Inconsistency Management in Reactive Mulit-Context Systems[1]

Gerhard Brewka[2] Stefan Ellmauthaler[2] Ricardo Gonçalves[3]
Matthias Knorr[3] João Leite[3] Jörg Pührer[2]

[2] Computer Science Institute, Leipzig University, Germany
[3] NOVA LINCS, Universidade NOVA de Lisboa, Portugal

Stream Reasoning Workshop
Berlin
December 8^{th}, 2016

[1] Research has been supported by DFG and FWF (projects BR 1817/7-1 and FOR 1513)

Recent Development

Reactive MCS

Evolving MCS

Recent Development

Reactive MCS

- presented at ECAI 2014
- developed in Leipzig
- equilibrium of one “step” is base kb in next “step”

Evolving MCS

- presented at ECAI 2014
- developed in Lisbon
- utilise a “next” operator

Recent Development

Reactive MCS

- presented at ECAI 2014
- developed in Leipzig
- equilibrium of one “step” is base kb in next “step”

Evolving MCS

- presented at ECAI 2014
- developed in Lisbon
- utilise a “next” operator

“new” reactive Multi-Context Systems

- combined ideas of rMCS and eMCS
- “bilateral” ongoing research on that topic

Outline

1. Motivation
2. Reactive Multi-Context Systems
3. Inconsistency Management

Motivation

- **integration** of heterogenous KR-formalisms
- **awareness** of continous flow of knowledge
  - information is constantly produced and shared
  - shift from static one-shot computation to stream processing
- distinguish between **persistent** and **non-persistent** effects of input streams
- **represent** state transitions over time

Motivation

- **integration** of heterogenous KR-formalisms
- **awareness** of continous flow of knowledge
  - information is constantly produced and shared
  - shift from static one-shot computation to stream processing
- distinguish between **persistent** and **non-persistent** effects of input streams
- **represent** state transitions over time

Inconsistency Management

Motivation

- **integration** of heterogenous KR-formalisms
- **awareness** of continous flow of knowledge
  - information is constantly produced and shared
  - shift from static one-shot computation to stream processing
- distinguish between **persistent** and **non-persistent** effects of input streams
- **represent** state transitions over time

Inconsistency Management

- How to ensure consistency?

Motivation

- **integration** of heterogenous KR-formalisms
- **awareness** of continous flow of knowledge
  - information is constantly produced and shared
  - shift from static one-shot computation to stream processing
- distinguish between **persistent** and **non-persistent** effects of input streams
- **represent** state transitions over time

Inconsistency Management

- How to ensure consistency?
- How to repair inconsistent cases?

Motivation

- **integration** of heterogenous KR-formalisms
- **awareness** of continous flow of knowledge
  - information is constantly produced and shared
  - shift from static one-shot computation to stream processing
- distinguish between **persistent** and **non-persistent** effects of input streams
- **represent** state transitions over time

Inconsistency Management

- How to ensure consistency?
- How to repair inconsistent cases?
- How to work with inconsistent cases?

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic
  Logic: defines the possible knowledge bases and their semantics
  Example: Logic programs with answer-set semantics

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic
  Logic: defines the possible knowledge bases and their semantics
  Example: Logic programs with answer-set semantics
- **Operations**: each context has a set of operations applicable to the knowledge bases of the context

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic
  Logic: defines the possible knowledge bases and their semantics
  Example: Logic programs with answer-set semantics
- **Operations**: each context has a set of operations applicable to the knowledge bases of the context
  Examples: addition, revision, updating, forgetting

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic
  Logic: defines the possible knowledge bases and their semantics
  Example: Logic programs with answer-set semantics

- **Operations**: each context has a set of operations applicable to the knowledge bases of the context
  Examples: addition, revision, updating, forgetting

- **Bridge rules**: declarative non-monotonic rules that model the flow of information between contexts

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic
  Logic: defines the possible knowledge bases and their semantics
  Example: Logic programs with answer-set semantics

- **Operations**: each context has a set of operations applicable to the knowledge bases of the context
  Examples: addition, revision, updating, forgetting

- **Bridge rules**: declarative non-monotonic rules that model the flow of information between contexts
  Apply the operation in the head of the rule, provided the queries (to other contexts) in the body are successful

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic
  Logic: defines the possible knowledge bases and their semantics
  Example: Logic programs with answer-set semantics

- **Operations**: each context has a set of operations applicable to the knowledge bases of the context
  Examples: addition, revision, updating, forgetting

- **Bridge rules**: declarative non-monotonic rules that model the flow of information between contexts
  Apply the operation in the head of the rule, provided the queries (to other contexts) in the body are successful

- **Semantics**: Notion of Equilibrium

Multi-Context Systems

- **Contexts**: knowledge base represented in some logic
  Logic: defines the possible knowledge bases and their semantics
  Example: Logic programs with answer-set semantics
- **Operations**: each context has a set of operations applicable to the knowledge bases of the context
  Examples: addition, revision, updating, forgetting
- **Bridge rules**: declarative non-monotonic rules that model the flow of information between contexts
  Apply the operation in the head of the rule, provided the queries (to other contexts) in the body are successful
- **Semantics**: Notion of Equilibrium
  Takes into account the semantics of each context and the operational formulas in the head of the applicable bridge rules

Reactive Multi-Context Systems

Definition (Reactive Multi-Context System)

A *reactive Multi-Context System (rMCS)* is a tuple $M = \langle \mathsf{C}, \mathsf{IL}, \mathsf{BR} \rangle$, where

- $\mathsf{C} = \langle C_1, \ldots, C_n \rangle$ is a tuple of contexts $C_i = \langle L_i, OP_i, \mathbf{mng}_i \rangle$
  - $L_i = \langle KB_i, BS_i, \mathbf{acc}_i \rangle$ is a logic,
  - OP_i is a *set of operations*,
  - $\mathbf{mng}_i : 2^{OP} \times KB \rightarrow KB$ is a *management function*.
- $\mathsf{IL} = \langle IL_1, \ldots, IL_k \rangle$ is a tuple of input languages;
- $\mathsf{BR} = \langle BR_1, \ldots, BR_n \rangle$ is a tuple such that each BR_i is a set of bridge rules for C_i over C and IL of the form

$$\mathbf{op} \leftarrow a_1, \ldots, a_j, \mathbf{not}\ a_{j+1}, \ldots, \mathbf{not}\ a_m$$

  - $\mathbf{op} = op$ or $\mathbf{op} = \mathbf{next}(op)$ for $op \in OP_i$.
  - and every *atom* a_ℓ, is a *context atom* $c{:}b$ or an *input atom* $s{::}b$.

Reactive Multi-Context Systems

Definition (Reactive Multi-Context System)

A *reactive Multi-Context System (rMCS)* is a tuple $M = \langle \mathsf{C}, \mathsf{IL}, \mathsf{BR} \rangle$, where

- $\mathsf{C} = \langle C_1, \ldots, C_n \rangle$ is a tuple of contexts $C_i = \langle L_i, OP_i, \mathbf{mng}_i \rangle$
  - $L_i = \langle KB_i, BS_i, \mathbf{acc}_i \rangle$ is a logic,
  - OP_i is a *set of operations*,
  - $\mathbf{mng}_i : 2^{OP} \times KB \rightarrow KB$ is a *management function*.
- $\mathsf{IL} = \langle IL_1, \ldots, IL_k \rangle$ is a tuple of input languages;
- $\mathsf{BR} = \langle BR_1, \ldots, BR_n \rangle$ is a tuple such that each BR_i is a set of bridge rules for C_i over C and IL of the form

$$\mathbf{op} \leftarrow a_1, \ldots, a_j, \mathbf{not}\ a_{j+1}, \ldots, \mathbf{not}\ a_m$$

  - $\mathbf{op} = op$ or $\mathbf{op} = \mathbf{next}(op)$ for $op \in OP_i$.
  - and every *atom* a_ℓ, is a *context atom* $c{:}b$ or an *input atom* $s{::}b$.

Reactive Multi-Context Systems

Definition (Reactive Multi-Context System)

A *reactive Multi-Context System (rMCS)* is a tuple $M = \langle \mathsf{C}, \mathsf{IL}, \mathsf{BR} \rangle$, where

- $\mathsf{C} = \langle C_1, \ldots, C_n \rangle$ is a tuple of contexts $C_i = \langle L_i, OP_i, \mathbf{mng}_i \rangle$
  - $L_i = \langle KB_i, BS_i, \mathbf{acc}_i \rangle$ is a logic,
  - OP_i is a *set of operations*,
  - $\mathbf{mng}_i : 2^{OP} \times KB \rightarrow KB$ is a *management function*.
- $\mathsf{IL} = \langle IL_1, \ldots, IL_k \rangle$ is a tuple of input languages;
- $\mathsf{BR} = \langle BR_1, \ldots, BR_n \rangle$ is a tuple such that each BR_i is a set of bridge rules for C_i over C and IL of the form

$$\mathbf{op} \leftarrow a_1, \ldots, a_j, \mathbf{not}\ a_{j+1}, \ldots, \mathbf{not}\ a_m$$

  - $\mathbf{op} = op$ or $\mathbf{op} = \mathbf{next}(op)$ for $op \in OP_i$.
  - and every *atom* a_ℓ, is a *context atom* $c{:}b$ or an *input atom* $s{::}b$.

Reactive Multi-Context Systems

Definition (Reactive Multi-Context System)

A *reactive Multi-Context System (rMCS)* is a tuple $M = \langle \mathsf{C}, \mathsf{IL}, \mathsf{BR} \rangle$, where

- $\mathsf{C} = \langle C_1, \ldots, C_n \rangle$ is a tuple of contexts $C_i = \langle L_i, OP_i, \mathbf{mng}_i \rangle$
  - $L_i = \langle KB_i, BS_i, \mathbf{acc}_i \rangle$ is a logic,
  - OP_i is a *set of operations*,
  - $\mathbf{mng}_i : 2^{OP} \times KB \rightarrow KB$ is a *management function*.
- $\mathsf{IL} = \langle IL_1, \ldots, IL_k \rangle$ is a tuple of input languages;
- $\mathsf{BR} = \langle BR_1, \ldots, BR_n \rangle$ is a tuple such that each BR_i is a set of bridge rules for C_i over C and IL of the form

$$\mathbf{op} \leftarrow a_1, \ldots, a_j, \mathbf{not}\ a_{j+1}, \ldots, \mathbf{not}\ a_m$$

  - $\mathbf{op} = op$ or $\mathbf{op} = \mathbf{next}(op)$ for $op \in OP_i$.
  - and every *atom* a_ℓ, is a *context atom* $c{:}b$ or an *input atom* $s{::}b$.

Semantics

Given

a rMCS $M = \langle\langle C_1, \ldots, C_n\rangle, \langle IL_1, \ldots, IL_k\rangle, \mathsf{BR}\rangle$, with

- an initial configuration of knowledge bases $\mathsf{KB} = \langle kb_i, \ldots, kb_n\rangle$, such that $kb_i \in KB_i$, for each $i \in \{1, \ldots, n\}$, and
- an input stream (until τ) $\mathcal{I} : [1..\tau] \rightarrow \mathsf{In}_M$

Semantics

Given

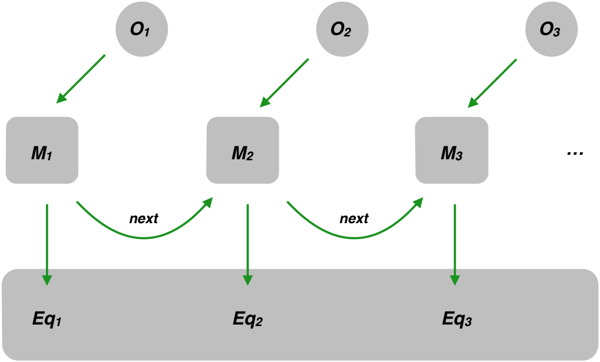
a rMCS $M = \langle\langle C_1, \dots, C_n\rangle, \langle IL_1, \dots, IL_k\rangle, \mathsf{BR}\rangle$, with

- an initial configuration of knowledge bases $\mathsf{KB} = \langle kb_i, \dots, kb_n\rangle$, such that $kb_i \in KB_i$, for each $i \in \{1, \dots, n\}$, and
- an input stream (until τ) $\mathcal{I} : [1..\tau] \to \mathsf{In}_M$

Equilibria Stream

- Static equilibrium at each time instant, with respect to management operations (op) in applicable bridge rules
- Knowledge bases are updated from one time instant to the next one by applying management operations ($\mathbf{next}(op)$) in applicable bridge rules

Semantics - Equilibria Stream



Semantics - Equilibria Stream

Definition (Equilibrium)

Let $M = \langle\langle C_1, \ldots, C_n\rangle, \mathsf{IL}, \mathsf{BR}\rangle$ be an rMCS, $\mathsf{KB} = \langle kb_1, \ldots, kb_n\rangle$ a configuration of knowledge bases for M, and I an input for M. Then, a belief state $\mathsf{B} = \langle B_1, \ldots, B_n\rangle$ for M is an equilibrium of M given KB and I if, for each $i \in \{1, \ldots, n\}$, we have that

$$B_i \in \mathbf{acc}_i(kb'), \text{ where } kb' = \mathbf{mng}_i(\mathbf{app}_i^{now}(\mathsf{I}, \mathsf{B}), kb_i).$$

Semantics - Equilibria Stream

Definition (Equilibria Stream)

Let $M = \langle\langle C_1, \ldots, C_n\rangle, \mathsf{IL}, \mathsf{BR}\rangle$ be an rMCS, $\mathsf{KB} = \langle kb_1, \ldots, kb_n\rangle$ a configuration of knowledge bases for M, and $\mathcal{I} : [1..\tau] \to \mathsf{In}_M$ an input stream for M until τ. Then, an equilibria stream of M given KB and $\mathcal{I}$ is a function $\mathcal{B} : [1..\tau] \to \mathsf{Bel}_M$ such that

- $\mathcal{B}^t$ is an equilibrium of M given $\mathcal{KB}^t$ and $\mathcal{I}^t$, where $\mathcal{KB}^t$ is
  - $\mathcal{KB}^1 = \mathsf{KB}$
  - $\mathcal{KB}^{t+1} = \mathbf{upd}_M(\mathcal{KB}^t, \mathcal{I}^t, \mathcal{B}^t)$, where $\mathbf{upd}_M(\mathsf{KB}, \mathsf{I}, \mathsf{B}) = \langle kb'_1, \ldots, kb'_n\rangle$, such that $kb'_i = \mathbf{mng}_i(\mathbf{app}_i^{next}(\mathsf{I}, \mathsf{B}), kb_i)$

Consistent rMCS

Definition

Let M be an rMCS, KB a configuration of knowledge bases for M, and $\mathcal{I}$ an input stream for M. Then:

- M is consistent with respect to KB and $\mathcal{I}$ if there exists an equilibria stream of M given KB and $\mathcal{I}$.
- M is strongly consistent with respect to KB if, for every input stream $\mathcal{I}$ for M, M is consistent with respect to KB and $\mathcal{I}$.

Strong Consistency of rMCS

Question

Can we ensure strong consistency of a rMCS?

Strong Consistency of rMCS

Question

Can we ensure strong consistency of a rMCS?

Definition

A context C_i is totally coherent if $\mathbf{acc}_i(kb) \neq \emptyset$, for every $kb \in KB_i$.

Strong Consistency of rMCS

Question

Can we ensure strong consistency of a rMCS?

Definition

A context C_i is totally coherent if $\mathbf{acc}_i(kb) \neq \emptyset$, for every $kb \in KB_i$.

Definition

An rMCS M is acyclic if the transitive closure of the dependency relation between contexts induced by the bridge rules is irreflexive.

Strong Consistency of rMCS

Question

Can we ensure strong consistency of a rMCS?

Definition

A context C_i is totally coherent if $\mathbf{acc}_i(kb) \neq \emptyset$, for every $kb \in KB_i$.

Definition

An rMCS M is acyclic if the transitive closure of the dependency relation between contexts induced by the bridge rules is irreflexive.

Proposition

Let $M = \langle\langle C_1, \ldots, C_n\rangle, \mathsf{IL}, \mathsf{BR}\rangle$ *be an acyclic rMCS such that every* C_i*,* $1 \leq i \leq n$*, is totally coherent, and* KB *a configuration of knowledge bases for* M*. Then,* M *is strongly consistent with respect to* KB*.*

Recovering Equilibria Streams

Question

What if there are no equilibria streams?

Recovering Equilibria Streams

Question

What if there are no equilibria streams?

Definition (Repair)

Let $M = \langle \mathsf{C}, \mathsf{IL}, \mathsf{BR} \rangle$ be an rMCS, KB a configuration of knowledge bases for M, and $\mathcal{I}$ an input stream for M until τ. Let

- br_M denote the set of all bridge rules of M
- $M[R]$ denote the rMCS obtained from M by restricting the bridge rules to those not in R

A repair for M given KB and $\mathcal{I}$ is a function $\mathcal{R} : [1..\tau] \to 2^{br_M}$ such that there exists a function $\mathcal{B} : [1..\tau] \to \mathsf{Bel}_M$ such that

- $\mathcal{B}^t$ is an equilibrium of $M[\mathcal{R}^t]$ given $\mathcal{KB}^t$ and $\mathcal{I}^t$, with $\mathcal{KB}^t$ inductively defined as
  - $\mathcal{KB}^1 = \mathsf{KB}$
  - $\mathcal{KB}^{t+1} = \mathbf{upd}_{M[\mathcal{R}^t]}(\mathcal{KB}^t, \mathcal{I}^t, \mathcal{B}^t)$,

On repairs of rMCS composed of totally coherent contexts

Proposition

Let $M = \langle\langle C_1, \ldots, C_n\rangle, \mathsf{IL}, \mathsf{BR}\rangle$ *be an rMCS such that each* C_i *is totally coherent,* KB *a configuration of knowledge bases for* M*, and* $\mathcal{I}$ *an input stream for* M *until* τ*. Then, there exists* $\mathcal{R} : [1..\tau] \to 2^{br_M}$ *and* $\mathcal{B} : [1..\tau] \to \mathsf{Bel}_M$ *such that* $\mathcal{B}$ *is a repaired equilibria stream given* KB*,* $\mathcal{I}$ *and* $\mathcal{R}$*.*

Types of Repairs

Question

Are all the repairs equally good?

Types of Repairs

Question

Are all the repairs equally good?

Definition

For two repairs $\mathcal{R}_a$ and $\mathcal{R}_b$, we say that $\mathcal{R}_a \leq \mathcal{R}_b$ if $\mathcal{R}_a^i \subseteq \mathcal{R}_b^i$ for every $i \leq \tau$, and that $\mathcal{R}_a < \mathcal{R}_b$ if $\mathcal{R}_a \leq \mathcal{R}_b$ and $\mathcal{R}_a^i \subset \mathcal{R}_b^i$ for some $i \leq \tau$.

Types of Repairs

Definition (Types of Repairs)

Let $\mathcal{R}$ be a repair for a rMCS M given KB and $\mathcal{I}$. We say that $\mathcal{R}$ is a:

- Minimal Repair if there is no repair $\mathcal{R}_a$ for M given KB and $\mathcal{I}$ such that $\mathcal{R}_a < \mathcal{R}$.
- Global Repair if $\mathcal{R}^i = \mathcal{R}^j$ for every $i, j \leq \tau$.
- Minimal Global Repair if $\mathcal{R}$ is global and there is no global repair $\mathcal{R}_a$ for M given KB and $\mathcal{I}$ such that $\mathcal{R}_a < \mathcal{R}$.
- Incremental Repair if $\mathcal{R}^i \subseteq \mathcal{R}^j$ for every $i \leq j \leq \tau$.
- Minimally Incremental Repair if $\mathcal{R}$ is incremental and there is no incremental repair $\mathcal{R}_a$ and $j \leq \tau$ such that $\mathcal{R}_a^i \subset \mathcal{R}^i$ for every $i \leq j$.

Partial Equilibria Stream

Question

What if there are no repairs?

Partial Equilibria Stream

Question

What if there are no repairs? ... Or we don't want to compute them?

Partial Equilibria Stream

Question

What if there are no repairs? ... Or we don't want to compute them?

Definition (Partial Equilibria Stream)

Let $M = \langle \mathsf{C}, \mathsf{IL}, \mathsf{BR} \rangle$ be an rMCS, KB a configuration of knowledge bases for M, and $\mathcal{I}$ an input stream for M until τ. A partial equilibria stream of M given KB and $\mathcal{I}$ is a partial function $\mathcal{B} : [1..\tau] \nrightarrow \mathsf{Bel}_M$ such that

- $\mathcal{B}^t$ is an equilibrium of M given $\mathcal{KB}^t$ and $\mathcal{I}^t$,
- or $\mathcal{B}^t$ is undefined otherwise.

$\mathcal{KB}^t$ inductively defined as

- $\mathcal{KB}^1 = \mathsf{KB}$
- $\mathcal{KB}^{t+1} = \begin{cases} \mathbf{upd}_M(\mathcal{KB}^t, \mathcal{I}^t, \mathcal{B}^t), & \text{if } \mathcal{B}^t \text{ is not undefined.} \\ \mathcal{KB}^t, & \text{otherwise.} \end{cases}$

On Partial Equilibria Stream

Proposition

Every equilibria stream of M *given* KB *and* $\mathcal{I}$ *is a partial equilibria stream of* M *given* KB *and* $\mathcal{I}$

On Partial Equilibria Stream

Proposition

Every equilibria stream of M *given* KB *and* $\mathcal{I}$ *is a partial equilibria stream of* M *given* KB *and* $\mathcal{I}$

Proposition (Partial equilibria streams always exist)

Let M *be an rMCS,* KB *a configuration of knowledge bases for* M*, and* $\mathcal{I}$ *an input stream for* M *until* τ*. Then, there exists* $\mathcal{B} : [1..\tau] \nrightarrow \mathsf{Bel}_M$ *such that* $\mathcal{B}$ *is a partial equilibria stream given* KB *and* $\mathcal{I}$*.*

Conclusion

- We have introduced the “new” rMCS
- acyclic rMCS whose contexts are totally coherent are strongly consistent
- for each rMCS with only totally coherent contexts there exist repairs
- partial equilibria streams are a way to work with cases without repairs

Thank you for your interest!

References I

[Brewka et al., 2016] Brewka, G., Ellmauthaler, S., Gonçalves, R., Knorr, M., Leite, J., and Pührer, J. (2016).
Towards inconsistency management in reactive multi-context systems.
In *Proceedings of the International Workshop on Defeasible and Ampliative Reasoning (DARe-16) co-located with the 22th European Conference on Artificial Intelligence (ECAI 2016), The Hague, Holland, August 29, 2016.*

[Gonçalves et al., 2014] Gonçalves, R., Knorr, M., and Leite, J. (2014).
Evolving multi-context systems.
In *Proc. ECAI'14*, pages 375–380.