Epistemic First-Order Queries over Description Logic Knowledge Bases

Giuseppe De Giacomo

Università di Roma "La Sapienza"

Joint work with
Diego Calvanese
Domenico Lembo
Maurizio Lenzerini
Riccardo Rosati

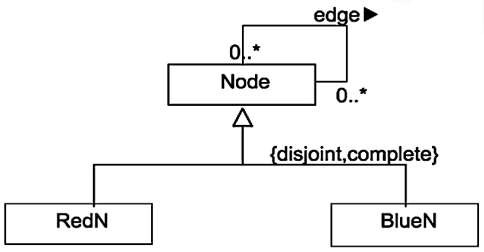
2006 International Workshop on Description Logics
Lake District, U.K., May 30 -- June 1, 2006

Motivation

- Good techniques for doing instance checking are known
  - But DLs are poor query languages [Lenzerini-Schaerf-AAAI'91]
  - Also for most DLs, coNP-hard in data complexity
- Techniques for answering CQs and UCQs are known
  - High complexity for expressive DLs [cf. next talk]
  - In LOGSPACE (as for SQL in DBs) for DL-lite
- What about going beyond UCQs?
  - FOL/SQL queries over KB are undecidable

But, often users expect to have SQL-like query capabilities!!!

Example

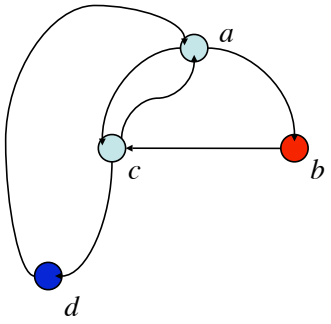


- TBox:

  $\exists\, edge^- \sqsubseteq Node$
  $\exists\, edge \sqsubseteq Node$
  $RedN \sqsubseteq Node$
  $BlueN \sqsubseteq Node$
  $RedN \sqsubseteq \neg BlueN$
  $Node \sqsubseteq RedN \sqcup BlueN$

- ABox:

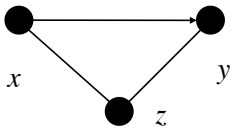
  $edge(a,b)$
  $edge(b,c)$
  $edge(c,a)$
  $edge(c,d)$
  $edge(d,a)$
  $RedN(b)$
  $BlueN(d)$



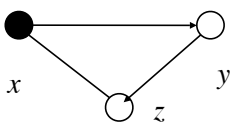
Queries

$q(x) \text{ :- } \exists\, y, z, w.\ edge(x,y) \wedge edge(y,z) \wedge edge(z,w)$

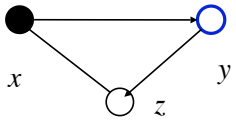
$q(x,y,z) \text{ :- } edge(x,y) \wedge edge(y,z) \wedge edge(z,x)$

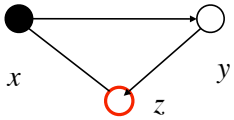


$q(x) \text{ :- } \exists\, y, z.\ edge(x,y) \wedge edge(y,z) \wedge edge(z,x)$



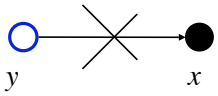
$q(x) \text{ :- } \exists\, y, z.\ edge(x,y) \wedge edge(y,z) \wedge edge(z,x) \wedge (BlueN(y) \vee RedN(z))$



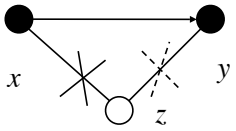


Queries

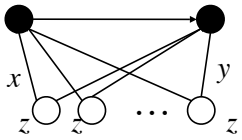
$q(x) :- \exists y.\ BlueN(y) \wedge \neg edge(x,y)$



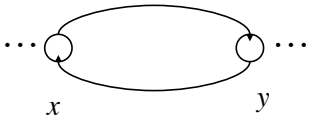
$q(x,y) :- edge(x,y) \wedge \neg\exists z.\ (edge\ (z,x) \wedge edge(z,y))$



$q(x,y) :- edge(x,y) \wedge \forall z.\ (edge\ (z,x) \Rightarrow edge(z,y))$



$q() :- \forall x,y.\ (edge\ (x,y) \Rightarrow edge(y,x))$



An experiment on relational databases

Person

name	birthdate
john	1940
paul	1942
george	1943
richard	null

SQL query:

$q(x) :- \exists b.(Person(x,b) \wedge b = 1940) \vee$
$\exists b.(Person(x,b) \wedge b \neq 1940)$

Answer:

$\{john, paul, george\}$

*What about richard? Since the DBMS **doesn't know** his birthdate, the DBMS can't extablish whether it is equal to 1940 or different from 1940, hence the DBMS skips it!*

Epistemic Query Language (EQL)

- Let KB be a DL KB, intepreted over fixed domain Δ and standard names
- EQL = FOL + epistemic operator (minimal knowledge) over KB

$$\varphi ::= A(t) \mid P(t_1,...,t_n) \mid t_1 = t_2 \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \exists x.\varphi \mid \mathbf{K}\varphi$$

A : concept name in *KB*
P : role/relation name in *KB*
t : constant in *KB* or variable

Cf:

- Levesque's "*Foundations of a Functional Approach to KR*" [– AIJ'84]
- Reiter's "*What should a DB know?*" [– JLP'92]
- Levesque & Lakemeyer's "*The Logic of KBs*" [- Book'01]
- Epistemic operator and DLs [Donini-Lenzerini-Nardi-Schaerf-Nutt-KR'92]

EQL: semantics

- Let KB a KB and φ a EQL formula
- Epistemic interpretation E,w
  - E is the set of **all** models of KB
  - w is **one** such model
- φ true in E,w, written $E,w \models \varphi$

$E,w \models A(c)$	iff	$c \in A^w$
$E,w \models P(c_1,...,c_n)$	iff	$(c_1,...,c_n) \in P^w$
$E,w \models c_1=c_2$	iff	$c_1=c_2$
$E,w \models \neg\varphi$	iff	$E,w \not\models \varphi$
$E,w \models \varphi_1 \wedge \varphi_2$	iff	$E,w \models \varphi_1$ and $E,w \models \varphi_2$
$E,w \models \exists x.\varphi(x)$	iff	$E,w \models \varphi(c)$ for some c
$E,w \models \mathbf{K}\varphi$	iff	$E,v \models \varphi$ for all $v \in E$

EQL: objective and subjective formulas

- Objective formulas
  - no occurrence of ***K***
  - talk about what is true in the world
  - example: $\exists x, y.\ edge(x,y)$
  - $E,w \models \varphi$ reduces to $w \models \varphi$
- Subjective formulas
  - all atoms under the scope of ***K***
  - talk about what is known by the KB
  - example: $\exists x, y.\ \boldsymbol{K}\ edge(x,y)$
  - $E,w \models \varphi$ reduces to $E \models \varphi$
- Non objective and non subjective formulas
  - talk about what is true in world in relation to what is known by the KB
  - example: $\exists x, y.\ edge(x,y) \wedge \boldsymbol{K}\ edge(x,y)$

EQL: knowledge & logical implication

Fundamental property of EQL: minimal knowledge

$$KB \models \varphi \quad \text{iff} \quad KB \models \boldsymbol{K}\varphi$$
$$KB \not\models \varphi \quad \text{iff} \quad KB \models \neg\boldsymbol{K}\varphi$$

In other words:

- $\boldsymbol{K}\varphi$ can be read as φ is logically implied
- $\neg\boldsymbol{K}\varphi$ can be read as φ is not logically implied
  ie $\neg\varphi$ is satisfiable

Example:

$$\boldsymbol{K}edge(a,b) \wedge \boldsymbol{K}edge(b,c) \wedge \boldsymbol{K}edge(c,a)$$

can be read:

- edges $(a,b), (b,c), (c,d)$ are known
- edges $(a,b), (b,c), (c,d)$ are logically implied

EQL: queries

- EQL query:

$$q(x_1,\ldots,x_n) \text{ :- } \varphi(x_1,\ldots,x_n)$$

- Answer:

$$ans(q, KB) = \{ (c_1,\ldots,c_n) \mid KB \models \varphi(c_1,\ldots,c_n), \quad c_i \in \Delta \}$$

EQL: queries - CQs without existential variables

Example [cf.LUBM, Ralph's talk, Bijan's talk]

$$q(x,y,z) \text{ :- } edge(x,y) \wedge edge(y,z) \wedge edge(z,x)$$

is equivalent to *(since* $KB \models \varphi$ *iff* $KB \models \mathbf{K}\varphi$ *)*

$$q(x,y,z) \text{ :- } \mathbf{K}(edge(x,y) \wedge edge(y,z) \wedge edge(z,x))$$

is equivalent to *(since* **K** *distributes over ANDs)*

$$q(x,y,z) \text{ :- } \mathbf{K}edge(x,y) \wedge \mathbf{K}edge(y,z) \wedge \mathbf{K}edge(z,x)$$

EQL-lite($\mathcal{Q}$)

- Restriction on EQL, parametric wrt an objective query language $\mathcal{Q}$

- EQL-lite($\mathcal{Q}$) queries have the form (with α in $\mathcal{Q}$)

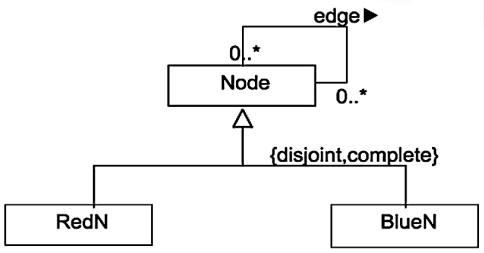
$$\varphi ::= \boldsymbol{K}\alpha \mid t_1 = t_2 \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \exists\, x.\varphi$$

and are domain independent (cf. relational algebra)

Example

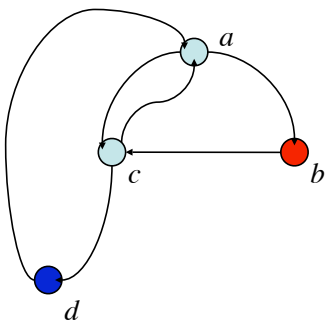
- TBox:

$\exists\, edge^- \sqsubseteq Node$
$\exists\, edge \sqsubseteq Node$
$RedN \sqsubseteq Node$
$BlueN \sqsubseteq Node$
$RedN \sqsubseteq \neg BlueN$
$Node \sqsubseteq RedN \sqcup BlueN$



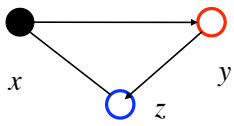
- ABox:

edge(a,b)
edge(b,c)
edge(c,a)
edge(c,d)
edge(d,a)
RedN(b)
BlueN(d)

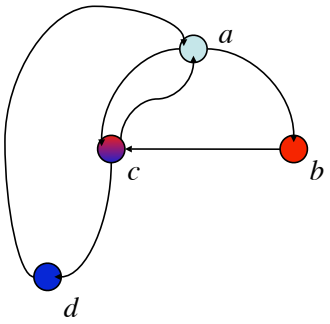


Example

- Query:
  $q(x) :- \exists y, z.\ edge(x,y) \wedge RedN(y) \wedge edge(y,z) \wedge BlueN(z) \wedge edge(z,x)$

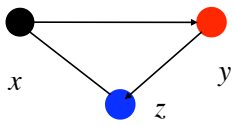


- Answer: $\{a\}$

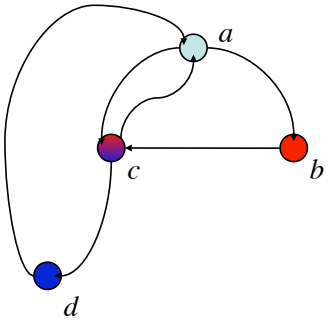


Example

- Query:
  $q(x) :- \exists y, z.\ \mathbf{K}\, edge(x,y) \wedge \mathbf{K}\, RedN(y) \wedge \mathbf{K}\, edge(y,z) \wedge \mathbf{K}\, BlueB(z) \wedge \mathbf{K}\, edge(z,x)$

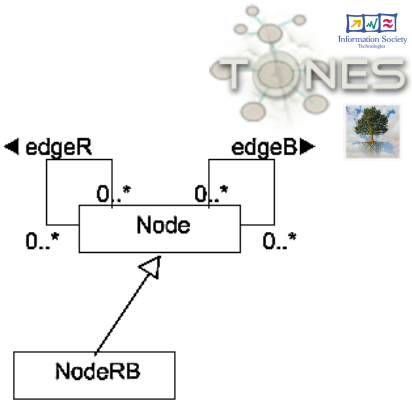


- Answer: $\{\}$

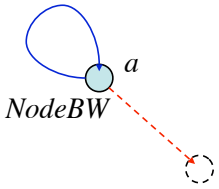


Example

- TBox:
  $\exists\, edgeR^- \sqsubseteq Node$
  $\exists\, edgeR \sqsubseteq Node$
  $\exists\, edgeB^- \sqsubseteq Node$
  $\exists\, edgeB \sqsubseteq Node$
  $NodeRB \sqsubseteq \exists\, edgeR$
  $NodeRB \sqsubseteq \exists\, edgeB$

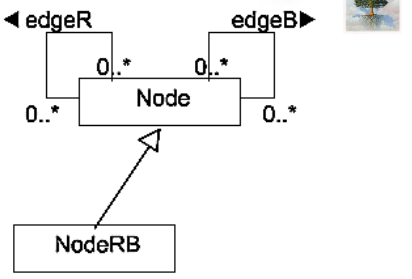


- ABox:
  $edgeB(a,a)$
  $NodeRB(a)$



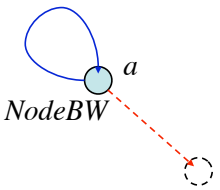
Queries

- Query:
  $q1(x) :- \exists\, y, z, w.\ edgeB(x,y) \wedge edgeR(x,z) \wedge edgeR(y,z)$
  Answer: {a}



- Query:
  $q2(x,y,z) :- edgeB(x,y) \wedge edgeR(x,z) \wedge edgeR(y,z)$
  Answer: {}

- Query:
  $q3(x) :- \exists\, y, z, w.\ \mathbf{K}\, edgeB(x,y) \wedge \mathbf{K}\, edgeR(x,z) \wedge \mathbf{K}\, edgeR(y,z)$
  Answer: {}



EQL-lite($\mathcal{Q}$): main result

- A $\mathcal{Q}$ query α is KB-range restricted iff ans(α,KB) is finite.
- An EQL-lite($\mathcal{Q}$) query is KB-range restricted iff all α appearing in it are KB-range restricted.
- **Thm:** if ans(α,KB) is finite, then it contains only constants occurring in $\mathcal{KB}$.
- **Thm:** Let $\mathcal{KB}$ be a KB expressed in the DL $\mathcal{L}$ and let ***C*** be the data complexity of answering queries in Q over KBs in $\mathcal{L}$, then, answering a $\mathcal{KB}$-range restricted EQL-lite($\mathcal{Q}$) is in LOGSPACE^{C} wrt data complexity.

EQL-lite on concepts/roles in $\mathcal{SHIQ}$ KB

[cf. Ralph's and Bijan's talks]

- $\mathcal{SHIQ}$ concepts and roles $\rightarrow \mathcal{Q}$
- $\mathcal{SHIQ} \rightarrow$ KB
  (or variants)
- Answering $\mathcal{Q}$ queries $\rightarrow$ instance checking which is coNP-complete in data complexity for $\mathcal{SHIQ}$
- Answering EQL-lite queries is LOGSPACE^{coNP}

EQL-lite on UCQ in $\mathcal{ALCQI}$ KB

- UCQs $\rightarrow \mathcal{Q}$
- $\mathcal{ALCQI} \rightarrow$ KB

- Query answering of UCQs in $\mathcal{ALCQI}$ KBs is coNP-complete in data complexity [cf. next talk]

- Answering EQL-lite queries is LOGSPACE$^{\text{coNP}}$

EQL-lite on concepts/roles in $\mathcal{EL}$ KB

- $\mathcal{EL}$ concepts and roles $\rightarrow \mathcal{Q}$
- $\mathcal{EL} \rightarrow$ KB
  (in fact any member of the $\mathcal{EL}$ family)

- Answering $\mathcal{Q}$ queries $\rightarrow$ instance checking, which is PTIME-complete in data complexity for $\mathcal{EL}$

- Answering EQL-lite queries is PTIME-complete

EQL-lite on UCQ in DL-lite KB

- UCQs $\rightarrow \mathcal{Q}$
- DL-lite $\rightarrow$ KB
  (in fact any member of the DL-lite family)
- Answering UCQs in DL-lite is LOGSPACE in data complexity, actually FOL reducible
- Answering EQL-lite queries is LOGSPACE, actually FOL reducible (rewritable in SQL)

Conclusions

- EQL-lite $\approx$ FOL queries for most users
- EQL-lite can be seen as a semantically well characterized approximation of FOL queries
- EQL-lite is based on a controlled use of the epistemic (minimal knowledge) operator
- Jumping from $\mathcal{Q}$ to EQL-lite($\mathcal{Q}$) is (almost) for free
- EQL-lite on UCQs over DL-lite is FOL-reducible (SQL!)
- EQL-lite is very interesting also for modeling constraints over ontologies