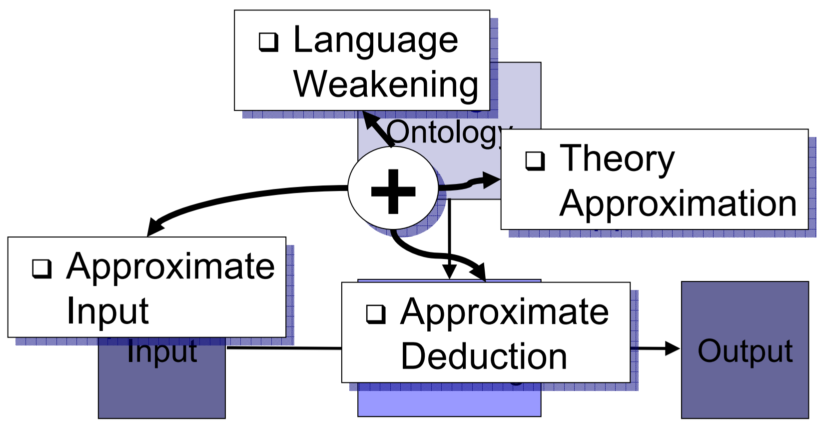
Rewrite Rule Example Applied

```
SELECT * FROM
  {Resource} subject {Subject},
  {Resource} title {Title},
  …
WHERE
  Subject Like "inference engines",
  …
```

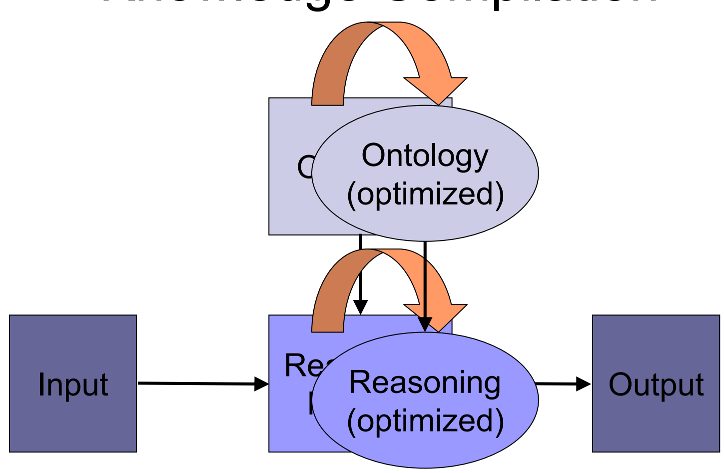
```
SELECT * FROM
  {Resource} subject {Subject},
  {Resource} title {TMPTitle},

  {Resource} title {Title},
  …
WHERE
  TMPTitle Like "*inference engines*",
  …
```

Approximation Approaches



Approximation through Knowledge Compilation



Thank you for your attention!