Datalog-Based Data Access over Ontology Knowledge Bases

Unit 2 – Programs on Top of Ontologies

Thomas Eiter

Institut für Informationsysteme, TU Wien

ICCL Summer School 2013, August 29-30, 2013

Austrian Science Fund (FWF) grants P20841, P24090
Unit Outline

1. Introduction
2. dl-Programs
3. HEX-Programs
4. Conclusion
Recall: Loose Coupling

- Strict semantic separation between rules / ontology

- View rule base $P$ and FO theory $O$ as separate, independent components. $\Sigma_O$ and $\Sigma_P$ do (a priori) not share meaning.

- They are connected through a minimal “safe interface” for exchanging knowledge (formulas, usually ground atoms).

- Prominent representative: nonmonotonic dl-programs
dl-Programs

- An extension of answer set programs with *queries to DL knowledge bases (DL KBs)* [E_ et al., 2008b]
- Queries can *temporarily update* the DL KB
  
  *bidirectional flow of information*, with clean technical separation of DL engine and ASP solver (“loose coupling”)

- Use dl-programs as “glue” for combining inferences on a DL KB.
- Experimental prototypes
  - NLP-DL [https://www.mat.unical.it/ianni/swlp/](https://www.mat.unical.it/ianni/swlp/)
  - dlvhex DL Plugin [http://www.kr.tuwien.ac.at/research/systems/dlvhex/dlplugin.html](http://www.kr.tuwien.ac.at/research/systems/dlvhex/dlplugin.html)
  - #F-Logic programs (Ontoprise, extension to F-logic programs)
  - DReW [http://www.kr.tuwien.ac.at/research/systems/drew/](http://www.kr.tuwien.ac.at/research/systems/drew/)
dl-atoms

**Basic Idea:**

- Query the DL KB $\mathcal{O}$ using the *query interface* of the DL engine

  Query $Q$ may be concept/role instance $C(X) \lor R(X, Y)$; subsumption test $C \sqsubseteq D$; etc (recent extension: conjunctive queries)

- **Important:** Possible to **modify** the extensional part (ABox) of $\mathcal{O}$, by adding positive ($\lor$) or negative ($\lor, \land$) assertions, before querying

- $Q$ evaluates to true iff the modified $\mathcal{O}$ proves $Q$. 
dl-atoms: Examples

Wine ontology

- DL[Wine](“ChiantiClassico”)
- DL[Wine](X)
- DL[DryWine ⊔ dry; Wine](W)

add all assertions DryWine(c) to O, such that dry(c) holds.
dl-Atoms: Syntax

**dl-atom**

A dl-atom has the form

\[ DL[S_1 op_1 p_1, \ldots, S_m op_m p_m; Q](t) , \quad m \geq 0, \]

where

- each \( S_i \) is either a concept or a role
- \( op_i \in \{ \lor, \lor, \land \} \),
- \( p_i \) is a unary resp. binary predicate (input predicate),
- \( Q(t) \) is a dl-query (\( t \) contains variables and/or constants).

Intuitively:

\[ op_i = \lor \text{ increases } S_i \text{ by } p_i; \quad op_i = \land \text{ increases } S_i \text{ by } \neg p_i; \]

\[ op_i = \lor \text{ increases } \neg S_i \text{ by } p_i. \]

Shorthand: \( \lambda = S_1 op_1 p_1, \ldots, S_m op_m p_m \)
dl-Queries

A dl-query $Q(t)$ is one of

(a) a concept inclusion axiom $C \sqsubseteq D$, or its negation $\neg(C \sqsubseteq D)$,
(b) $C(t)$ or $\neg C(t)$, for a concept $C$ and term $t$, or
(c) $R(t_1, t_2)$ or $\neg R(t_1, t_2)$, for a role $R$ and terms $t_1, t_2$.

Note:

- The queries above are standard queries
- Further queries are conceivable (e.g., conjunctive queries, union of conjunctive queries [E_ et al., 2008a]),
  - Assumption: decidability
**dl-Programs**

dl-programs are hybrid KBs with dl-atoms in rules

---

**dl-Program**

A (disjunctive) dl-program is a pair $\Pi = (\mathcal{O}, P)$ where

- $\mathcal{O}$ is a DL knowledge base ("ontology")
- $P$ consists of dl-rules

$$a_1 \lor \cdots \lor a_n \leftarrow b_1, \ldots, b_k, \text{not } b_{k+1}, \ldots, \text{not } b_m,$$

$m + n > 0$, where

- **not** is default negation ("unless derivable"),
- $a_1, \ldots, a_n$ are atoms,
- $b_1, \ldots, b_m, m \geq 0$, are atoms or dl-atoms (no function symbols).

Note: rules with classical ("strong") negation ($\neg a_i, \neg b_j$) can be emulated
### Semantics

- $HB_P^Φ$: set of all ground (classical) atoms with predicate symbol in $P$ and constants $C$ from finite relational alphabet $Φ$.
- Constants $C$: those in $P$ and (all) individuals in the ABox of $O$.
- Herbrand interpretation: subset $I \subseteq HB_P^Φ$.

#### Satisfaction ($I \models_Ο a, I \models_Ο r, I \models_Ο P$)

- $I$ satisfies a classical ground atom $a$ iff $a \in I$;
- $I$ satisfies a ground dl-atom $a = DL[λ; Q](c)$ iff $O \cup \bigcup_{i=1}^{m} A_i(I) \models Q(c)$, where
  - $A_i(I) = \{ S_i(e) \mid p_i(e) \in I \}$, for $op_i = \sqcup$;
  - $A_i(I) = \{ \neg S_i(e) \mid p_i(e) \in I \}$, for $op_i = \sqcup$.
  - $A_i(I) = \{ \neg S_i(e) \mid p_i(e) \in HB_P^Φ \setminus I \}$, for $op_i = \cap$. non-monotonic in $I$
- $I$ satisfies a ground rule $r$ of form (1) if $I \models_Ο b_i, 1 \leq i \leq k$ and $I \not\models_Ο b_j, k < j \leq m$ implies $I \models_Ο a_i$, for some $1 \leq i \leq n$.
- $I$ satisfies $(Ο, P)$ if $I \models_Ο r$ for each $r$ in the grounding $grnd_C(P)$ of $P$ wrt. $C$. 
Examples

Suppose $\mathcal{O} \models \text{Wine("TaylorPort")}$, and $I$ contains $\text{wineBottle("TaylorPort")}$.

Then $I \models_{\mathcal{O}} DL[\text{"Wine"}](\text{"TaylorPort"})$ and

$I \models_{\mathcal{O}} \text{wineBottle("TaylorPort")} \leftarrow DL[\text{"Wine"}](\text{"TaylorPort"})$

Suppose $I = \{\text{white("siw")}, \text{not\_dry("siw")}\}$.

Then $I \models_{\mathcal{O}} DL[\text{"WhiteWine"} \uplus \text{white}, \text{"DryWine"} \uplus \text{not\_dry}; \text{"Wine"}](\text{"siw"})$
Examples /2

- Suppose $O \not\models DL["Wine"]("Milk")$. Then for every $I$,
  
  $I \models_O not\ DL["Wine"]("Milk")$

  $I \models_O compliant(joe,"Milk") \leftarrow DL["Wine"]("Milk")$.

- Note that $I \models_O not\ DL["Wine"]("Milk")$ is different from
  
  $I \models_O DL[\neg "Wine"]("Milk")$.

- Possibility to check satisfiability of the ontology $O$ by rules
  
  - Unsatisfiability: e.g.
    
    $unsat_{\text{ontology}} \leftarrow DL[; \bot](a)$

  - Satisfiability: e.g.
    
    $sat_{\text{ontology}} \leftarrow not\ DL[; T \sqsubseteq \bot]()$

  - Similar for updated ontology:
    
    $unsat_{\text{ontology}} \leftarrow DL["WhiteWine" \uplus white; T \sqsubseteq \bot]()$
Answer Sets

- Use a reduct $\Pi^I$ akin to the Gelfond-Lifschitz reduct $P^I$.
- In building $\Pi^I$, treat dl-atoms like not-literals: “guess” their truth value.

Reduct $\Pi^I$ of $\Pi = (\mathcal{O}, P)$

$\Pi^I = (\mathcal{O}, P^I)$ where $P^I$ contains all rules obtained from $grnd_c(P)$ by

(i) removing all rule instances

$$a_1 \lor \cdots \lor a_n \leftarrow b_1, \ldots, b_k, \text{not } b_{k+1}, \ldots, \text{not } b_m$$

such that either $I \models b_j$ for some $b_j$, $k < j \leq m$, or $I \not\models b_i$ for some dl-atom $b_i$, $1 \leq i \leq k$, and

(ii) removing all dl-literals and literals not $b_j$ from the remaining rules.

- $P^I$ is ordinary; if $I$ is reconstructible from the guess it is “stable”

(Weak) Answer Set

$I$ is a (weak) answer set of $\Pi$ iff $I$ is a minimal model of $\Pi^I$ ($\Leftrightarrow$ of $P^I$).
Network Example

\[ \Pi = (\mathcal{O}, P) \]

Ontology \( \mathcal{O} : \)

\[
\begin{align*}
\geq & 1. \text{wired} \sqsubseteq \text{Node} \quad \top \sqsubseteq \forall \text{wired.Node} \\
\text{wired} & = \text{wired}^{-} ; \\
n_1 & \neq n_2 \neq n_3 \neq n_4 \neq n_5 \\
\text{wired}(n_1,n_2) & \text{ wired}(n_2,n_3) \text{ wired}(n_2,n_4) \\
\text{wired}(n_2,n_5) & \text{ wired}(n_3,n_4) \text{ wired}(n_3,n_5). \\
\geq & 4. \text{wired} \sqsubseteq \text{HighTrafficNode}
\end{align*}
\]

Rules \( P \)

\[
\begin{align*}
\text{newnode}(x_1). & \quad \text{newnode}(x_2). \\
\text{overloaded}(X) & \leftarrow \text{DL[wired} \cup \text{ connect;} \text{ HighTrafficNode]}(X). \\
\text{connect}(X,Y) & \leftarrow \text{newnode}(X), \text{DL[Node]}(Y), \\
& \text{not overloaded}(Y), \text{not excl}(X,Y). \\
\text{excl}(X,Y) & \leftarrow \text{connect}(X,Z), \text{DL[Node]}(Y), Y \neq Z. \\
\text{excl}(X,Y) & \leftarrow \text{connect}(Z,Y), \text{newnode}(Z), \text{newnode}(X), Z \neq X. \\
\text{excl}(x_1,n_4). 
\end{align*}
\]
Answer Sets

\[
\begin{align*}
&\text{newnode}(x_1). \quad \text{newnode}(x_2). \\
&\text{overloaded}(X) \leftarrow \text{DL}[\text{wired} \cup \text{connect}; \text{HighTrafficNode}](X). \\
&\text{connect}(X, Y) \leftarrow \text{newnode}(X), \text{DL}[\text{Node}](Y), \\
&\quad \text{not overloaded}(Y), \text{not excl}(X, Y). \\
&\text{excl}(X, Y) \leftarrow \text{connect}(X, Z), \text{DL}[\text{Node}](Y), Y \neq Z. \\
&\text{excl}(X, Y) \leftarrow \text{connect}(Z, Y), \text{newnode}(Z), \text{newnode}(X), Z \neq X. \\
&\text{excl}(x_1, n_4). \\
\end{align*}
\]

\[ M_1 = \{\text{connect}(x_1, n_1), \text{connect}(x_2, n_4), \ldots\} \] (new connections in blue)
Answer Sets

\[ \text{newnode}(x_1). \quad \text{newnode}(x_2). \]
\[ \text{overloaded}(X) \leftarrow \text{DL}[\text{wired} \cup \text{connect}; \text{HighTrafficNode}](X). \]
\[ \text{connect}(X, Y) \leftarrow \text{newnode}(X), \text{DL}[\text{Node}](Y), \]
\[ \quad \text{not overloaded}(Y), \quad \text{not excl}(X, Y). \]
\[ \text{excl}(X, Y) \leftarrow \text{connect}(X, Z), \text{DL}[\text{Node}](Y), Y \neq Z. \]
\[ \text{excl}(X, Y) \leftarrow \text{connect}(Z, Y), \text{newnode}(Z), \text{newnode}(X), Z \neq X. \]
\[ \text{excl}(x_1, n_4). \]

- \( M_1 = \{\text{connect}(x_1, n_1), \text{connect}(x_2, n_4), \ldots\} \),
- \( M_2 = \{\text{connect}(x_1, n_1), \text{connect}(x_2, n_5), \ldots\} \),
- \( M_3 = \{\text{connect}(x_1, n_5), \text{connect}(x_2, n_1), \ldots\} \),
- \( M_4 = \{\text{connect}(x_1, n_5), \text{connect}(x_2, n_4), \ldots\} \).
Further Semantics of $\mathcal{dl}$-Programs

- Different proposals for semantics, depending on refined consideration of evaluation of external access cf. [Wang et al., 2012]

- **Issue**: cyclic information flow

---

**Example**

\[ \Pi = (\emptyset, P), \text{ where } P = \{ p(a) \leftarrow \text{DL}[C \sqcup p; C](a) \}, \text{ has weak answer sets } \{ p(a) \}, \emptyset \]

- **$\text{strong answer sets}$**: treat monotone $\mathcal{dl}$-atoms (relative to $I$) like ordinary atoms

  Use $\text{strong reduct } s\Pi^I$: “... or $I \not\models \emptyset b_i$ for some non-monotone $\mathcal{dl}$-atom $b_i$.” (don’t touch monotone positive $\mathcal{dl}$-atoms $b_i$)

- **$\text{FLP-answer sets}$**: use FLP reduct [Fink and Pearce, 2010]
- (strong/weak) well-supported answer sets [Shen, 2011]
- well-founded semantics [E_ et al., 2011b]
Well-founded semantics

Lift well-founded semantics for ordinary to $\text{dl}$-programs \[E_\text{et al.}, 2011a\]

- Let $\gamma_\Pi(I) = \text{LM}(s\Pi^I)$ the least model of the strong reduct $s\Pi^I$.
- if all $\text{dl}$-atoms are monotone, $\gamma_\Pi$ is anti-monotone, thus $\gamma_\Pi^2$ is monotone and has a least fixpoint $\text{lfp}(\gamma_\Pi^2)$.

### Well-founded atoms of $\Pi = (\mathcal{O}, P)$

- $\text{WFS}(\Pi) = \text{lfp}(\gamma_\Pi^2)$ is the set of *well-founded atoms* of $\Pi$;
- For every ground atom $a$,
  - $\Pi \models_{w} a$ denotes that $a \in \text{WFS}(\Pi)$
  - $\Pi \models_{w} \neg a$ denotes that $a \notin \gamma_\Pi(\text{WFS}(\Pi))$
- Well-founded model:
  $$\text{WFM}(\Pi) = \text{WFS}(\Pi) \cup \{ \neg a \mid a \notin \gamma_\Pi(\text{WFS}(\Pi)) \}$$

- Well-founded and answer set semantics relate similarly as for ordinary programs
Network Example: Well-founded Semantics

\[
\begin{align*}
\text{newnode}(x_1). & \quad \text{newnode}(x_2). \\
\text{overloaded}(X) \leftarrow \text{DL}[\text{wired} \sqcup \text{connect}; \text{HighTrafficNode}](X). \\
\text{connect}(X, Y) \leftarrow \text{newnode}(X), \text{DL}[\text{Node}](Y), \\
& \quad \text{not overloaded}(Y), \text{not excl}(X, Y). \\
\text{excl}(X, Y) \leftarrow \text{connect}(X, Z), \text{DL}[\text{Node}](Y), Y \neq Z. \\
\text{excl}(X, Y) \leftarrow \text{connect}(Z, Y), \text{newnode}(Z), \text{newnode}(X), Z \neq X. \\
\text{excl}(x_1, n_4). 
\end{align*}
\]

- \( WFS(\Pi) = \{\text{overloaded}(n_2), \ldots\} \)
- \( \Pi \models_{\text{wf}} \neg \text{connect}(x_1, n_4), \ldots \)
- \( WFM(\Pi) = \{\text{overloaded}(n_2), \neg \text{connect}(x_1, n_4), \ldots\} \)
Some Semantical Properties

- **Conservative extension**: For $dI$-program $\Pi = (\mathcal{O}, P)$ without $dI$-atoms, the answer sets are the answer sets of $P$.

- **Existence**: Positive $dI$-programs without constraints (empty rule heads) always have an answer set.

- **Minimality**: answer sets of $\Pi$ are models, and strong answer sets are minimal models if all $dI$-atoms are monotone.

- **Uniqueness**: If $P$ is normal ($\lor$-free) and has a layered use of “not” (stratified) then it has a single strong answer set (if any).

- **Fixpoint Semantics**: Positive and stratified normal $dI$-programs possess fixpoint constructions of the strong answer set.
Computational Complexity

Deciding strong answer set existence for \(\frac{\text{normal}}{\text{disjunctive}}\) dl-programs (completeness results)

<table>
<thead>
<tr>
<th>(\Pi = (O, P))</th>
<th>no dl-atoms</th>
<th>(O) in (SHIF(D))</th>
<th>(O) in (SHOIN(D))</th>
</tr>
</thead>
<tbody>
<tr>
<td>positive</td>
<td>(\text{ExpTime})</td>
<td>(\text{ExpTime})</td>
<td>(\text{NExpTime})</td>
</tr>
<tr>
<td>stratified</td>
<td>(\text{ExpTime}^{NP})</td>
<td>(\text{ExpTime}^{NP})</td>
<td>(\text{PTIME}^{NExpTime})</td>
</tr>
<tr>
<td>general</td>
<td>(\text{NExpTime}^{NP})</td>
<td>(\text{NExpTime}^{NP})</td>
<td>(\text{PTIME}^{NExpTime})</td>
</tr>
</tbody>
</table>

Note:

- Satisfiability in \(SHIF(D) / SHOIN(D)\) is \(\text{ExpTime} / \text{NExpTime}\)-complete.
- **Key observation:** The number of ground dl-atoms is polynomial
- \(\text{PTIME}^{NExpTime} = \text{NP}^{NExpTime} = \text{PSPACE}^{NP}\) is less powerful than Answer Sets for disjunctive programs (\(\equiv \text{NExpTime}^{NP}\))
- Same complexity as for no dl-atoms, if \(O\) is from a polynomial DL class (e.g., OWL 2 Profiles RL, EL, QL)
Applications

dl-programs facilitate some advanced reasoning tasks

- **Default Reasoning**
  
  Poole-style and Reiter-style Default Logic over DL knowledge bases (for restricted fragments, to the effect of *Terminological Default Logic* [Baader and Hollunder, 1995]).

  Front-end [Dao-Tran *et al.*, 2009]

- **Closed World Reasoning**
  
  Emulate CWA and *Extended CWA (ECWA)* on top of a DL KB.

- **Minimal Model Reasoning**
  
  Single out “minimal” models of a DL KB
Example: Reviewer Candidate Selection using Defaults

\[
O = \{ \neg \text{ConflictingReviewer} \sqsubseteq \text{CandidateReviewer}, \\
\text{Senior}(joe), \text{Senior}(bob), \text{ConflictingReviewer}(bob) \}.
\]

- Besides known candidate reviewers, by default also every senior author is a candidate reviewer (unless a conflict is apparent).
- This is mimicked by the following dl-program:

\[
\begin{align*}
\text{r}_0 &: \text{cand\_rev}(P) \leftarrow \text{DL[CandidateReviewer]}; \\
\text{r}_1 &: \text{cand\_rev}(P) \leftarrow \text{DL[Senior]}(P), \text{not conflict}(P); \\
\text{r}_2 &: \text{conflict}(P) \leftarrow \text{DL[CandidateReviewer} \sqcup \text{cand\_rev}; \\
&\quad \text{ConflictingReviewer]}(P).
\end{align*}
\]

- Under Answer Set Semantics, \text{r}_2 effects maximal application of \text{r}_1.
- Single strong answer set: \( I = \{ \text{cand\_rev}(joe), \text{conflict}(bob) \} \); reduct \( sP^I \)
Closed World Assumption (CWA)

Reiter’s Closed World Assumption (CWA)

For ground atom $p(c)$, infer $\neg p(c)$ if $KB \not\models p(c)$

- Express CWA for concepts $C_1, \ldots, C_k$ wrt. individuals in $L$:
  
  use predicate $c_i$ for concept $C_i$ in the program, and $\overline{c_i}$ for its negation:

  \[
  \overline{c_1}(X) \leftarrow \text{not } DL[C_1](X) \\
  \ldots \\
  \overline{c_k}(X) \leftarrow \text{not } DL[C_k](X)
  \]

- CWA for roles $R$: similar
Query Answering under CWA

Example: \( \mathcal{O} = \{ \text{SparklingWine(“VeuveCliquot”)}, \\
\quad (\text{SparklingWine} \sqcap \neg \text{WhiteWine})(“Lambrusco”) \} \).

Query: \( \text{WhiteWine(“VeuveCliquot”) (Y/N)?} \)

Add CWA-literals to \( \mathcal{O} \):

\[
\begin{align*}
\overline{sp}(X) & \leftarrow \text{not } DL[\text{SparklingWine}](X) \\
\overline{ww}(X) & \leftarrow \text{not } DL[\text{WhiteWine}](X) \\
ww(X) & \leftarrow DL[\text{SparklingWine} \sqcup \overline{sp}, \text{WhiteWine} \sqcup \overline{ww}; \text{WhiteWine}](X)
\end{align*}
\]

Ask whether \( \Pi \models \overline{ww}(“VeuveCliquot”) \) or
\( \Pi \models \overline{ww}(“VeuveCliquot”) \)}
Extended CWA

- CWA can be inconsistent (disjunctive knowledge)

- Example:
  Knowledge base

  \[ O \equiv \{ \text{Artist(“Jody”), Artist} \equiv \text{Painter} \land \text{Singer} \} \]

  - CWA for Painter, Singer adds

    \[ \neg \text{Painter(“Jody”)}, \neg \text{Singer(“Jody”)} \]

  - This implies \[ \neg \text{Artist(“Jody”)} \]
Minimal Models

- ECWA [Gelfond et al., 1986] singles out “minimal” models of \( \mathcal{O} \) wrt Painter and Singer (UNA in \( \mathcal{O} \) on ABox):

\[
\bar{p}(X) \leftarrow \neg p(X) \\
\bar{s}(X) \leftarrow \neg s(X) \\
p(X) \leftarrow DL[\text{Painter} \cup \bar{p}, \text{Singer} \cup \bar{s}; \text{Painter}](X) \\
s(X) \leftarrow DL[\text{Painter} \cup \bar{p}, \text{Singer} \cup \bar{s}; \text{Singer}](X) \\
\leftarrow DL[; \top \sqsubseteq \bot] \quad /* \text{no answer set of } \mathcal{O} \text{ is unsatisfiable } */
\]

Answer sets:

\[
M_1 = \{ p(\text{“Jody”}), \bar{s}(\text{“Jody”}) \}, \\
M_2 = \{ s(\text{“Jody”}), \bar{p}(\text{“Jody”}) \}
\]

- Extendible to keep concepts “fixed” in a DL knowledge base \( \mathcal{O} \)

\(
\leadsto \text{ECWA}(\text{ont}; P; Q; Z) \quad \text{(assuming UNA, domain closure)}
\)
HEX-Programs

- Motivated to meet needs of heterogeneous data access on the Web
- Generalize d1- programs
- Allow to access sources of whatever type (abstract modeling)

**Features:**

- **Hilog-style atoms:** variables for predicate names (syntactic sugar)
  
  E.g., \( P(X, Z) \leftarrow P(X, Y), P(Y, Z) \)  

  \( \Rightarrow \) reify atoms \( p(t_1, \ldots, t_n) \) to \( (p, t_1 \ldots, t_n) \)

- **External atoms:** access to external sources, e.g.,
  
  - ontologies (RDF, OWL, ...)
  - planners,
  - data structures (libraries, built-ins)
HEX Programs (cont’d)

\[
\text{invites}(john, X) \lor \text{skip}(X) \leftarrow X \neq john, \\
\ \ \ \ \ \ \ & \text{DL}\_\text{Query}[\text{my}\_\text{ontology}, \text{relativeOf}](john, X).
\]

\[
some\text{Invited} \leftarrow \text{invites}(john, X).
\]

\[
\leftarrow \text{not someInvited}.
\]

\[
\leftarrow \&\text{degs[invites]}(\text{Min}, \text{Max}), \text{Max} > 2.
\]

Example

**Input:** Data about John’s relatives (from an ontology)

**Output:** Possible picks for persons John might want to invite, according to some constraints (some evaluated externally)
\[ \text{\&DL\_Query}[\text{my\_ontology, relativeOf}](\text{john, } X) \]  
\[ \text{\&degs}[\text{invites}](\text{Min, Max}) \]

### External Atom

**External atoms** are of the form

\[ \&g[\vec{Y}](\vec{X}) \]

where \( \vec{Y} = Y_1, \ldots, Y_n \) and \( \vec{X} = X_1, \ldots, X_m \) are two lists of terms (**input/output** list), and \( \&g \) is an external predicate name.

- External atoms may occur only in rule bodies
- \( \&g \) has an associated function

\[ f_{\&g} : 2^{HB_p} \times C^{n+m} \rightarrow \{0, 1\} \]

mapping each \((I, y_1 \ldots, y_n, x_1, \ldots, x_m)\) to either 0 or 1, where \( I \) is an interpretation and \( x_i, y_j \) are ground terms (no functions)

- Typically, inputs \( y_i \) are predicate names, \( x_j \) are individuals
External Atoms – Examples

Example

\&DL_{Query} \text{ corresponds to } f_{\&DL_{Query}}.

- Informally, \&DL_{Query}[my\_ontology, relativeOf](john, c) is true if relativeOf(john, c) is provable in my\_ontology.

- This is formally captured via \( f_{\&DL_{Query}} \):

  For a given interpretation \( I \),
  \[
  I \models \&DL_{Query}[my\_ontology, relativeOf](john, c)
  \text{ iff }
  f_{\&DL_{Query}}(I, my\_ontology, relativeOf, john, c) = 1
  \]

\( d\ell \)-atoms \( DL[\lambda; Q](\vec{t}) \) can be served with a uniform external atom (encode \( \lambda, Q, \vec{t} \) in input) \( \sim \) DL-plugin
HEX Programs

- HEX-rules and HEX-programs are like d1-programs, with external atoms in place of d1-atoms

- Semantics is defined by Herbrand models via grounding

- Answer sets are defined using the FLP reduct [Faber et al., 2004]: \( I \) is minimal model of \( fP^I = \{ H \leftarrow \text{Body} \in \text{grnd}(P) \mid I \models \text{Body} \} \)
  (advantage: minimality of models is for free)

Example (RDF import)

&\textit{rdf} provides RDF access

- Informally,
  \[ \text{triple}(X, Y, Z) \leftarrow \&\textit{rdf}[\text{uri}](X, Y, Z) \]
  imports all triples from RDF file \textit{uri} into the program
HEX Programs, cont’d

Example (Generic CWA for concepts)

- use $map(P, C)$ to link rule predicates and concepts: $map(w, Wine)$
- use $neg(P, N)$ to say $N$ is the complement of $P$: $neg(w, \overline{w})$
- single HEX rule:
  \[ N(X) \leftarrow map(P, Y), \text{neg}(P, N), \text{not } DL[Y](X) \]
- Higher-order instance:
  \[ \overline{w}(X) \leftarrow map(w, Wine), \text{neg}(w, \overline{w}), \text{not } DL[Wine](X) \]
Implementation: dlvhex

http://www.kr.tuwien.ac.at/research/systems/dlvhex/

- **plugin architecture** (C++ code)
- **dl-programs**: DL-Plugin (RacerPro)
- **rewriting approach**:
  - (HEX2ASP) use replacement atoms: $\&g[\vec{x}](\vec{y}) \leadsto e_{\&g[\vec{y}]}(\vec{x})$
  - a-posteriori compatibility check for replacement atoms
- **new algorithm (HEX-2)**
  - native model building
  - conflict-driven learning [E_ et al., 2012a]
  - unfounded set checking [E_ et al., 2012b]
- **issue**: new values from external sources (value invention)
  - framework: liberal domain-expansion safety [E_ et al., 2013]
Applications

- querying data and ontologies [Hoehndorf et al., 2007], [Marano et al., 2010]
- e-government [Zirtiloğlu and Yolum, 2008]
- fuzzy answer set programming [Nieuwenborgh et al., 2007a]
- multi-context reasoning [Brewka and E_, 2007]
- user interface adaptation [Zakraoui and Zagler, 2011]
- reasoning about actions and planning [Nieuwenborgh et al., 2007b], [Basol et al., 2010]
- ...
Conclusion

- dl-programs realize Loose Coupling
  - rules on top of ontologies (query access)
  - bidirectional information flow
- Different semantics for dl-programs (cyclic dependencies)
- HEX-programs generalize dl-programs for flexible data access
- Many further issues:
  - inconsistency management
  - optimization and implementation
  - relation to other formalisms (e.g., hybrid MKNF KBs)
  - ...
Franz Baader and Bernhard Hollunder.
Embedding defaults into terminological knowledge representation formalisms.

Selen Basol, Ozan Erdem, Michael Fink, and Giovambattista Ianni.
HEX Programs with Action Atoms.

Gerd Brewka and Thomas Eiter.
Equilibria in Heterogeneous Nonmonotonic Multi-Context Systems.

Minh Dao-Tran, Thomas Eiter, and Thomas Krennwallner.
Realizing default logic over description logic knowledge bases.
References II


Thomas Eiter, Giovambattista Ianni, Thomas Lukasiewicz, Roman Schindlauer, and Hans Tompits.

T. Eiter, G. Ianni, T. Lukasiewicz, and R. Schindlauer.

Thomas Eiter, Giovambattista Ianni, Thomas Lukasiewicz, and Roman Schindlauer.


References IV

Michael Fink and David Pearce.
A logical semantics for description logic programs.

M. Gelfond, H. Przymusinska, and T. Przymusinski.
The Extended Closed World Assumption and its Relationship to Paralell Circumscription.

Robert Hoehndorf, Frank Loebe, Janet Kelso, and Heinrich Herre.
Representing default knowledge in biomedical ontologies: application to the integration of anatomy and phenotype ontologies.

Marco Marano, Philipp Obermeier, and Axel Polleres.
Processing RIF and OWL2RL within DLVHEX.

Davy Van Nieuwenborgh, Martine De Cock, and Dirk Vermeir.
Computing fuzzy answer sets using dlvhex.
Davy Van Nieuwenborgh, Thomas Eiter, and Dirk Vermeir.
Conditional planning with external functions.

Yi-Dong Shen.
Well-supported semantics for description logic programs.

Yisong Wang, Jia-Huai You, Li-Yan Yuan, Yi-Dong Shen, and Mingyi Zhang.
The loop formula based semantics of description logic programs.

Jesia Zakraoui and Wolfgang L. Zagler.
A logical approach to web user interface adaptation.

Hande Zirtiloğlu and Pinar Yolum.
Ranking semantic information for e-government: complaints management.
In 1st International Workshop on Ontology-supported Business Intelligence (OBI’08), number 5 in OBI’08, page 7. ACM, 2008.
Appendix: Reviewer selection [E_ et al., 2008b] (adapted)

\[
paper(p_1); \ kw(p_1, ”Semantic\_Web”); \tag{3}
\]
\[
paper(p_2); \ kw(p_2, ”Bioinformatics”); \ kw(p_2, ASP); \tag{4}
\]
\[
kw(P, K_2) \leftarrow kw(P, K_1), DL[hasMember](S, K_1),
DL[hasMember](S, K_2); \tag{5}
\]
\[
paperArea(P, A) \leftarrow DL[keywords \cup kw; inArea](P, A); \tag{6}
\]
\[
cand\_rev(X, P) \leftarrow paperArea(P, A), DL[CandidateReviewer](X),
DL[expert](X, A); \tag{7}
\]
\[
assign(X, P) \leftarrow cand\_rev(X, P), not \ unassign(X, P); \tag{8}
\]
\[
unassign(Y, P) \leftarrow cand\_rev(Y, P), assign(X, P), X \neq Y; \tag{9}
\]
\[
has\_rev(P) \leftarrow assign(X, P); \tag{10}
\]
\[
error(P) \leftarrow paper(P), not \ has\_rev(P). \tag{11}
\]

- Determine paper area with enhanced keyword info (key word clusters) (5), (6)
- Use ontology to determine candidate reviewers (7)
- (8)–(11) is a plain ASP selection program (choose one cand\_rev per paper)
Reviewer selection (ctd.)

- Answer sets of $\Pi$ depend on the instances of $\text{hasMember}$, $\text{keywords}$, $\text{inArea}$, expert $\text{CandidateReviewer}$

- Suppose in $\mathcal{O}$ $\text{expert}(\text{jim},"A1")$, $\text{expert}(\text{tim},"A1")$, $\text{expert}(\text{sue},"A2")$
  $\text{ReviewerCandidate}(\text{jim})$, $\text{ReviewerCandidate}(\text{tim})$, $\text{ReviewerCandidate}(\text{sue},"LP")$
  $\text{hasMember}(c_1, "\text{ASP}")$, $\text{hasMember}(c_1, "LP")$ are true (named clusters)

- Further, that $\text{inArea}(p_1,"A1")$ is true and $\text{inArea}(p_2,"A2")$ is true after asserting $\text{keywords}(p_2,"LP")$.

- $M = \begin{cases} 
(1), (2), kw(p_2,"LP"), \text{paperArea}(p_1,"A1"), \text{paperArea}(p_2,"A2"), 
\text{cand}_\text{rev}(p_1,jim), \text{cand}_\text{rev}(p_1,\text{tim}), \text{cand}_\text{rev}(p_2,\text{sue}), 
\text{assign}(\text{jim},p_1), \text{unassign}(\text{tim},p_1), \text{assign}(\text{sue},p_2), 
\text{has}_\text{rev}(p_1), \text{has}_\text{rev}(p_2) 
\end{cases}$

  is a (strong) answer set of $\Pi$. 
Example: Reviewer selection (ctd.) /2

\[ M = \{ (1), (2), \text{kw}(p_2, "LP"), \text{paperArea}(p_1, "A1"), \text{paperArea}(p_2, "A2"), \]  
\[ \quad \text{cand}_rev(p_1, jim), \text{cand}_rev(p_1, tim), \text{cand}_rev(p_2, sue), \]  
\[ \quad \text{assign}(jim, p_1), \text{unassign}(tim, p_1), \text{assign}(sue, p_2), \]  
\[ \quad \text{has}_rev(p_1), \text{has}_rev(p_2) \} \]

- Part 0: Facts
- Part 1: \text{kw}, \text{paperArea}, (LP, ASP in same cluster)
- Part 2 \text{cand}_rev
- Part 3: choice for \text{assign}; \text{has}_rev; reduct \text{sP}^M \text{ (relevant part)}

Note: second (strong) answer set is \[ M = \{ \ldots \text{unassign}(jim, p_1), \text{assign}(tim, p_1) \ldots \} \]