Realizing Default Logic over Description Logic Knowledge Bases
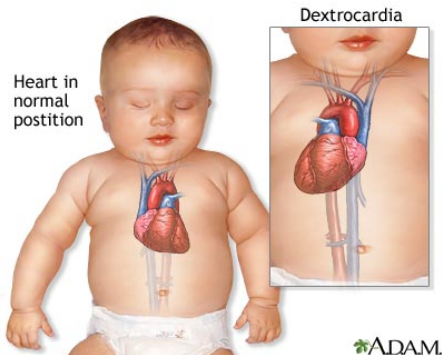
Minh Dao-Tran, Thomas Eiter, Thomas Krennwallner

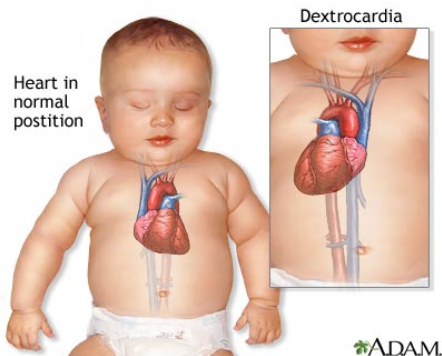KBS Group, Institute of Information Systems, Vienna University of Technology
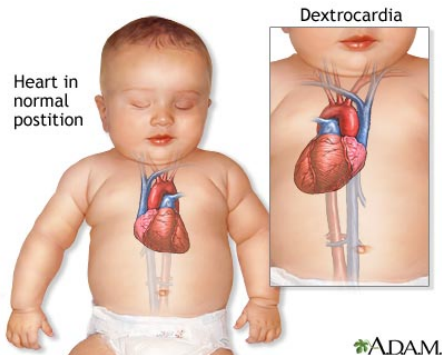
ECSQARU 2009 — July 2, 2009

The need of common-sense reasoning on top of ontologies
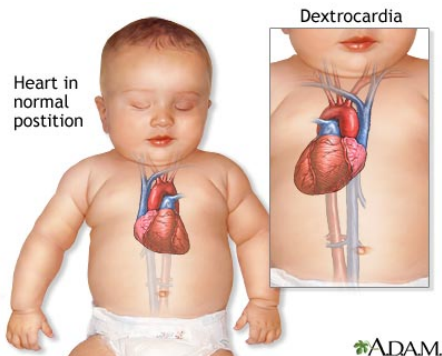
The need of common-sense reasoning on top of ontologies

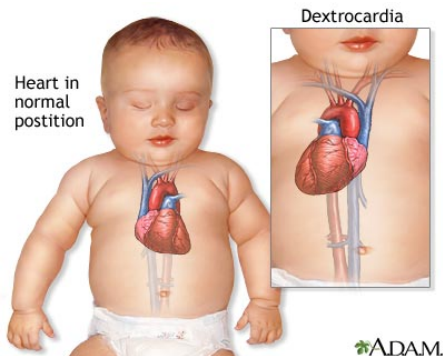# The need of common-sense reasoning on top of ontologies

The need of common-sense reasoning on top of ontologies

- *default reasoning* on top of ontologies?

The need of common-sense reasoning on top of ontologies

❓ *default reasoning* on top of ontologies?

💡 *integrations of rules and ontologies*: *cq-programs*

Description Logic Knowledge Bases (DL-KBs)

Syntax and Semantics

Name	Syntax	Semantics
Top/Bottom	$\top/\bot$	$\Delta^{\mathcal{I}}/\emptyset$
Intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Union	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Value restriction	$\forall R.C$	$\{a \in \Delta^{\mathcal{I}} \mid \forall b.(a,b) \in R^{\mathcal{I}} \to b \in C^{\mathcal{I}}\}$
Existential quant.	$\exists R.C$	$\{a \in \Delta^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$

Modeling: TBox & ABox

Description Logic Knowledge Bases (DL-KBs)

Syntax and Semantics

Name	Syntax	Semantics
Top/Bottom	$\top / \bot$	$\Delta^{\mathcal{I}} / \emptyset$
Intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Union	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Value restriction	$\forall R.C$	$\{a \in \Delta^{\mathcal{I}} \mid \forall b.(a, b) \in R^{\mathcal{I}} \to b \in C^{\mathcal{I}}\}$
Existential quant.	$\exists R.C$	$\{a \in \Delta^{\mathcal{I}} \mid \exists b.(a, b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$

Modeling: TBox & ABox

Translation to first-order logic

$\pi_x(A) = A(x)$

$\pi_x(C \sqcap D) = \pi_x(C) \wedge \pi_x(D)$ $\qquad \pi_x(\forall R.C) = \forall y.R(x, y) \supset \pi_y(C)$

$\pi_x(C \sqcup D) = \pi_x(C) \vee \pi_x(D)$ $\qquad \pi_x(\exists R.C) = \forall y.R(x, y) \wedge \pi_y(C)$

Default Theories over DL-KBs $\Delta = \langle L, D \rangle$

similar to [Baader and Hollunder, 1995]

Default rule

$$\overbrace{\alpha_1(\vec{X_1}) \wedge \cdots \wedge \alpha_k(\vec{X_k})}^{\alpha(\vec{X})} : \beta_1(\vec{Y_1}), \ldots, \overbrace{\beta_{i,1}(\vec{Y_{i,1}}) \wedge \cdots \wedge \beta_{i,\ell_i}(\vec{Y_{i,\ell_i}})}^{\beta_i(\vec{Y_i})}, \ldots, \beta_m(\vec{Y_m})$$
$$\underbrace{\underbrace{\gamma_1(\vec{Z_1}) \wedge \cdots \wedge \gamma_n(\vec{Z_n})}_{\gamma(\vec{Z})}}$$

Default Theories over DL-KBs $\Delta = \langle L, D \rangle$

similar to [Baader and Hollunder, 1995]

Default rule

$$\overbrace{\underbrace{\alpha_1(\vec{X_1}) \wedge \cdots \wedge \alpha_k(\vec{X_k})}_{\alpha(\vec{X})} : \beta_1(\vec{Y_1}), \ldots, \overbrace{\beta_{i,1}(\vec{Y}_{i,1}) \wedge \cdots \wedge \beta_{i,\ell_i}(\vec{Y}_{i,\ell_i})}^{\beta_i(\vec{Y_i})}, \ldots, \beta_m(\vec{Y_m})}{\underbrace{\gamma_1(\vec{Z_1}) \wedge \cdots \wedge \gamma_n(\vec{Z_n})}_{\gamma(\vec{Z})}}$$

Semantics: based on the Γ_Δ *operator*

- ▶ Let S be a set of assertions, then $\Gamma_\Delta(S)$ is the smallest set that
    - ▸ contains $Cn(L)$
    - ▸ is deductively closed
    - ▸ if $\alpha(\vec{X}) \in \Gamma_\Delta(S)$ and $\neg\beta_i(\vec{Y_i}) \notin S$, then $\gamma(\vec{Z}) \in \Gamma_\Delta(S)$

Default Theories over DL-KBs $\Delta = \langle L, D \rangle$

similar to [Baader and Hollunder, 1995]

Default rule

$$\overbrace{\alpha_1(\vec{X_1}) \wedge \cdots \wedge \alpha_k(\vec{X_k})}^{\alpha(\vec{X})} : \beta_1(\vec{Y_1}), \ldots, \overbrace{\beta_{i,1}(\vec{Y_{i,1}}) \wedge \cdots \wedge \beta_{i,\ell_i}(\vec{Y_{i,\ell_i}})}^{\beta_i(\vec{Y_i})}, \ldots, \beta_m(\vec{Y_m})$$
$$\underbrace{\gamma_1(\vec{Z_1}) \wedge \cdots \wedge \gamma_n(\vec{Z_n})}_{\gamma(\vec{Z})}$$

Semantics: based on the Γ_Δ *operator*

- ▶ Let S be a set of assertions, then $\Gamma_\Delta(S)$ is the smallest set that
  - ▶ contains $Cn(L)$
  - ▶ is deductively closed
  - ▶ if $\alpha(\vec{X}) \in \Gamma_\Delta(S)$ and $\neg\beta_i(\vec{Y_i}) \notin S$, then $\gamma(\vec{Z}) \in \Gamma_\Delta(S)$

- ▶ E is an *extension* of Δ iff $\Gamma_\Delta(E) = E$

Example

$$\Delta = \langle L, D \rangle$$

$$L = \left\{ \begin{array}{l} \textit{Flier} \sqsubseteq \neg \textit{NonFlier}, \textit{Penguin} \sqsubseteq \textit{Bird}, \\ \textit{Penguin} \sqsubseteq \textit{NonFlier}, \textit{Bird}(\textit{tweety}) \end{array} \right\}$$

$$D = \left\{ \frac{\textit{Bird}(X) : \textit{Flier}(X)}{\textit{Flier}(X)} \right\}$$

$$E = Cn(L \cup \{\textit{Flier}(\textit{tweety})\})$$

Example

$$\Delta' = \langle L', D \rangle$$

$$L' = \left\{ \begin{array}{l} \textit{Flier} \sqsubseteq \neg\textit{NonFlier}, \textit{Penguin} \sqsubseteq \textit{Bird}, \\ \textit{Penguin} \sqsubseteq \textit{NonFlier}, \textit{Penguin}(\textit{tweety}) \end{array} \right\}$$

$$D = \left\{ \frac{\textit{Bird}(X) : \textit{Flier}(X)}{\textit{Flier}(X)} \right\}$$

$$E' = Cn(L')$$

Conjunctive Query Programs [Eiter *et al.*, 2008a]

▶ (union of) conjunctive queries:

$$q(X) = \{X \mid \textit{AirCraft}(X) \lor (\textit{UFO}(X) \land \neg\textit{Hoax}(X))\}$$

Conjunctive Query Programs [Eiter *et al.*, 2008a]

▶ (union of) conjunctive queries:

$$q(X) = \{X \mid AirCraft(X) \vee (UFO(X) \wedge \neg Hoax(X))\}$$

▶ cq-atom:

$$DL[AirCraft \uplus isBoeing; AirCraft(X) \vee (UFO(X), \neg Hoax(X))](X)$$

Conjunctive Query Programs [Eiter *et al.*, 2008a]

- ▶ (union of) conjunctive queries:

$$q(X) = \{X \mid AirCraft(X) \vee (UFO(X) \wedge \neg Hoax(X))\}$$

- ▶ cq-atom:

$$\mathrm{DL}[AirCraft \uplus isBoeing; AirCraft(X) \vee (UFO(X), \neg Hoax(X))](X)$$

- ▶ cq-rule:

$$flying\_thing(X) \leftarrow thing(X),$$
$$\mathrm{DL}[AirCraft \uplus isBoeing; AirCraft(X) \vee (UFO(X), \neg Hoax(X))](X).$$

Conjunctive Query Programs [Eiter *et al.*, 2008a]

- ▶ (union of) conjunctive queries:

$$q(X) = \{X \mid \mathit{AirCraft}(X) \vee (\mathit{UFO}(X) \wedge \neg\mathit{Hoax}(X))\}$$

- ▶ cq-atom:

$$\mathrm{DL}[\mathit{AirCraft} \uplus \mathit{isBoeing}; \mathit{AirCraft}(X) \vee (\mathit{UFO}(X), \neg\mathit{Hoax}(X))](X)$$

- ▶ cq-rule:

$$\mathit{flying\_thing}(X) \leftarrow \mathit{thing}(X),$$
$$\mathrm{DL}[\mathit{AirCraft} \uplus \mathit{isBoeing}; \mathit{AirCraft}(X) \vee (\mathit{UFO}(X), \neg\mathit{Hoax}(X))](X).$$

- ▶ cq-program: $KB = (L, P)$ — based on *answer set semantics*

Answer Set Semantics of cq-Programs

generalized [Gelfond and Lifschitz, 1991]

P is a set of rules of form $a \leftarrow b_1, \ldots, b_m, \text{not } c_1, \ldots, \text{not } c_n.$

Answer Set Semantics of cq-Programs

generalized [Gelfond and Lifschitz, 1991]

P is a set of rules of form $a \leftarrow b_1, \ldots, b_m, \text{not } c_1, \ldots, \text{not } c_n.$

- ▶ Interpretation $I \subseteq HB_P$

Answer Set Semantics of cq-Programs

generalized [Gelfond and Lifschitz, 1991]

P is a set of rules of form $a \leftarrow b_1, \ldots, b_m, \text{not } c_1, \ldots, \text{not } c_n.$

- Interpretation $I \subseteq HB_P$

- I is an answer set of P if I is the least model of the reduct P^I

Answer Set Semantics of cq-Programs

generalized [Gelfond and Lifschitz, 1991]

P is a set of rules of form $a \leftarrow b_1, \ldots, b_m, \text{not } c_1, \ldots, \text{not } c_n.$

- ▶ Interpretation $I \subseteq HB_P$

- ▶ I is an answer set of P if I is the least model of the reduct P^I

  P^I is constructed by
  - ▶ removing rules $r \in P$ such that $c_i \in I$

  - ▶ removing all c_i and nonmonotonic cq-atoms from remaining rules

Transformation Ω (inspired by [Eiter *et al.*, 2008b])

D is a set of defaults of form

$$\cfrac{\alpha(\vec{X}) : \beta_1(\vec{Y}_1), \ldots, \overbrace{\beta_{i,1}(\vec{Y}_{i,1}) \land \cdots \land \beta_{i,\ell_i}(\vec{Y}_{i,\ell_i})}^{\beta_i(\vec{Y}_i)}, \ldots, \beta_m(\vec{Y}_m)}{\underbrace{\gamma_1(\vec{Z}_1) \land \cdots \land \gamma_n(\vec{Z}_n)}_{\gamma(\vec{Z})}}$$

Transformation Ω (inspired by [Eiter *et al.*, 2008b])

D is a set of defaults of form

$$\frac{\alpha(\vec{X}) : \beta_1(\vec{Y}_1), \ldots, \overbrace{\beta_{i,1}(\vec{Y}_{i,1}) \wedge \cdots \wedge \beta_{i,\ell_i}(\vec{Y}_{i,\ell_i})}^{\beta_i(\vec{Y}_i)}, \ldots, \beta_m(\vec{Y}_m)}{\underbrace{\gamma_1(\vec{Z}_1) \wedge \cdots \wedge \gamma_n(\vec{Z}_n)}_{\gamma(\vec{Z})}}$$

▶ Concluding rules $\mathcal{R} = \{ in_{\gamma_i}(\vec{Z}_i) \leftarrow in_\gamma(\vec{Z}) \mid 1 \leq i \leq n \}$

Transformation Ω **(inspired by [Eiter *et al.*, 2008b])**

D is a set of defaults of form

$$\frac{\alpha(\vec{X}) : \beta_1(\vec{Y}_1), \ldots, \overbrace{\beta_{i,1}(\vec{Y}_{i,1}) \wedge \cdots \wedge \beta_{i,\ell_i}(\vec{Y}_{i,\ell_i})}^{\beta_i(\vec{Y}_i)}, \ldots, \beta_m(\vec{Y}_m)}{\underbrace{\gamma_1(\vec{Z}_1) \wedge \cdots \wedge \gamma_n(\vec{Z}_n)}_{\gamma(\vec{Z})}}$$

▶ Concluding rules $\qquad \mathcal{R} = \{ in_{\gamma_i}(\vec{Z}_i) \leftarrow in_\gamma(\vec{Z}) \mid 1 \leq i \leq n \}$

▶ A rule which simulates the Γ_Δ operator

$$in_\gamma(\vec{Z}) \leftarrow \mathrm{DL}[\lambda; \alpha(\vec{X})](\vec{X}),$$

$$\mathrm{not}\ \mathrm{DL}[\lambda; \neg\beta_{1,1}(\vec{Y}_{1,1}) \vee \cdots \vee \neg\beta_{1,\ell_1}(\vec{Y}_{1,\ell_1})](\vec{Y}_1),$$

$$\vdots$$

$$\mathrm{not}\ \mathrm{DL}[\lambda; \neg\beta_{m,1}(\vec{Y}_{m,1}) \vee \cdots \vee \neg\beta_{m,\ell_m}(\vec{Y}_{m,\ell_m})](\vec{Y}_m).$$

Transformation Ω **(inspired by [Eiter *et al.*, 2008b])**

D is a set of defaults of form

$$\frac{\alpha(\vec{X}) : \beta_1(\vec{Y}_1), \ldots, \overbrace{\beta_{i,1}(\vec{Y}_{i,1}) \wedge \cdots \wedge \beta_{i,\ell_i}(\vec{Y}_{i,\ell_i})}^{\beta_i(\vec{Y}_i)}, \ldots, \beta_m(\vec{Y}_m)}{\underbrace{\gamma_1(\vec{Z}_1) \wedge \cdots \wedge \gamma_n(\vec{Z}_n)}_{\gamma(\vec{Z})}}$$

▶ Concluding rules $\qquad \mathcal{R} = \{in_{\gamma_i}(\vec{Z}_i) \leftarrow in_\gamma(\vec{Z}) \mid 1 \leq i \leq n\}$

▶ A rule which simulates the Γ_Δ operator

$$in_\gamma(\vec{Z}) \leftarrow \mathrm{DL}[\lambda; \alpha(\vec{X})](\vec{X}),$$

$$\mathrm{not\ DL}[\lambda; \neg\beta_{1,1}(\vec{Y}_{1,1}) \vee \cdots \vee \neg\beta_{1,\ell_1}(\vec{Y}_{1,\ell_1})](\vec{Y}_1),$$

$$\vdots$$

$$\mathrm{not\ DL}[\lambda; \neg\beta_{m,1}(\vec{Y}_{m,1}) \vee \cdots \vee \neg\beta_{m,\ell_m}(\vec{Y}_{m,\ell_m})](\vec{Y}_m).$$

where $\lambda = (\gamma_i^* \uplus in_{\gamma_i}, \gamma_i^* \uplus in_{\neg\gamma_i} \mid \delta \in D)$; $\qquad \gamma_i^*$ is γ_i without $\neg$

Transformation Ω - Bird Example

$$
\begin{aligned}
in_{Flier}(X) \quad \leftarrow \quad & \mathrm{DL}[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; Bird(X)](X), \\
& \mathrm{not} \; \mathrm{DL}[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).
\end{aligned}
$$

Transformation Ω - Bird Example

$$in_{Flier}(X) \leftarrow \text{DL}[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; Bird(X)](X),$$
$$\text{not } \text{DL}[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$I_{\Omega} = \{in_{Flier}(tweety)\}$

Transformation Ω - Bird Example

$$in_{Flier}(X) \quad \leftarrow \quad DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; Bird(X)](X),$$
$$\text{not } DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$I_{\Omega} = \{in_{Flier}(tweety)\} \; ; E_{\Omega} = Cn(L \cup \{Flier(tweety)\})$

Transformation Ω - Bird Example

$$in_{Flier}(X) \quad \leftarrow \quad \text{DL}[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; Bird(X)](X),$$
$$\text{not } \text{DL}[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$I_{\Omega} = \{in_{Flier}(tweety)\} \; ; E_{\Omega} = Cn(L \cup \{Flier(tweety)\})$

Add $\left\{ \begin{array}{c} in_{\neg Flier}(X) \leftarrow broken\_wing(X). \\ broken\_wing(tweety). \end{array} \right\}$

Transformation Ω - Bird Example

$$in_{Flier}(X) \quad \leftarrow \quad \mathrm{DL}[Flier \uplus in_{Flier}, Flier \cupdot in_{\neg Flier}; Bird(X)](X),$$
$$\mathrm{not} \; \mathrm{DL}[Flier \uplus in_{Flier}, Flier \cupdot in_{\neg Flier}; \neg Flier(X)](X).$$

$I_\Omega = \{in_{Flier}(tweety)\}$; $E_\Omega = Cn(L \cup \{Flier(tweety)\})$

Add $\left\{ \begin{array}{c} in_{\neg Flier}(X) \leftarrow broken\_wing(X). \\ broken\_wing(tweety). \end{array} \right\}$

Then $I_\Omega = \{broken\_wing(tweety), in_{\neg Flier}(tweety)\}$; $E_\Omega = Cn(L)$

Transformation Υ

based on [Cholewinski and Truszczynski, 1994]

- ▶ Concluding rules $\quad \mathcal{R} = \{in_{\gamma_i}(\vec{Z}_i) \leftarrow in_\gamma(\vec{Z}) \mid 1 \leq i \leq n\}$

Transformation Υ

based on [Cholewinski and Truszczynski, 1994]

- ► Concluding rules $\qquad \mathcal{R} = \{in_{\gamma_i}(\vec{Z}_i) \leftarrow in_{\gamma}(\vec{Z}) \mid 1 \leq i \leq n\}$

- ► Rules that select justifications:

$$cons_{\beta_i}(\vec{Y}_i) \leftarrow \text{ not } \overline{cons}_{\beta_i}(\vec{Y}_i).$$
$$\overline{cons}_{\beta_i}(\vec{Y}_i) \leftarrow \text{ not } cons_{\beta_i}(\vec{Y}_i).$$

Transformation Υ

based on [Cholewinski and Truszczynski, 1994]

- ▶ Concluding rules $\quad \mathcal{R} = \{in_{\gamma_i}(\vec{Z_i}) \leftarrow in_\gamma(\vec{Z}) \mid 1 \le i \le n\}$

- ▶ Rules that select justifications:

$$cons_{\beta_i}(\vec{Y_i}) \leftarrow \text{ not } \overline{cons}_{\beta_i}(\vec{Y_i}).$$
$$\overline{cons}_{\beta_i}(\vec{Y_i}) \leftarrow \text{ not } cons_{\beta_i}(\vec{Y_i}).$$

- ▶ A rule which simulates the Γ_Δ operator:

$$in_\gamma(\vec{Z}) \leftarrow DL[\lambda; \alpha(\vec{X})](\vec{X}), cons_{\beta_1}(\vec{Y_1}), \ldots, cons_{\beta_m}(\vec{Y_m}).$$

where $\lambda = (\gamma_i^* \uplus in_{\gamma_i}, \gamma_i^* \uplus in_{\neg\gamma_i} \mid \delta \in D)$ $\qquad$ γ_i^* is γ_i without $\neg$

Transformation Υ - cont.

▶ Constraints that check the compliance of our guess with the result

$$fail \leftarrow \mathrm{DL}[\lambda; \neg\beta_{i,1}(\vec{Y}_{i,1}) \vee \cdots \vee \neg\beta_{i,\ell_i}(\vec{Y}_{i,\ell_i})](\vec{Y}_i), cons_{\beta_i}(\vec{Y}_i), \mathrm{not}\ fail.$$

$$fail \leftarrow \mathrm{not}\ \mathrm{DL}[\lambda; \neg\beta_{i,1}(\vec{Y}_{i,1}) \vee \cdots \vee \neg\beta_{i,\ell_i}(\vec{Y}_{i,\ell_i})](\vec{Y}_i), \overline{cons}_{\beta_i}(\vec{Y}_i), \mathrm{not}\ fail.$$

Transformation Υ - Bird Example

$$cons_{Flier}(X) \quad \leftarrow \quad \text{not } \overline{cons}_{Flier}(X).$$

$$\overline{cons}_{Flier}(X) \quad \leftarrow \quad \text{not } cons_{Flier}(X).$$

Transformation Υ - Bird Example

$$cons_{Flier}(X) \quad \leftarrow \quad \text{not } \overline{cons}_{Flier}(X).$$

$$\overline{cons}_{Flier}(X) \quad \leftarrow \quad \text{not } cons_{Flier}(X).$$

$$in_{Flier}(X) \quad \leftarrow \quad \text{DL}[Flier \uplus in_{Flier}; Bird(X)](X), cons_{Flier}(X).$$

Transformation Υ - Bird Example

$cons_{Flier}(X) \quad \leftarrow \quad$ not $\overline{cons}_{Flier}(X).$

$\overline{cons}_{Flier}(X) \quad \leftarrow \quad$ not $cons_{Flier}(X).$

$in_{Flier}(X) \quad \leftarrow \quad \text{DL}[Flier \uplus in_{Flier}; Bird(X)](X), cons_{Flier}(X).$

$fail \quad \leftarrow \quad$ not $fail, cons_{Flier}(X),$
$\text{DL}[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$

Transformation ϒ - Bird Example

$$cons_{Flier}(X) \quad \leftarrow \quad \text{not } \overline{cons}_{Flier}(X).$$

$$\overline{cons}_{Flier}(X) \quad \leftarrow \quad \text{not } cons_{Flier}(X).$$

$$in_{Flier}(X) \quad \leftarrow \quad DL[Flier \uplus in_{Flier}; Bird(X)](X), cons_{Flier}(X).$$

$$fail \quad \leftarrow \quad \text{not } fail, cons_{Flier}(X),$$
$$DL[Flier \uplus in_{Flier}, Flier \dot\cup in_{\neg Flier}; \neg Flier(X)](X).$$

$$fail \quad \leftarrow \quad \text{not } fail, \overline{cons}_{Flier}(X),$$
$$\text{not } DL[Flier \uplus in_{Flier}, Flier \dot\cup in_{\neg Flier}; \neg Flier(X)](X).$$

Transformation Υ - Bird Example

$$cons_{Flier}(X) \leftarrow \text{not } \overline{cons}_{Flier}(X).$$

$$\overline{cons}_{Flier}(X) \leftarrow \text{not } cons_{Flier}(X).$$

$$in_{Flier}(X) \leftarrow DL[Flier \uplus in_{Flier}; Bird(X)](X), cons_{Flier}(X).$$

$$fail \leftarrow \text{not } fail, cons_{Flier}(X),$$
$$DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$$fail \leftarrow \text{not } fail, \overline{cons}_{Flier}(X),$$
$$\text{not } DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$$I_\Upsilon = \{in_{Flier}(tweety), cons_{Flier}(tweety)\}$$

Transformation Υ - Bird Example

$$cons_{Flier}(X) \leftarrow not\ \overline{cons}_{Flier}(X).$$

$$\overline{cons}_{Flier}(X) \leftarrow not\ cons_{Flier}(X).$$

$$in_{Flier}(X) \leftarrow DL[Flier \uplus in_{Flier}; Bird(X)](X), cons_{Flier}(X).$$

$$fail \leftarrow not\ fail, cons_{Flier}(X),$$
$$DL[Flier \uplus in_{Flier}, Flier \cup in_{\neg Flier}; \neg Flier(X)](X).$$

$$fail \leftarrow not\ fail, \overline{cons}_{Flier}(X),$$
$$not\ DL[Flier \uplus in_{Flier}, Flier \cup in_{\neg Flier}; \neg Flier(X)](X).$$

$I_\Upsilon = \{in_{Flier}(tweety), cons_{Flier}(tweety)\}\ ;\ E_\Omega = Cn(L \cup \{Flier(tweety)\})$

Transformation ϒ - Bird Example

$$cons_{Flier}(X) \leftarrow not\ \overline{cons}_{Flier}(X).$$

$$\overline{cons}_{Flier}(X) \leftarrow not\ cons_{Flier}(X).$$

$$in_{Flier}(X) \leftarrow DL[Flier \uplus in_{Flier}; Bird(X)](X), cons_{Flier}(X).$$

$$fail \leftarrow not\ fail, cons_{Flier}(X),$$
$$DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$$fail \leftarrow not\ fail, \overline{cons}_{Flier}(X),$$
$$not\ DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$I_{\Upsilon} = \{in_{Flier}(tweety), cons_{Flier}(tweety)\}$; $E_{\Omega} = Cn(L \cup \{Flier(tweety)\})$

$L = L \cup \{Penguin(tweety)\}$

Transformation Υ - Bird Example

$$cons_{Flier}(X) \leftarrow not\ \overline{cons}_{Flier}(X).$$

$$\overline{cons}_{Flier}(X) \leftarrow not\ cons_{Flier}(X).$$

$$in_{Flier}(X) \leftarrow DL[Flier \uplus in_{Flier}; Bird(X)](X), cons_{Flier}(X).$$

$$fail \leftarrow not\ fail, cons_{Flier}(X),$$
$$DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$$fail \leftarrow not\ fail, \overline{cons}_{Flier}(X),$$
$$not\ DL[Flier \uplus in_{Flier}, Flier \uplus in_{\neg Flier}; \neg Flier(X)](X).$$

$I_{\Upsilon} = \{in_{Flier}(tweety), cons_{Flier}(tweety)\}$; $E_{\Omega} = Cn(L \cup \{Flier(tweety)\})$

$L = L \cup \{Penguin(tweety)\}$, then $I_{\Upsilon} = \{\overline{cons}_{Flier}(tweety)\}$; $E_{\Omega} = Cn(L)$

Comparison
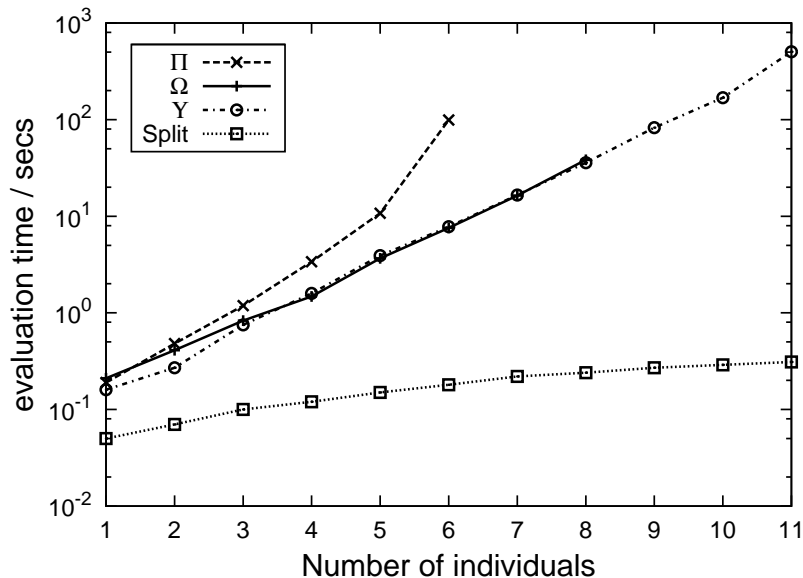
Conclusion & Future Work

- ▶ Conclusions:
    - ▶ Consider default theories over DL-KBs $\Delta = \langle L, D \rangle$
    - ▶ Adapt Reiter's Default Logic using Γ_Δ operator
    - ▶ Two new transformations to cq-programs (Ω and Υ)
    - ▶ Ω and Υ perform better than an old transformation in [Eiter *et al.*, 2008b]

Conclusion & Future Work

- Conclusions:
  - Consider default theories over DL-KBs $\Delta = \langle L, D \rangle$
  - Adapt Reiter's Default Logic using Γ_Δ operator
  - Two new transformations to cq-programs (Ω and Υ)
  - Ω and Υ perform better than an old transformation in [Eiter *et al.*, 2008b]

- Future work:
  - Investigate special default theories (normal/semi-normal defaults)
  - Implement caching for cq-programs
  - Interface to different DL-reasoners, eg., Pellet, KAON2

References I

📖 Franz Baader and Bernhard Hollunder.
Embedding Defaults into Terminological Knowledge Representation Formalisms.
Journal of Automated Reasoning, 14(1):149–180, February 1995.

📖 P. Cholewinski and M. Truszczynski.
Minimal number of permutations sufficient to compute all extensions a finite default theory.
unpublished note, 1994.

📖 Thomas Eiter, Giovambattista Ianni, Thomas Krennwallner, and Roman Schindlauer.
Exploiting conjunctive queries in description logic programs.
Annals of Mathematics and Artificial Intelligence: Logic in AI: A Special Issue Dedicated to Victor W. Marek on the Occasion of His 65th birthday, 53(1–4):115–152, August 2008.
Published online: 27 January 2009.

References II

📑 Thomas Eiter, Giovambattista Ianni, Thomas Lukasiewicz, Roman Schindlauer, and Hans Tompits.
Combining answer set programming with description logics for the semantic web.
Artificial Intelligence, 172(12-13):1495–1539, August 2008.

📑 Michael Gelfond and Vladimir Lifschitz.
Classical negation in logic programs and deductive databases.
New Generation Computing, 9:365–385, 1991.