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”

Image copyrights

Icons by http://everaldo.com, released under the terms of GPL
Where do we want to get to?
Where do we want to get to?

Semantic Web language that is ...

- focussed on data,
- rule-oriented,
- able to express schema knowledge,
- easy to implement,
- of polynomial worst-case complexity,
- compatible with OWL.
OWL and Description Logics
OWL and Description Logics

- Description logics
- FOL
"Sebastian ordered some Thai curry."

\[ sebastian: \exists \text{orderedDish} \cdot \text{ThaiCurry} \]

\[ \exists x. \text{orderedDish}(sebastian,x) \land \text{ThaiCurry}(x) \]
OWL and Description Logics

“Everything ordered as a dish is actually a dish.”

\[ \top \subseteq \forall \text{orderedDish}.\text{Dish} \]

\[ \forall x. \forall y. \text{orderedDish}(x,y) \rightarrow \text{Dish}(y) \]
"Every Thai curry dish contains peanut oil."

\[
\text{ThaiCurry} \sqsubseteq \exists \text{contains.}\{\text{peanutOil}\}
\]

\[
\forall x. \text{ThaiCurry}(x) \rightarrow \text{contains}(x, \text{peanutOil})
\]
Rules in First-Order Logic
Rules in First-Order Logic

FOL

Datalog
Rules in First-Order Logic

“Nut allergics dislike nut products.”

\[ \text{NutAllergic}(x) \land \text{NutProduct}(y) \rightarrow \text{dislikes}(x, y) \]
“People who order a dish they dislike are unhappy.”

\[ \text{orderedDish}(x,y) \land \text{dislikes}(x,y) \rightarrow \text{Unhappy}(x) \]
Rules in First-Order Logic

“If someone dislikes an ingredient of a dish, she will also dislike the dish.”

\[
dislikes(x,z) \land Dish(y) \land contains(y,z) \rightarrow dislikes(x,y)
\]
Rules in First-Order Logic

“Sebastian is a nut allergic, and peanut oil is a nut product.”

$\rightarrow \text{NutAllergic(sebastian)}$

$\rightarrow \text{NutProduct(peanutOil)}$
Can we combine datalog rules and DL axioms?
"ThaiCurry ⊑ ∃ contains. {peanutOil}

⊤ ⊑ ∀ orderedDish. Dish

sebastian: ∃ orderedDish. ThaiCurry

NutAllergic(x) ∧ NutProduct(y) → dislikes(x, y)

orderedDish(x, y) ∧ dislikes(x, y) → Unhappy(x)

dislikes(x, z) ∧ Dish(y) ∧ contains(y, z) → dislikes(x, y)

→ NutAllergic(sebastian)

→ NutProduct(peanutOil) "
Combining OWL and Rules
DLP: “OWL ∩ datalog”

\[ \text{ThaiCurry} \subseteq \exists \text{contains.}\{\text{peanutOil}\} \checkmark \]
\[ \top \subseteq \forall \text{orderedDish.Dish} \checkmark \]
\[ \text{sebastian: } \exists \text{orderedDish.ThaiCurry} \times \]
\[ \text{NutAllergic}(x) \land \text{NutProduct}(y) \rightarrow \text{dislikes}(x,y) \times \times \]
\[ \text{orderedDish}(x,y) \land \text{dislikes}(x,y) \rightarrow \text{Unhappy}(x) \times \times \times \]
\[ \text{dislikes}(x,z) \land \text{Dish}(y) \land \text{contains}(y,z) \rightarrow \text{dislikes}(x,y) \times \times \times \]
\[ \rightarrow \text{NutAllergic}(\text{sebastian}) \checkmark \]
\[ \rightarrow \text{NutProduct}(\text{peanutOil}) \checkmark \]
SWRL: “OWL ⋃ datalog”

ThaiCurry ⊑ ∃contains.{peanutOil}

⊤ ⊑ ∀orderedDish.Dish

sebastian: ∀orderedDish.ThaiCurry

NutAllergic(x) ∧ NutProduct(y) → dislikes(x,y)

orderedDish(x,y) ∧ dislikes(x,y) → Unhappy(x)

dislikes(x,z) ∧ Dish(y) ∧ contains(y,z) → dislikes(x,y)

→ NutAllergic(sebastian)

→ NutProduct(peanutOil)
SWRL is undecidable.
DL-safe Rules
DL-safe Rules

Restrict rules to apply only to named individuals.
DL-safe Rules

ThaiCurry ⊑ ∃contains.{peanutOil}

⊤ ⊑ ∀orderedDish.Dish

sebastian: ∃orderedDish.ThaiCurry

NutAllergic(x) ∧ NutProduct(y) ⊃ dislikes(x,y)

orderedDish(x,y) ∧ dislikes(x,y) ⊃ Unhappy(x)

dislikes(x,z) ∧ Dish(y) ∧ contains(y,z) ⊃ dislikes(x,y)

→ NutAllergic(sebastian)

→ NutProduct(peanutOil)
DL Rules
DL Rules

Restrict to rules that could (indirectly) be encoded with DL anyway.*

*) rules with “tree-shaped” bodies
DL Rules*

\[ \text{dislikes}(x,z) \land \text{Dish}(y) \land \text{contains}(y,z) \rightarrow \text{dislikes}(x,y) \]

*) rules with “tree-shaped” bodies
DL Rules

ThaiCurry \sqsubseteq \exists \text{contains.}\{\text{peanutOil}\}

\top \sqsubseteq \forall \text{orderedDish.Dish}

sebastian: \exists \text{orderedDish.ThaiCurry}

\text{NutAllergic}(x) \land \text{NutProduct}(y) \rightarrow \text{dislikes}(x,y)

\text{orderedDish}(x,y) \land \text{dislikes}(x,y) \rightarrow \text{Unhappy}(x)

\text{dislikes}(x,z) \land \text{Dish}(y) \land \text{contains}(y,z) \rightarrow \text{dislikes}(x,y)

\rightarrow \text{NutAllergic}(\text{sebastian})

\rightarrow \text{NutProduct}(\text{peanutOil})
DL-safe rules + DL Rules
DL-safe rules + DL Rules

\[ \text{ThaiCurry} \subseteq \exists \text{contains.}\{\text{peanutOil}\} \]

\[ \top \subseteq \forall \text{orderedDish.Dish} \]

\[ \text{sebastian: } \exists \text{orderedDish.ThaiCurry} \]

\[ \text{NutAllergic}(x) \land \text{NutProduct}(y) \rightarrow \text{dislikes}(x,y) \]

\[ \text{orderedDish}(x,y) \land \text{dislikes}(x,y) \rightarrow \text{Unhappy}(x) \]

\[ \text{dislikes}(x,z) \land \text{Dish}(y) \land \text{contains}(y,z) \rightarrow \text{dislikes}(x,y) \]

\[
\rightarrow \text{NutAllergic}(\text{sebastian})
\]

\[ \rightarrow \text{NutProduct}(\text{peanutOil}) \]
DL-safe rules + DL Rules

Desired conclusion does not follow

It is still computationally expensive

DL-safe rules: ExpTime

DL Rules: like DL, i.e. NExpTime for OWL DL
Tractable Profiles in OWL 2
Tractable Profiles in OWL 2

**OWL RL:** *Horn logic* fragment, similar to DLP, no existentials

**OWL EL:** includes existentials, based on DL *EL++*
Regaining Tractability: OWL 2 EL

\[\text{ThaiCurry} \sqsubseteq \exists \text{contains.}\{\text{peanutOil}\}\]

\[\top \sqsubseteq \forall \text{orderedDish.Dish}\]

\[\text{sebastian: } \exists \text{orderedDish.ThaiCurry}\]

\[\text{NutAllergic}(x) \land \text{NutProduct}(y) \rightarrow \text{dislikes}(x,y)\]

\[\text{orderedDish}(x,y) \land \text{dislikes}(x,y) \rightarrow \text{Unhappy}(x)\]

\[\text{dislikes}(x,z) \land \text{Dish}(y) \land \text{contains}(y,z) \rightarrow \text{dislikes}(x,y)\]

\[\rightarrow \text{NutAllergic}(\text{sebastian})\]

\[\rightarrow \text{NutProduct}(\text{peanutOil})\]
Regaining Tractability: OWL 2 RL

ThaiCurry ⊆ ∃contains.{peanutOil} ✔

⊤ ⊆ ∀orderedDish.Dish ✔

sebastian: ∃orderedDish.ThaiCurry ❌

NutAllergic(x) ∧ NutProduct(y) → dislikes(x,y) ❌

orderedDish(x,y) ∧ dislikes(x,y) → Unhappy(x) ❌

dislikes(x,z) ∧ Dish(y) ∧ contains(y,z) → dislikes(x,y) ❌

→ NutAllergic(sebastian) ✔

→ NutProduct(peanutOil) ✔
OWL EL: PTime complete

OWL RL: PTime complete
OWL EL: PTime complete

OWL RL: PTime complete

OWL EL+RL: N2ExpTime complete
Bringing it all together: ELP
Bringing it all together: ELP

DL Rules for OWL EL
+ Conjunctions of Roles*
+ DL-safe variables

*) those roles must be “simple”
Theorem

Inferencing in ELP is PTime complete.
Bringing it all together: ELP

ThaiCurry ⊑ ∃contains.{peanutOil}

⊤ ⊑ ∀orderedDish.Dish

sebastian: ∃orderedDish.ThaiCurry

NutAllergic(x) ∧ NutProduct(y) → dislikes(x,y)

orderedDish(x,y) ∧ dislikes(x,y) → Unhappy(x)

dislikes(x,z) ∧ Dish(y) ∧ contains(y,z) → dislikes(x,y)

→ NutAllergic(sebastian)

→ NutProduct(peanutOil)
Bringing it all together: ELP

→ Unhappy(sebastian)
Note

ELP supports inferencing in OWL EL and OWL RL.
Understanding DL-safety

\[ \text{ThaiCurry} \subseteq \exists \text{contains.} \text{FishProduct} \]

\[ \top \subseteq \forall \text{orderedDish.Dish} \]

\textit{markus:} \[ \exists \text{orderedDish.ThaiCurry} \]

\textbf{Vegetarian}(x) \land \textbf{FishProduct}(y) \rightarrow \text{dislikes}(x,y)

\text{orderedDish}(x,y) \land \text{dislikes}(x,y) \rightarrow \text{Unhappy}(x)

\text{dislikes}(x,z) \land \text{Dish}(y) \land \text{contains}(y,z) \rightarrow \text{dislikes}(x,y)

\rightarrow \textbf{Vegetarian}(\text{markus})
Understanding DL-safety

Unhappy(markus)

cannot be concluded
Towards Implementation
Theorem

Inferencing in ELP can be reduced in linear time to inferencing in 3-variable datalog.
Reasoning through Datalog

- Transformation to datalog is completely syntactic.
- Each axiom/rule can be transformed individually.
- Datalog engines can be used as blackbox.
- Instance and subsumption checking directly in datalog.
Summary

ELP: DL-based tractable rule language

- Almost completely expressible in OWL 2
- Support for OWL EL and OWL RL
- Linear-time conversion to 3-var datalog
  → simple implementation strategy

Happy(markus) Happy(sebastian)
Happy(pascal)
[Full paper available at http://korrekt.org/page/ELP]