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

LabMAg Seminar
December 6th, 2013

Outline

Outline

Outline

Outline

Outline

Outline

Motivation

Motivation

- Proliferation of software
- Expert systems
- Technology reuse
- Capitalize on domain-specific technology

Motivation

- Proliferation of software
- Expert systems
- Technology reuse
- Capitalize on domain-specific technology

Particular problem: combining description logics and rules

Homogenous approach

Homogenous approach

Homogeneous systems

Several components of the *same* kind.

- (Large) Java programs
- MKNF knowledge bases

Homogenous approach

Homogeneous systems

Several components of the *same* kind.

- (Large) Java programs
- MKNF knowledge bases

- “Easy” to understand
- Require specific technology
- Hard to reuse existing tools

Heterogenous approach

Heterogenous approach

Heterogeneous systems

Several components of *different* kinds.

- Service-oriented computing
- dl-programs and their variants

Heterogenous approach

Heterogeneous systems

Several components of *different* kinds.

- Service-oriented computing
- dl-programs and their variants

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

Combining description logics with rules (I)

Combining description logics with rules (I)

dl-programs (Eiter et al.)

One DL knowledge base + one logic program
Communication via special atoms
Generalized to several knowledge bases

Combining description logics with rules (I)

dl-programs (Eiter et al.)

HEX-programs (Eiter et al.)

Several knowledge bases (no restrictions)
One higher-order logic program
Communication atoms can access arbitrary resources

Combining description logics with rules (I)

dl-programs (Eiter et al.)

HEX-programs (Eiter et al.)

Multi-context systems (Brewka et al.)

Arbitrary knowledge bases
No logic program
Symmetric communication via bridge rules

Combining description logics with rules (I)

dl-programs (Eiter et al.)

HEX-programs (Eiter et al.)

Multi-context systems (Brewka et al.)

MKNF (Motik et al.)

Homogeneous approach (so only one component)
Modal quantifiers over ontologies
"Mysterious" semantics

Combining description logics with rules (I)

dl-programs (Eiter et al.)

HEX-programs (Eiter et al.)

Multi-context systems (Brewka et al.)

MKNF (Motik et al.)

How do these compare?

Combining description logics with rules (II)

Combining description logics with rules (II)

Correspondence results:

- (M)dl-programs $\subsetneq$ HEX-programs (trivial)
- HEX-programs and MCSs incomparable
- MKNF $\subseteq$ MCS (Homola et al.)
- (M)dl-programs $\subsetneq$ MCSs (hinted at in Brewka et al.)

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

Semantics

Herbrand models (with constants from the knowledge bases)

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

(M)dl-programs in practice

- Simple!
- Modular
- Allow reuse of technology (ontologies)
- Several publications with domain-specific examples
- Interpretable in HEX-programs or MCSs
- Reasoning tools
- Design patterns

Example

Example

Σ_1 is a travel ontology, Σ_2 is a wine ontology

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

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

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*
- $ACC : KB \rightarrow 2^{BS}$ 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.

Equilibrium

It's a kind of fixpoint, dude!

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.

Equilibrium

It's a kind of fixpoint, dude!

Same idea as that of models of logic programming.

- Minimal equilibria
- Grounded equilibria
- Well-founded equilibria

MCSs in practice

- Highly expressive
- Modular
- Allow reuse of technology (not only ontologies)
- Even more publications with domain-specific examples

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

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

Idea

- Generate a logic program context
- Split the original program between that context and bridge rules
- Generate a context for each view of a knowledge base
- Add bridge rules to these contexts corresponding the the desired view

Big but. . .

Problem

How does an ontology generate a context?

Big but...

Problem

How does an ontology generate a context?

Example

$\mathcal{O} = \{C(a)\}$, $U = \{a, b\}$

How do we close the world?

Big but. . .

Problem

How does an ontology generate a context?

Example

$\mathcal{O} = \{C(a)\}$, $U = \{a, b\}$

How do we close the world?

Bridge rule: "$\neg C(X) \leftarrow \text{not } (1 : C(X))$"

Big but...

Problem

How does an ontology generate a context?

Example

$\mathcal{O} = \{C(a)\}$, $U = \{a, b\}$

How do we close the world?

Bridge rule: "$\neg C(X) \leftarrow \text{not } (1 : C(X))$"

Problem

Does not work!

The translation

- Knowledge base Σ_i induces a context C_i^j for each input context in dl-atoms querying Σ_i
- The logic C_i^j defines $ACC(kb)$ as the (singleton set containing the) set of logical consequences of kb

The translation

- Knowledge base Σ_i induces a context C_i^j for each input context in dl-atoms querying Σ_i
- The logic C_i^j defines $ACC(kb)$ as the (singleton set containing the) set of logical consequences of kb
  May not be a model of kb!

The translation

- Knowledge base Σ_i induces a context C_i^j for each input context in dl-atoms querying Σ_i
- The logic C_i^j defines $ACC(kb)$ as the (singleton set containing the) set of logical consequences of kb
- The logic program induces a context C_0 containing its purely logical part
- Rules with dl-atoms become bridge rules

Our example

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

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

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

$$\mathsf{wineDest}(\mathsf{Tasmania}) \leftarrow$$
$$\mathsf{wineDest}(\mathsf{Sydney}) \leftarrow$$

with bridge rules

$$\mathsf{wineDest}(X) \leftarrow (2 : \mathsf{Region}(X))$$
$$\mathsf{overnight}(X) \leftarrow (1^1 : \mathsf{hasAccommodation}(X, Y))$$
$$\mathsf{oneDayTrip}(X) \leftarrow (1^2 : \mathsf{Destination}(X)), (0 : \mathsf{not\ overnight}(X))$$

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.