

Part 2

Hybrid Reasoning

Integration of Rules and Description Logic

Włodzimierz Drabent

Linköping University (Sweden)
Institute of Computer Science, Polish Academy of Sciences

Version of August 30, 2010

1 / 51

Outline

- ▶ Integration: homogeneous vs. heterogeneous
- ▶ Heterogeneous integration
 - ▶ Preliminaries
 - ▶ Loose & tight coupling; definitions & examples
 - ▶ Further technical issues
- ▶ Overview of existing approaches
 - ▶ Operational semantics
- ▶ Conclusions

3 / 51

Inference for Semantic Web

<http://www.w3.org/standards/semanticweb/inference>:

Inference – reasoning over data

ontologies
OWL, OWL 2

rule sets
RIF

Description Logics (DL)
first order logics (FOL)

Logic Programming (LP)

Integration of both approaches needed

Our main interest

monotonic reasoning

non-monotonic reasoning

2 / 51

Approaches to integration, **homogeneous**

A DL + a rule formalism \rightsquigarrow a more general one

Original languages – sublanguages of the new one

Their semantics preserved (the generalization is faithful).

- DLP (description logic programs) [Grosz+].
A restricted DL, expressible in positive Datalog.
OWL 2 RL profile.
- SWRL – OWL DL + FOL implications.
DL-safe rules [Motik,Sattler,Studer'05]
- ▶ F-logic
- ▶ Hybrid MKNF Knowledge Bases
- ▶ Quantified Equilibrium Logic

Reasoning:

Using existing tools for DL / LP reasoning – often impossible

4 / 51

Approaches to integration, heterogeneous

A DL + a rule formalism made to cooperate

Distinction – predicates defined by the rules (rule predicates)
 DL (external predicates or DL-predicates)

Interface DL – rules: external predicates allowed in the rules.

Often possible:
 implementation based on reasoning tools for DL and LP.

Our presentation:

Heterogeneous integration

first a generic solution, then relating it to existing approaches.

A generic approach to integrate LP with FOL theories.
 Not restricted to Datalog

Two semantics of LP – **WFS**, **SMS**

Two kinds of integration (loose, tight)

Preliminaries

A **hybrid program** (P, T)

P – a logic program, T – a FO theory

Predicate symbols $\mathcal{P} = \mathcal{P}_P \cup \mathcal{P}_T$

– rule (resp. external) predicates

Rule (resp. external) literals, etc

Function symbols \mathcal{F}

External predicates may occur in rule bodies. Not vice versa.

(T influences P , not vice versa. But...)

T – a black box.

Preliminaries (2)

Detail: Symbol \neg – two kinds of negation.

- ▶ With an external predicate – negation of FOL.
- ▶ With a rule predicate - non-monotonic negation of LP.

Feasible in practice two symbols instead: \neg , *not*.

Loose and tight coupling

Different semantics of integration

Semantics of a rule $C = H \leftarrow \dots, L, \dots$

L – external literal

Loose coupling – Interpretations of \mathcal{P}_P ,
logical consequences of T for \mathcal{P}_T

$L^{\mathcal{I}}$ true if $T \models L^{\mathcal{I}}$, otherwise L false

L treated like a procedure call

Tight coupling – Interpretations of $\mathcal{P}_P \cup \mathcal{P}_T$

An interpretation applied to all the body literals

9 / 51

Ex. Reasoning by cases (2)

No negation in P . Reasoning monotonic.

$P \cup T \models student(ann)$. (In standard FOL)

Tight coupling compatible with FOL.

Loose coupling – not.

11 / 51

Ex. Reasoning by cases

Ex.: T : $Project(X) \vee Lecture(X) \leftrightarrow Course(X)$
 $Lecture(cs1) \quad Course(cs3)$
 $Project(cs2) \quad \dots$

P : $student(X) \leftarrow enrolled(X, Y), Lecture(Y)$
 $student(X) \leftarrow enrolled(X, Y), Project(Y)$
 $enrolled(ann, cs3)$
 \dots

$T \models Course(cs3)$, but $T \not\models Lecture(cs3)$, $T \not\models Project(cs3)$.

Loose coupling: $Lecture(cs3)$, $Project(cs3)$ are false
 $student(ann)$ **cannot** be derived.

Tight coupling: In any interpretation, $Lecture(cs3)$ is true or
 $Project(cs3)$ is true. In each case $student(ann)$ **is** derived.

10 / 51

Df. Integration, loose coupling

(P, T)

$P/\models T$ – $ground(P)$ with deleted

each rule with an external literal L , s.t. $T \not\models L$,
 all external literals in the remaining rules (for them $T \models L$)

- ▶ The well-founded model of $P/\models T$
 – the *well-founded model* of (P, T) under loose coupling
- ▶ A stable model of $P/\models T$
 – a *stable model* of (P, T) under loose coupling

Abbreviations: WFSL, SMSL

12 / 51

Example: Two-person game

P:

$w(X) :- m(X,Y), \neg w(Y).$

$m(a,b).$

$m(b,a).$

$m(a,c).$

$m(c,d).$

$m(d,e).$

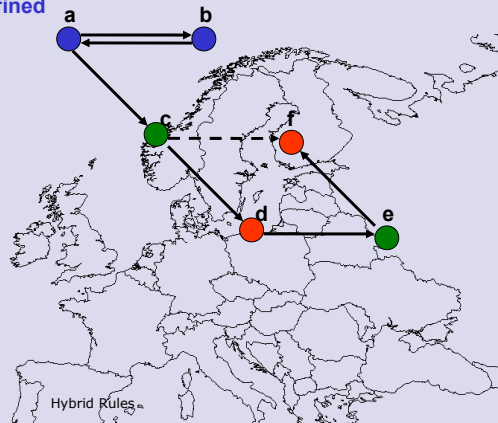
$m(c,f) :- \neg Fi(f).$

$m(e,f) :- E(f).$

Ontology:

$Fi \sqsubseteq E$

true
false
undefined



3

Ex. loose coupling

Insurance should be arranged for X
if from T it does not follow that X is insured.

$P:$ $insured(X) \leftarrow Insured(X).$
 $insurance_needed(X) \leftarrow \neg insured(X).$

Insured – external predicate (the others – rule predicates).

Under loose coupling (and WFS or SMS) – works as required:

If $T \not\models Insured(a)$ then $insurance_needed(a)$ derived.

Not derived under tight coupling

if $Insured(a)$ is true in some models of T .

19 / 51

under loose coupling...

$w(X) \leftarrow m(X,Y), \neg w(Y)$

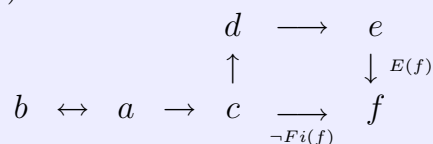
$m(b,a) \quad m(c,f) \leftarrow \neg Fi(f)$

$m(a,b) \quad m(e,f) \leftarrow E(f)$

$m(a,c)$

$m(c,d)$

$m(d,e)$



Assume $T \not\models E(f)$ and $T \not\models \neg Fi(f)$

In the well-founded model $WF(P/\models T)$ we have

$\neg m(c,f), \neg m(e,f),$

and thus $\neg w(f), \neg w(e), w(d), \neg w(c), w(a), \neg w(b)$

Not as intended.

18 / 51

Loose & tight coupling

are complementary.

E.g. loose c. – impossible reasoning by cases;

tight c. – impossible querying $T \stackrel{?}{\models} L$.

The existing approaches – either tight or loose.

Needed – approaches that include both.

20 / 51

A non Datalog program

The game ... Each node visited at most once

$win(x, h)$ – x winning without visiting any node from h .

$w(X) \leftarrow win(X, [])$

$win(X, History) \leftarrow move(X, Y, History), \neg win(Y, [X|History])$

$move(A, B, History) \leftarrow m(A, B), \neg member(B, History)$

$m(b, a) \quad m(c, f) \leftarrow \neg Fi(f)$

$m(a, b) \quad m(e, f) \leftarrow E(f)$

$m(a, c) \quad \dots$

$m(c, d) \quad \uparrow$

$m(d, e) \quad b \leftrightarrow a \rightarrow c \rightarrow \dots$

WFST entails ... $w(c)$, $w(a)$ and $w(b)$.

Two kinds of interpretations

FOL – arbitrary interpretation domains

Unnamed objects possible

WFS, SMS – Herbrand interpretations

No unnamed objects

Integration – we use both kinds of interpretations

Interface – ground literals, $M_0 \stackrel{?}{\models} L, T \stackrel{?}{\models} L$

Result: **Imperfect compatibility** with FOL, even for tight coupling

Ex.: $T = \{\exists x.q(x)\}, P = \{p \leftarrow q(x)\}$.

$P \cup T \models p$

$(P, T) \not\models_{\text{wf}} p$ (the same for SMST)

as there exist models of T in which each ground $q(t)$ is false.

Compatibility with First Order Logic

Tight coupling (WFST, SMST)

P rules without negation, A – ground atom.

If $(P, T) \models_{\text{wf}} A$ then $P \cup T \models A$.

If $P \cup T \models A$ then $I \models_3 A$, for each well-founded model I of P based on a Herbrand interpretation M_0 of T .

Equality

In Herbrand interpretations, distinct ground terms never equal.

T may imply $a = b$, but $a \neq b$ in any Herbrand interpretation.

Ex.: $P = \{p(a)\}$. Both $p(a)$, $\neg p(b)$ in WFS & SMS, loose & tight c.

Not a bug, a feature.

(Alternative: non-standard semantics of T .)

If T makes t_1, t_2 equal then P should treat them in the same way.

Such P called **congruent**.

The problem well-known in Constraint Logic Programming.

(Smoothly dealt with, hardly made explicit.)

Ex.: Program $WINH$, where $T \models c = g$, not congruent:

$$WINH \models_{wf} w(c), \neg w(g).$$

Assume that T does not imply $c_1 = c_2$

for any other pair of constants $\{c_1, c_2\} \neq \{c, g\}$.

To make $WINH$ congruent, add to P

	already in P
$m(a, g)$	$m(a, c)$
$m(g, d)$	$m(c, d)$
$m(g, f) \leftarrow \neg Fi(f)$	$m(c, f) \leftarrow \neg Fi(f)$

To make it congruent independently from T :

$$w(X) \leftarrow X = X', Y = Y', m(X', Y'), \neg w(Y).$$

where $X = X', Y = Y'$ – external literals

Existing approaches, loose coupling

dl-programs [Eiter, Ianni, Lukasiewicz, Schindlauer, Tompits]

SMSL, but also WFSL, for disjunctive Datalog

External theories – DL (those of OWL Lite, OWL DL)

Main extensions w.r.t. SMSL:

- ▶ disjunctive rules
- ▶ classical negation
- ▶ dl-atoms – querying a modified T
incl. subsumption querying

Generalizations exist (HEX-programs)

Decidability

Assume Herbrand Universe is finite (\mathcal{F} – finite set of constants).

Then WFSL, SMSL, WFST, SMST **decidable** if T decidable.

In FOL, **undecidable** whether $P \cup T \models A$

for most DL of interest and P without negation

The difference: We have only Herbrand interpretations for \mathcal{P}_P

More details:

WFSL, SMSL, decidable if $T \stackrel{?}{\models} L$ decidable
(for any external literal L in $ground(P)$).

WFST, SMST decidable
if satisfiability of $T \cup \{L_1, \dots, L_n\}$ decidable
(L_i or $\neg L_i$ in $ground(P)$).

dl-atoms (loose coupling, dl-programs)

Ex.: $D = DL[keyw \uplus kw; inA](P, A)$

$keyw, inA$ – external predicates, kw – rule predicate

The meaning of D (in an interpretation I of \mathcal{P}_P):

query $inA(P, A)$ applied to $T \cup \{keyw(c) \mid kw(c) \in I\}$

Possible in a dl-atom: multiple modifications of T (maybe 0);
other kinds of modifications:

\uplus	$T \cup \{keyw(c) \mid kw(c) \in I\}$
\cup	$T \cup \{\neg keyw(c) \mid kw(c) \in I\}$
\cap	$T \cup \{\neg keyw(c) \mid kw(c) \notin I\}$

dl-atoms (loose coupling, dl-programs) [duplicate]

Ex.: $D = DL[keyw \uplus kw; inA](P, A)$

$keyw, inA$ – external predicates, kw – rule predicate

The meaning of D (in an interpretation I of \mathcal{P}_P):

query $inA(P, A)$ applied to $T \cup \{keyw(\bar{c}) \mid kw(\bar{c}) \in I\}$

Possible in a dl-atom: multiple modifications of T (maybe 0);
other kinds of modifications:

$$\uplus \quad T \cup \{keyw(\bar{c}) \mid kw(\bar{c}) \in I\}$$

$$\uplus \quad T \cup \{\neg keyw(\bar{c}) \mid kw(\bar{c}) \in I\}$$

$$\sqcap \quad T \cup \{\neg keyw(\bar{c}) \mid kw(\bar{c}) \notin I\}$$

Existing approaches, loose coupling

Prolog

SWI Prolog, XSB –

possible to call a DL reasoner from a program.

Operational semantics (loose c., dl-programs)

Idea:

Use an implementation of LP with negation (for WFS or SMS)

Whenever the logical value of a ground external L needed
– run a reasoner for the external theory.

Implementation of dl-programs

employs DLV for SMS, RACER for DL.

Existing approaches, tight coupling

CARIN [Levy,Rousset], \mathcal{AL} -log [Donini+]

Positive Datalog + certain DL

semantics + Unique Name Assumption (UNA)

($a \neq b$ for any distinct a, b)

(arbitrary interpretations for rules, in contrast to out approach)

Conditions on rules / DL for decidability

(Can also be seen as homogeneous approach)

Existing approaches, tight coupling

DL rules [Krötsch,Rudolph,Hitzler,ECAI08]
 Rules expressible in a DL (\mathcal{SROIQ}).

Syntactic restrictions on rules

FOL semantics

(Can also be seen as homogeneous approach)

r-hybrid KBs, ... – operational semantics

Herbrand base over constants and external predicates from P .
 (Intuitively, the interface between P and T .)

Enumerating all Herbrand interpretations I .

For each I

Check satisfiability of $T \cup I \cup \{\neg A \mid A \notin I\}$

If YES, construct a Datalog program (similar to) P/I

look for a stable model of P/I

The number of interpretations to enumerate

– exponential in the number of constants in P .

Seems that no implementation exists.

Existing approaches, tight coupling, SMS

r-hybrid KBs, $\mathcal{DL}+log$ [Rosati]

Disjunctive Datalog

External predicates permitted in rule heads

not permitted in negative literals

Interpretations with Standard Name Assumption:

infinite set \mathcal{F} of constants with a fixed meaning,

1-1 correspondence constants \leftrightarrow domain elements

(Equivalent to Herbrand interpretations over \mathcal{F})

DL-safeness for decidability.

(Each variable has to occur in a positive rule literal in the body,
 unless it occurs only in the external literals of the body.)

A version without UNA [Rosati PPSWR'05]

with stable models over arbitrary interpretation domains

Existing approaches, tight coupling, SMS (2)

f-hybrid knowledge bases (fKBs) [Feier,Heymans]

Disjunctive Datalog

External predicates permitted in rule heads

Open answer set programming.

(Non fixed set \mathcal{F} of constants, maybe ∞ .)

Syntactic restrictions on the form of rules (for decidability)

The chosen DL (\mathcal{SHOQ}) can be encoded by the rules

Thus hybrid program can be translated into logic programs
 and a reasoner for disjunctive Datalog used

(So fKBs can also be seen as homogeneous approach)

Existing approaches, tight coupling, SMS (3)

Tightly coupled dl-programs [Lukasiewicz]

Disjunctive Datalog

External predicates permitted in rule heads

No syntactic restrictions on rules

Specific semantics. Roughly: minimal models of T

Ex.: $P = \{p \leftarrow \neg q(b)\}$ $T = \{q(a)\}$ ($\mathcal{P}_P = \{p\}$, $\mathcal{P}_T = \{q\}$)
 (P, T) entails p , despite $T \not\models \neg q(b)$

37 / 51

Existing approaches, tight coupling, WFS

HD-rules [D..., Maluszynski]

Follows our definition of WFST.

Permits non-constant function symbols.

In the rules: A class of formulae over \mathcal{P}_T (not only literals).
 Called **constraints**.

38 / 51

HD-rules, operational semantics (WFST)

Based on ideas from CLP (constraint logic programming) and constructive negation.

Computations – top down, query driven.

Upper layer – deals with P ,
 tree of trees (like in SLS- and SLDNF-resolution).

Lower layer – deals with T ,
 querying an external theory reasoner.

For positive P , coincides with the algorithm for \mathcal{AL} -log.

Upper layer – a single tree.

39 / 51

Example, positive program

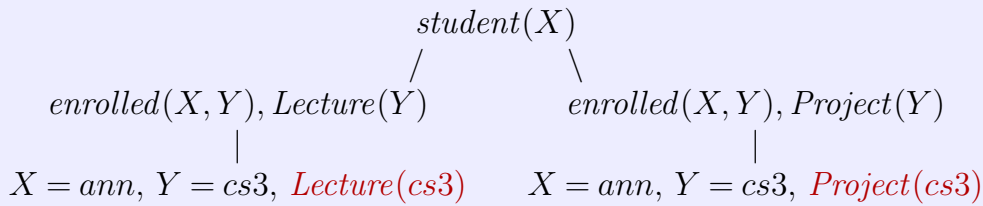
T : $Project(X) \vee Lecture(X) \leftrightarrow Course(X)$
 $Lecture(cs1)$ $Course(cs3)$
 $Project(cs2)$ \dots

P : $student(X) \leftarrow enrolled(X, Y), Lecture(Y)$
 $student(X) \leftarrow enrolled(X, Y), Project(Y)$
 $enrolled(ann, cs3)$
 \dots

Upper layer:

$$\begin{array}{c}
 student(X) \\
 / \quad \backslash \\
 enrolled(X, Y), Lecture(Y) \quad enrolled(X, Y), Project(Y) \\
 | \qquad \qquad \qquad | \\
 X = ann, Y = cs3, Lecture(cs3) \quad X = ann, Y = cs3, Project(cs3)
 \end{array}$$

40 / 51



From the tree, $\text{student}(X)$ implied by

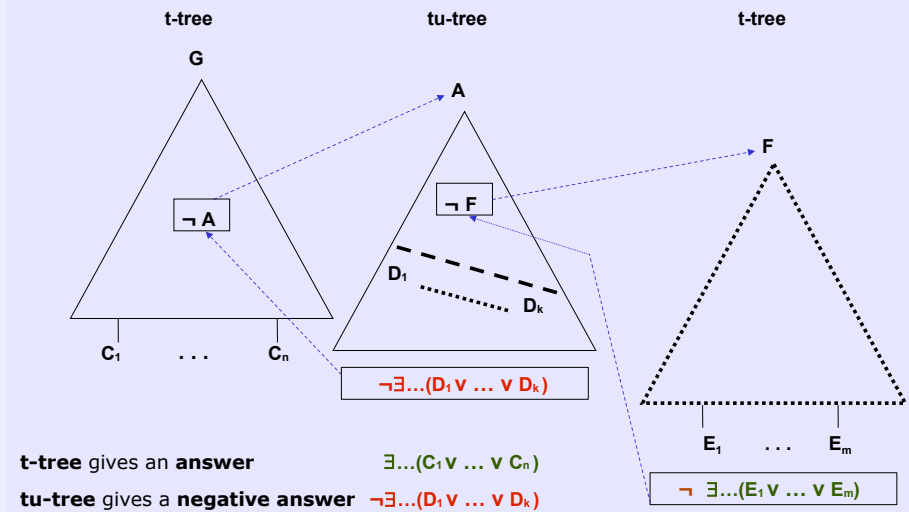
$$(\exists Y. X = \text{ann}, Y = \text{cs3}, \text{Lecture}(\text{cs3})) \vee (\exists Y. X = \text{ann}, Y = \text{cs3}, \text{Project}(\text{cs3}))$$

Hence $\text{Lecture}(\text{cs3}) \vee \text{Project}(\text{cs3})$ implies $\text{student}(\text{ann})$.

A query issued $T \stackrel{?}{\models} \text{Lecture}(\text{cs3}) \vee \text{Project}(\text{cs3})$.

Answer YES. $X = \text{ann}$ returned as an answer to $\text{student}(X)$.

Upper layer



HD-rules, operational semantics, upper layer

SLS-resolution: goals (conjunctions of literals) + substitutions.

Our generalization:

- goals – conjunctions of rule literals and constraints;
- constraints – over $\mathcal{P}_T \cup \{=\}$,
- = – syntactic equality.

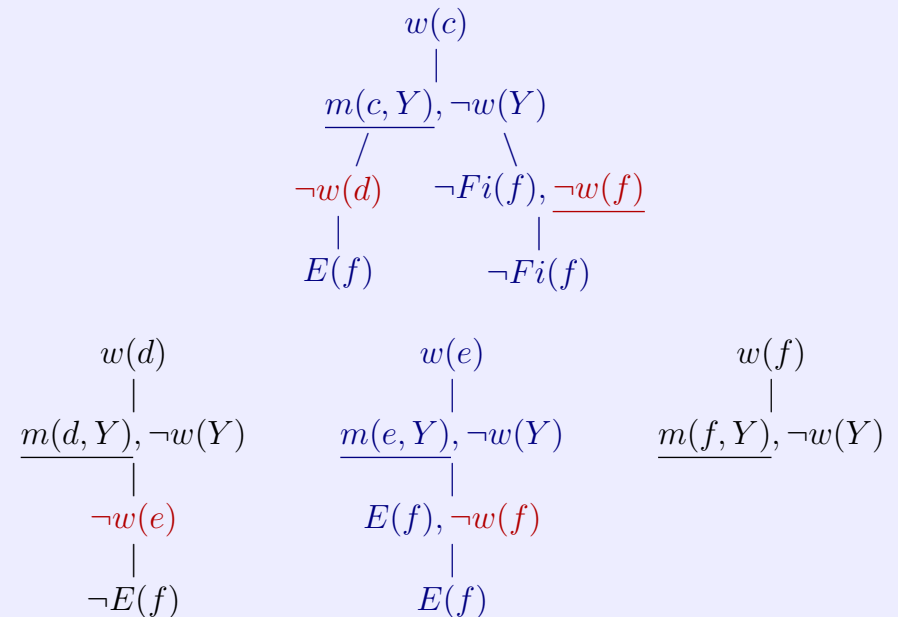
Upper layer – a tree of trees; **two** kinds of trees needed.

t-trees – conditions for the root to be **t**

tu-trees – conditions for the root to be **t** or **u**

Nontrivial handling of constraints+(in)equalities
based on constructive negation for LP.

Ex.: t-trees and tu-trees for *WINH*, nodes simplified



HD-rules, operational semantics

Prototype

XSB Prolog + Pellet DL reasoner

Constraints: DL concepts, \wedge , \vee , \exists ,

Compilation to Prolog with tabulation.

DL reasoner – a black box.

Design decisions (... simplicity)

Sub-trees only for ground literals

Building whole t- tu-trees

Querying DL *once*, when the trees are ready.

HD-rules, operational semantics (summary)

Operational semantics for WFST

Easy implementation by *re-using* reasoners

for LP and external theory

DL-reasoner - a black box

For the rule part

Not only Datalog, data structures possible

Prolog style programming possible (Logic + Control)

Prolog built-ins available

Rather few queries to the DL solver

Provides negation with WFS to Constraint Logic Programming

Rules \longrightarrow external theory

Hybrid programs ($\{\text{WFS|SMS}\}\{\text{T|L}\}$) – asymmetric.

External predicates allowed in P , rule predicates not allowed in T .

Rule predicates defined by P, T , external by T .

The communication is one way,

$$P \leftarrow\!\!\!\leftarrow T$$

Really?

Not in SMST.

Rules \longrightarrow external theory, SMST

A stable model of P/M_0 may not exist

Ex.: $T = \emptyset$, $\mathcal{P}_T = \{q\}$, $P = \{f \leftarrow \neg f, \neg q; p \leftarrow q\}$.

Two ground programs to consider:

$$P/M_0 = \{f \leftarrow \neg f\} \text{ and } P/M_1 = \{p\}.$$

No stable model of P/M_0 ; one stable model $\{p\}$ of P/M_1 .

(P, T) entails p (under SMST).

In a sense, P made true the external predicate q .

Rules → external theory (3)

1. For some models M_0 of T , a stable model of P/M_0 may not exist. P influences T by excluding some its models.
2. Many SMST approaches allow external predicates in rule heads.
3. Under loose c., dl-atoms can query a modified T .

Can 2. be expressed by 1.?

Conjecture: YES

1., 2. seem applicable only to SMST, 3. only to loose coupling.

Summary

- ▶ Semantics for negation in LP (3CS, WFS, SMS)
- ▶ Integration rules (LP) – FOL (e.g. DL)
 - mainly heterogeneous
- ▶ Loose / tight coupling, WFS / SMS as LP semantics
- ▶ Overview of existing approaches
 - 3 kinds of operational semantics
- ▶ Minor conclusions:
 - Both kinds of coupling needed
 - External predicates in rule heads unnecessary in SMST (conjecture)