

Description Logic Rules

Markus Krötzsch

Sebastian Rudolph

Pascal Hitzler

Institute AIFB
Universität Karlsruhe (TH)
Karlsruhe, Germany

Copyright Markus Krötzsch

Terms of Licensing

Creative Commons
Attribution

<http://creativecommons.org/licenses/by/2.0/de/deed.en>

All redistributions and reuses of this work
must include the following:

“Based on slides by Markus Krötzsch, <http://korrekt.org>”

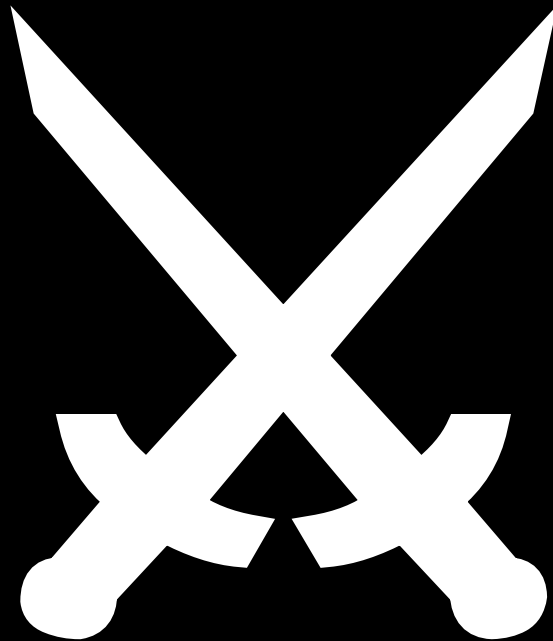
Embedded images are excluded from these conditions

introduction

Markus Krötzsch (AIFB Karlsruhe)

Description Logic Rules

Motivation:
Description logics and rules

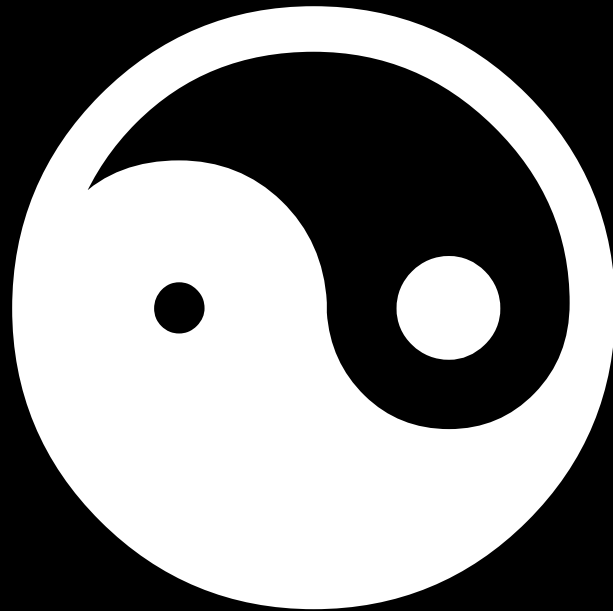


Motivation:

Description logics and rules



Motivation:
Description logics and rules



DL concept inclusions:

$$C \sqsubseteq D$$

$$\forall \mathbf{x}. C(\mathbf{x}) \rightarrow D(\mathbf{x})$$

DL concept expressions:

$C \sqcap D$

$C(x) \wedge D(x)$

DL concept expressions:

$C \sqcup D$

$C(x) \vee D(x)$

DL concept expressions:

$\neg C$

$\neg C(x)$

DL concept expressions:

$\exists R.C$

$\exists y.R(x,y) \wedge C(y)$

DL concept expressions:

$\forall R.C$

$\forall y.R(x,y) \rightarrow C(y)$

DL concept expressions:

$\{a\}$

$x \approx a$

DL concept expressions:

and many more . . .

rules in DL

some FOL rules can be
written in DL

$$C(x) \wedge \neg D(x) \rightarrow E(x) \vee F(x)$$

$$C \wedge \neg D \subseteq E \cup F$$

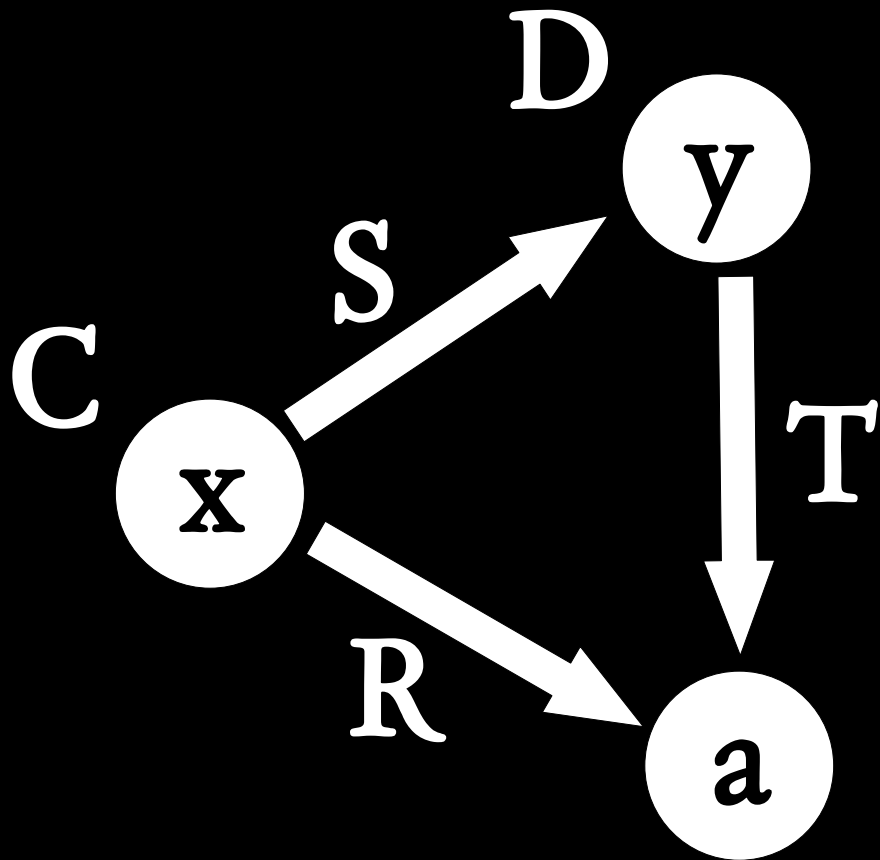
$$C(\mathbf{x}) \wedge R(\mathbf{x}, y) \rightarrow E(\mathbf{x})$$

$$C \cap \exists R.T \subseteq E$$

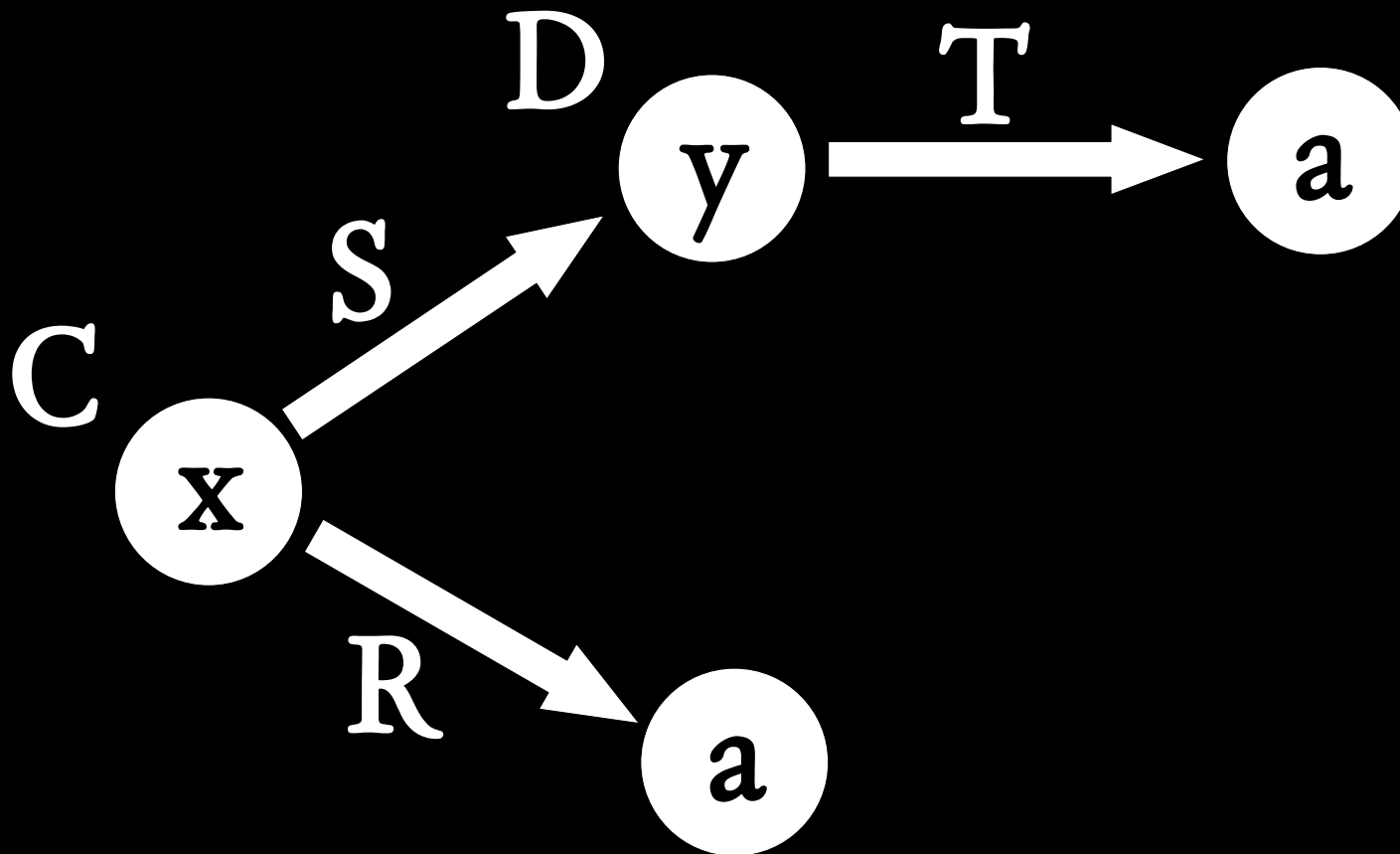
Concept inclusions are
tree-shaped rules

$$\begin{aligned} & C(x) \wedge R(x, a) \wedge S(x, y) \wedge \\ & D(y) \wedge T(y, a) \rightarrow E(x) \end{aligned}$$

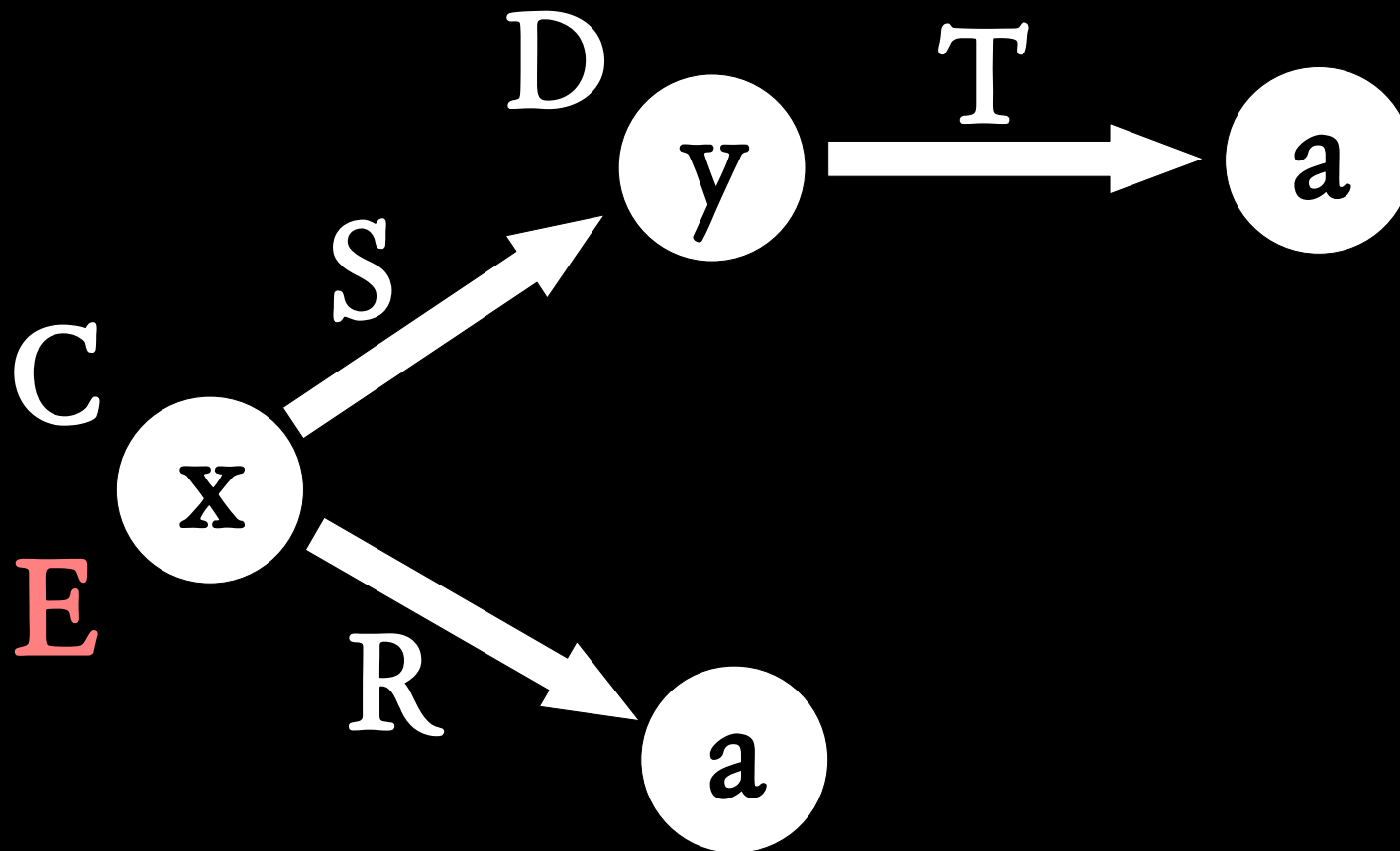
$$C(x) \wedge R(x,a) \wedge S(x,y) \wedge D(y) \wedge T(y,a) \rightarrow E(x)$$



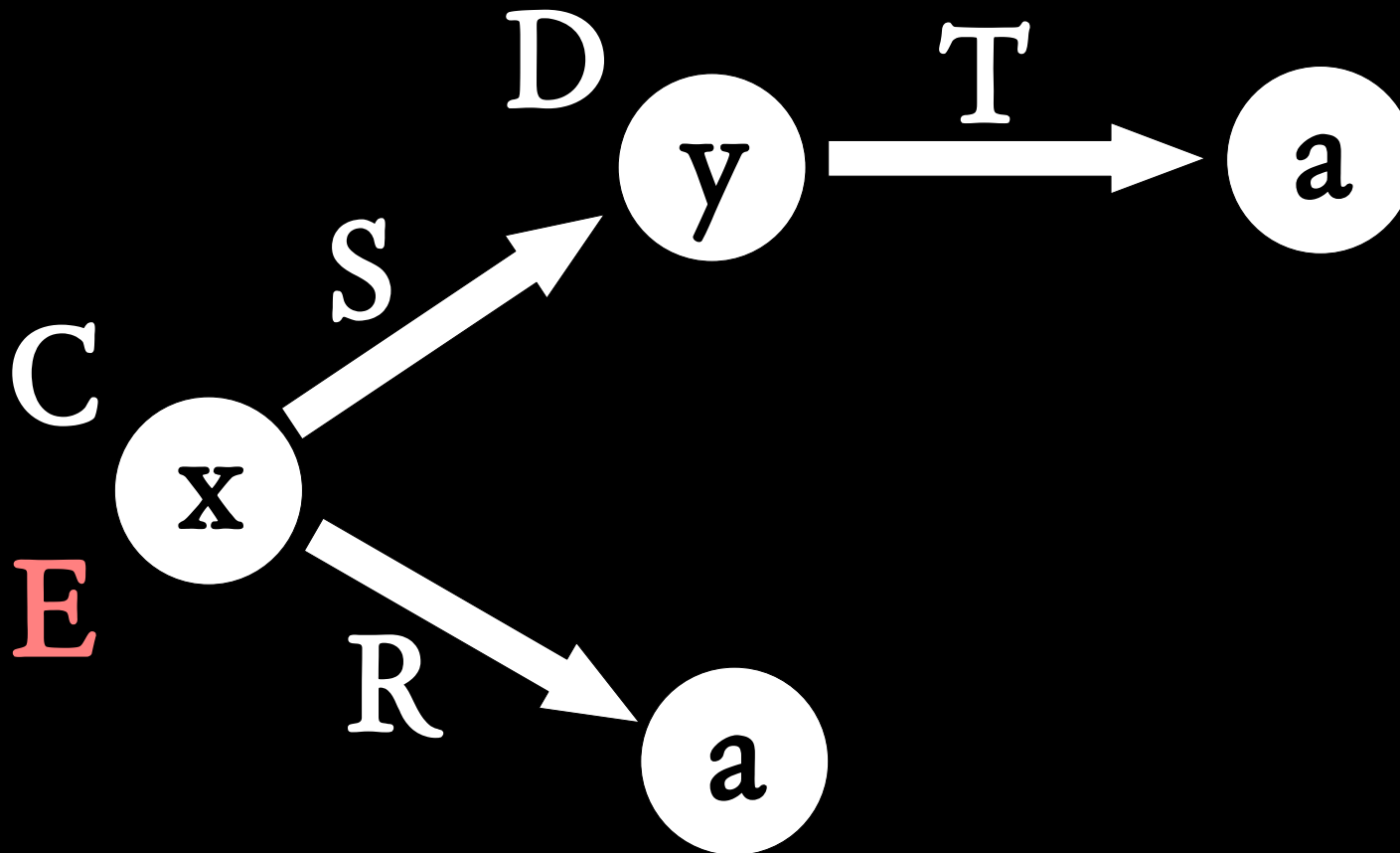
$$C(x) \wedge R(x,a) \wedge S(x,y) \wedge \\ D(y) \wedge T(y,a) \rightarrow E(x)$$



$$C(x) \wedge R(x,a) \wedge S(x,y) \wedge D(y) \wedge T(y,a) \rightarrow E(x)$$



Dependency graph tree-shaped Conclusion at root



Dependency graph tree-shaped
Conclusion at root

$C \sqcap \exists R.\{a\} \sqcap$

$\exists S.(D \sqcap \exists T.\{a\}) \sqsubseteq E$

SROIQ* can do more

***) and hence OWL2**

SROIQ role inclusion axioms

$$R \circ S \sqsubseteq T$$

$$R(x, y) \wedge S(y, z) \rightarrow T(x, z)$$

Local reflexivity

$\exists R. \text{Self}$

$R(x, x)$

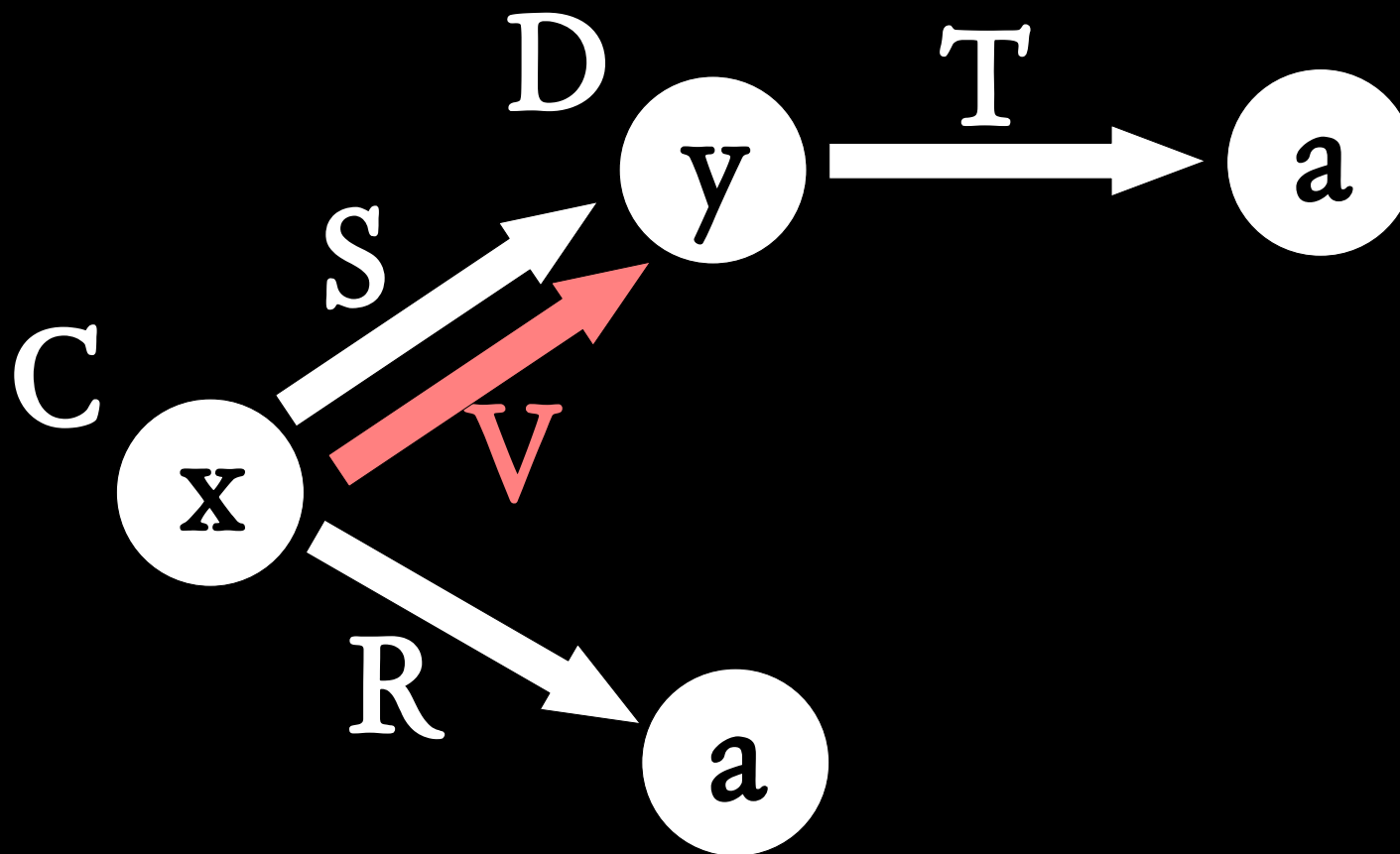
The Universal Role

U

→ U(x, y)

defining DL Rules

$$C(x) \wedge R(x,a) \wedge S(x,y) \wedge D(y) \wedge T(y,a) \rightarrow V(x,y)$$



role-up side branches

$$C(\mathbf{x}) \wedge R(\mathbf{x}, a) \wedge S(\mathbf{x}, \mathbf{y}) \wedge \\ D(\mathbf{y}) \wedge T(\mathbf{y}, a) \rightarrow V(\mathbf{x}, \mathbf{y})$$

$$(C \cap \exists R.\{a\})(\mathbf{x}) \wedge S(\mathbf{x}, \mathbf{y}) \wedge \\ (D \cap \exists T.\{a\})(\mathbf{y}) \rightarrow V(\mathbf{x}, \mathbf{y})$$

substitute roles for concepts

$$(C \cap \exists R.\{a\})(x) \wedge S(x,y) \wedge \\ (D \cap \exists T.\{a\})(y) \rightarrow V(x,y)$$

$$C \cap \exists R.\{a\} \sqsubseteq \exists R_1.\text{Self} \\ D \cap \exists T.\{a\} \sqsubseteq \exists R_2.\text{Self}$$

$$R_1(x,x) \wedge S(x,y) \wedge \\ R_2(y,y) \rightarrow V(x,y)$$

write rule as role inclusion

$$R_1(x,x) \wedge S(x,y) \wedge \\ R_2(y,y) \rightarrow V(x,y)$$

$$C \cap \exists R.\{a\} \sqsubseteq \exists R_1.\text{Self} \\ D \cap \exists T.\{a\} \sqsubseteq \exists R_2.\text{Self}$$

$$R_1 \circ S \circ R_2 \sqsubseteq V$$

from trees to forests

**Elephant(x) \wedge Mouse(y) \rightarrow
biggerThan(x,y)***

Elephant \sqsubseteq $\exists R_e$.Self

Mouse \sqsubseteq $\exists R_m$.Self

$R_e \circ U \circ R_m \sqsubseteq$ biggerThan

***) cf. [Rudolph, K_, Hitzler: DL-2008]**

DL Rules

dependency graph forest-shaped
conclusion starts at root
arbitrary DL concepts in rule

DL rules depend on the
power of the given DL

Complexity results

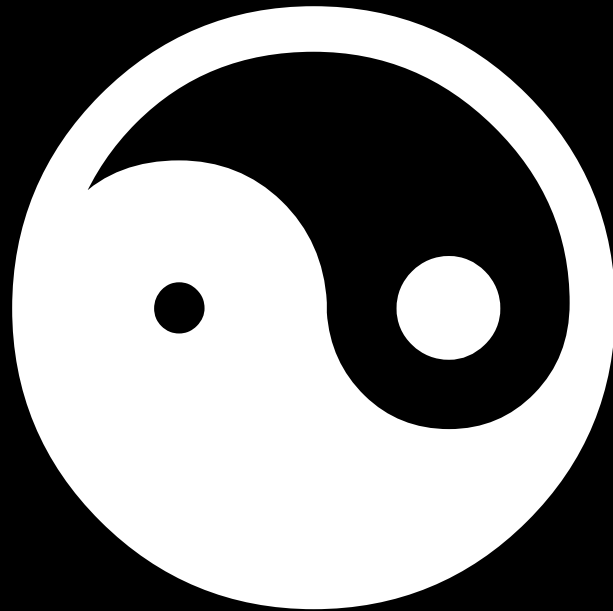
- SROIQ rules: $N_2ExpTime$ complete
- EL++ rules: PTime complete
- DLP rules: PTime complete

DL Rules vs. DL-safe rules

- restricted structure
- unrestricted “domain”
- variable complexity ($\geq PTime$)

- unrestricted structure
- restricted “domain”
- high basic complexity (ExpTime)

DL Rules vs. DL-safe rules



extensions and restrictions

Restrictions

Simplicity of roles (SROIQ)

Regularity (SROIQ)

Extensions

Inverse roles (undirected tree)

Self-concept (single object loops)

Boolean role expressions (cycles)*

*) cf. [Rudolph, K_, Hitzler: Jelia-2008]

ELP*

“merger” of EL++ and DLP

safe variables

role conjunctions

Self and U

...

*) cf. [K_, Rudolph, Hitzler: ISWC-2008]

ELP*

still in PTime

*) cf. [K_, Rudolph, Hitzler: ISWC-2008]

summary



DL Rules are
useful syntactic sugar
for SROIQ



DL Rules are
useful syntactic sugar
for SROIQ

but

can truly extend
the expressivity
of EL++ and DLP



Slides, Paper, References
at

<http://korrekt.org/>