# Description Logics, Rules and Multi-Context Systems

Luís Cruz-Filipe
(joint work with R. Henriques and I. Nunes)

Escola Superior Náutica Infante D. Henrique / CMAF / LabMAg

LPAR-19
December 18th, 2013

# Outline

# Outline

# Outline

# Outline

1. Combinations of reasoning systems
2. (M)dl-programs
3. Multi-context systems
4. Correspondence

# Outline

# Outline

# Motivation

- Proliferation of software for reasoning
- Technology reuse
- Capitalize on domain-specific technology

# Motivation

- Proliferation of software for reasoning
- Technology reuse
- Capitalize on domain-specific technology

Particular problem: combining description logics and rules

# Two main approaches

## Homogeneous systems

New language including all desired features

- "Easy" to understand
- Require specific technology

# Two main approaches

## Homogeneous systems

New language including all desired features

- "Easy" to understand
- Require specific technology

## Heterogeneous systems

Several components of *different* kinds.

- Harder to understand
- Rely on communication/interface
- Highly modular

# Combining description logics with rules

- (M)dl-programs
- HEX-programs
- Multi-context systems
- MKNF

# Combining description logics with rules

- (M)dl-programs
- HEX-programs
- Multi-context systems
- MKNF

Correspondence results:

- (M)dl-programs $\subsetneq$ HEX-programs (trivial)
- HEX-programs and MCSs incomparable
- MKNF $\subseteq$ MCS
- (M)dl-programs $\subsetneq$ MCSs

# Outline

# Syntax & semantics

# Syntax & semantics

## Syntax

- Logic program + DL knowledge bases
- Special *dl-atoms* for communication

$$\underbrace{DL_i}_{\text{KB identifier}}\ [\underbrace{S_1 \bullet_1 p_1, \ldots, S_n \bullet_n p_n}_{\text{input context}};\ \underbrace{Q}_{\text{query}}](\vec{X})$$

with $\bullet_k \in \{\uplus, \cup\!\!\!\!-\}$

# Syntax & semantics

## Syntax

- Logic program + DL knowledge bases
- Special *dl-atoms* for communication

$$\underbrace{DL_i}_{\text{KB identifier}}\ [\underbrace{S_1 \bullet_1 p_1, \ldots, S_n \bullet_n p_n}_{\text{input context}};\underbrace{Q}_{\text{query}}](\vec{X})$$

with $\bullet_k \in \{\uplus, \cupdot\}$

## Semantics

Herbrand models (with constants from the knowledge bases)

- Minimal models
- Answer-sets
- Well-founded semantics

# Example

## Example

$\Sigma_1$ is a travel ontology, $\Sigma_2$ is a wine ontology

$$
\begin{aligned}
\mathsf{wineDest}(X) &\leftarrow DL_2[;\mathsf{Region}](X) \\
\mathsf{wineDest}(\mathsf{Stellenbosch}) &\leftarrow \\
\mathsf{wineDest}(\mathsf{Sydney}) &\leftarrow
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{overnight}(X) &\leftarrow DL_1[;\mathsf{hasAccommodation}](X, Y) \\
\mathsf{oneDayTrip}(X) &\leftarrow DL_1[\mathsf{Destination} \uplus \mathsf{wineDest};\mathsf{Destination}](X), \\
&\quad\ \mathsf{not}\ \mathsf{overnight}(X)
\end{aligned}
$$

# Outline

# Syntax (I)

## Logic

A *logic* is the language underlying a context, specifying its syntax and “semantics”:
$L = \langle KB, BS, ACC \rangle$

- $KB$ is the set of *knowledge bases*
- $BS$ is the set of *belief sets*
- $ACC : KB \rightarrow 2^{BS}$ assigns acceptable belief sets to knowledge bases

# Syntax (I)

## Logic

A *logic* is the language underlying a context, specifying its syntax and "semantics":

$L = \langle KB, BS, ACC \rangle$

- $KB$ is the set of *knowledge bases*
- $BS$ is the set of *belief sets*
- $ACC : KB \to 2^{BS}$ assigns acceptable belief sets to knowledge bases

Examples: Reiter's default logic; FOL; logic programs; description logics; . . .

# Syntax (II)

## Context

A *context* is a specific knowledge base in a given logic:
$C = \langle L, kb, br \rangle$

- $L$ is a logic
- $kb$ is a particular knowledge base
- $br$ is a set of *bridge rules* connecting $C$ to other contexts

# Syntax (II)

## Context

A *context* is a specific knowledge base in a given logic:
$C = \langle L, kb, br \rangle$

- $L$ is a logic
- $kb$ is a particular knowledge base
- $br$ is a set of *bridge rules* connecting $C$ to other contexts

A bridge rule:

$$\mathrm{p} \leftarrow (i_1 : q_i), \ldots, (i_n : q_n), \mathrm{not}\ (i_{n+1}, q_{n+1}), \ldots, \mathrm{not}\ (i_m, q_m)$$

where $i_k$ are context identifiers (numbers) and $q_k$ are elements of belief sets in the corresponding context

# Syntax (III)

## Multi-context system

A *Multi-context system* (MCS) is a set of contexts whose bridge rules connect to contexts in the same set:
$M = \langle C_1, \ldots, C_n \rangle$
and all context identifiers in bridge rules are numbers ranging from 1 to $n$.

# Syntax (III)

**Multi-context system**

A *Multi-context system* (MCS) is a set of contexts whose bridge rules connect to contexts in the same set:
$M = \langle C_1, \ldots, C_n \rangle$
and all context identifiers in bridge rules are numbers ranging from 1 to $n$.

Technically: *non-monotonic heterogenous multi-context systems*

# Semantics

A *belief state* is a set of belief sets, one for each context.

# Semantics

A *belief state* is a set of belief sets, one for each context.

An *equilibrium* is a belief state such that that each belief set is acceptable w.r.t. the knowledge base of that context extended with the input from that context's bridge rules, given the belief state.

# Semantics

A *belief state* is a set of belief sets, one for each context.

An *equilibrium* is a belief state such that that each belief set is acceptable w.r.t. the knowledge base of that context extended with the input from that context's bridge rules, given the belief state.

Same idea as that of models of logic programming.

- Minimal equilibria
- Grounded equilibria
- Well-founded equilibria

# Outline

# Motivation

MCSs were proposed as a generalization of dl-programs, but there are some differences.

- No logic program (where do the rules go?)
- Many local "views" of the knowledge base vs only global changes

# Motivation

MCSs were proposed as a generalization of dl-programs, but there are some differences.

- No logic program (where do the rules go?)
- Many local "views" of the knowledge base vs only global changes

**Example**

$$
\begin{aligned}
\text{wineDest}(X) &\leftarrow DL_2[;\text{Region}](X) \\
\text{overnight}(X) &\leftarrow DL_1[;\text{hasAccommodation}](X, Y) \\
\text{oneDayTrip}(X) &\leftarrow DL_1[\text{Destination} \uplus \text{wineDest};\text{Destination}](X), \\
&\quad\ \text{not overnight}(X)
\end{aligned}
$$

# Idea

- Define a context $C_0$ containing the purely logical part of the logic program.

# Idea

- Define a context $C_0$ containing the purely logical part of the logic program.
- Define contexts $C_i^j$ for each knowledge base $\Sigma_i$ and each distinct input context in dl-atoms querying $\Sigma_i$.

# Idea

- Define a context $C_0$ containing the purely logical part of the logic program.
- Define contexts $C_i^j$ for each knowledge base $\Sigma_i$ and each distinct input context in dl-atoms querying $\Sigma_i$.
- The logic underlying each $C_i^j$ defines $ACC(kb)$ as the (singleton set containing the) set of logical consequences of $kb$.

# Idea

- Define a context $C_0$ containing the purely logical part of the logic program.
- Define contexts $C_i^j$ for each knowledge base $\Sigma_i$ and each distinct input context in dl-atoms querying $\Sigma_i$.
- The logic underlying each $C_i^j$ defines $ACC(kb)$ as the (singleton set containing the) set of logical consequences of $kb$.
- Rules with dl-atoms become bridge rules.

# Example (cont'd)

## Example

$\Sigma_1$ is a travel ontology, $\Sigma_2$ is a wine ontology

$$\begin{aligned}
\mathrm{wineDest}(X) &\leftarrow DL_2[;\mathrm{Region}](X) \\
\mathrm{wineDest}(\mathrm{Stellenbosch}) &\leftarrow \\
\mathrm{wineDest}(\mathrm{Sydney}) &\leftarrow
\end{aligned}$$

$$\begin{aligned}
\mathrm{overnight}(X) &\leftarrow DL_1[;\mathrm{hasAccommodation}](X, Y) \\
\mathrm{oneDayTrip}(X) &\leftarrow DL_1[\mathrm{Destination} \uplus \mathrm{wineDest}; \mathrm{Destination}](X), \\
&\qquad \mathrm{not}\ \mathrm{overnight}(X)
\end{aligned}$$

# Our example

- $C_1^1$: travel ontology with no bridge rules
- $C_1^2$: travel ontology with bridge rule

$$\text{Destination}(X) \leftarrow (0 : \text{wineDest}(X))$$

- $C_2$: wine ontology with no bridge rules
- $C_0$: the logic program

$$\begin{aligned} \text{wineDest}(\text{Stellenbosch}) &\leftarrow \\ \text{wineDest}(\text{Sydney}) &\leftarrow \end{aligned}$$

with bridge rules

$$\begin{aligned} \text{wineDest}(X) &\leftarrow (2 : \text{Region}(X)) \\ \text{overnight}(X) &\leftarrow (1^1 : \text{hasAccommodation}(X, Y)) \\ \text{oneDayTrip}(X) &\leftarrow (1^2 : \text{Destination}(X)), (0 : \text{not overnight}(X)) \end{aligned}$$

# At the semantic level

Belief state $S$ induced by interpretation $I$ for the logic program

**Theorem**

- *$S$ is equilibrium (for the MCS) iff $I$ is a model (of the Mdl-program)*
- *$S$ is minimal iff $I$ is minimal*
- *$S$ is grounded iff $I$ is answer-set*
- *$S$ is well-founded iff $I$ is well-founded*

# Outline

# Mdl-programs vs Multi-context systems

- Strictly included
- Equivalence of semantics
- Portability of results

# Thank you.