Inconsistency Management in Reactive Mult-Context Systems

Gerhard Brewka\textsuperscript{2}  Stefan Ellmauthaler\textsuperscript{2}  Ricardo Gonçalves\textsuperscript{3}  
Matthias Knorr\textsuperscript{3}  João Leite\textsuperscript{3}  Jörg Pührer\textsuperscript{2}

\textsuperscript{2} Computer Science Institute, Leipzig University, Germany  
\textsuperscript{3} NOVA LINCS, Universidade NOVA de Lisboa, Portugal

Stream Reasoning Workshop  
Berlin  
December 8\textsuperscript{th}, 2016

\textsuperscript{1}\textit{Research has been supported by DFG and FWF (projects BR 1817/7-1 and FOR 1513)}
Recent Development

Reactive MCS

Evolving MCS
## Recent Development

<table>
<thead>
<tr>
<th>Reactive MCS</th>
<th>Evolving MCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>- presented at ECAI 2014</td>
<td>- presented at ECAI 2014</td>
</tr>
<tr>
<td>- developed in Leipzig</td>
<td>- developed in Lisbon</td>
</tr>
<tr>
<td>- equilibrium of one “step” is base kb in next “step”</td>
<td>- utilise a “next” operator</td>
</tr>
</tbody>
</table>

---

S. Ellmuthaler

Stream Reasoning Workshop 2016

CSI Leipzig 2 / 20
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 continuous 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 continuous 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 continuous 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 continuous 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

- **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
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

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

- \( C = \langle C_1, \ldots, C_n \rangle \) is a tuple of contexts \( C_i = \langle L_i, OP_i, mng_i \rangle \):
  - \( L_i = \langle KB_i, BS_i, acc_i \rangle \) is a logic,
  - \( OP_i \) is a set of operations,
  - \( mng_i : 2^{OP_i} \times KB \to KB \) is a management function.

- \( IL = \langle IL_1, \ldots, IL_k \rangle \) is a tuple of input languages;

- \( BR = \langle BR_1, \ldots, BR_n \rangle \) is a tuple such that each \( BR_i \) is a set of bridge rules for \( C_i \) over \( C \) and \( IL \) of the form

\[
\text{op} \leftarrow a_1, \ldots, a_j, \text{not } a_{j+1}, \ldots, \text{not } a_m
\]

- \( \text{op} = \text{op} \text{ or } \text{op} = \text{next}(\text{op}) \) for \( \text{op} \in OP_i \).
- and every atom \( a_\ell \) is a context atom \( c:b \) or an input atom \( s::b \).
Definition (Reactive Multi-Context System)

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

- \( C = \langle C_1, \ldots, C_n \rangle \) is a tuple of contexts \( C_i = \langle L_i, OP_i, mng_i \rangle \)
  - \( L_i = \langle KB_i, BS_i, acc_i \rangle \) is a logic,
  - \( OP_i \) is a set of operations,
  - \( mng_i : 2^{OP_i} \times KB \rightarrow KB \) is a management function.

- \( IL = \langle IL_1, \ldots, IL_k \rangle \) is a tuple of input languages;

- \( BR = \langle BR_1, \ldots, BR_n \rangle \) is a tuple such that each \( BR_i \) is a set of bridge rules for \( C_i \) over \( C \) and \( IL \) of the form

\[
  op \leftarrow a_1, \ldots, a_j, \text{not } a_{j+1}, \ldots, \text{not } a_m
\]

- \( op = op \) or \( op = \text{next}(op) \) for \( op \in OP_i \).
- and every atom \( a_\ell \), is a context atom \( c:b \) or an input atom \( s::b \).
A reactive Multi-Context System (rMCS) is a tuple $M = \langle C, IL, BR \rangle$, where

- $C = \langle C_1, \ldots, C_n \rangle$ is a tuple of contexts $C_i = \langle L_i, OP_i, mng_i \rangle$
  - $L_i = \langle KB_i, BS_i, acc_i \rangle$ is a logic,
  - $OP_i$ is a set of operations,
  - $mng_i : 2^{OP} \times KB \rightarrow KB$ is a management function.

- $IL = \langle IL_1, \ldots, IL_k \rangle$ is a tuple of input languages;

- $BR = \langle BR_1, \ldots, BR_n \rangle$ is a tuple such that each $BR_i$ is a set of bridge rules for $C_i$ over $C$ and $IL$ of the form

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

  - $\text{op} = \text{op or next}\ (\text{op})$ for $\text{op} \in OP_i$.
  - and every atom $a_\ell$, is a context atom $c:b$ or an input atom $s::b$. 
A reactive Multi-Context System (rMCS) is a tuple $M = \langle C, IL, BR \rangle$, where

- $C = \langle C_1, \ldots, C_n \rangle$ is a tuple of contexts $C_i = \langle L_i, OP_i, mng_i \rangle$
  - $L_i = \langle KB_i, BS_i, acc_i \rangle$ is a logic,
  - $OP_i$ is a set of operations,
  - $mng_i : 2^{OP} \times KB \rightarrow KB$ is a management function.

- $IL = \langle IL_1, \ldots, IL_k \rangle$ is a tuple of input languages;

- $BR = \langle BR_1, \ldots, BR_n \rangle$ is a tuple such that each $BR_i$ is a set of bridge rules for $C_i$ over $C$ and $IL$ of the form

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

- $\text{op} = \text{op}$ or $\text{op} = \text{next}(\text{op})$ for $\text{op} \in OP_i$.
- and every atom $a_\ell$, is a context atom $c:b$ or an input atom $s::b$. 
Given a rMCS \( M = \langle \langle C_1, \ldots, C_n \rangle, \langle IL_1, \ldots, IL_k \rangle, \text{BR} \rangle \), with

- an initial configuration of knowledge bases \( KB = \langle kb_1, \ldots, kb_n \rangle \), such that \( kb_i \in KB_i \), for each \( i \in \{1, \ldots, n\} \), and
- an input stream (until \( \tau \)) \( I : [1..\tau] \rightarrow \text{In}_M \)
Semantics

Given

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

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

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 ($\text{next}(op)$) in applicable bridge rules
Semantics - Equilibria Stream

$M_1 \rightarrow O_1 \rightarrow E_{q1}$

$M_2 \rightarrow O_2 \rightarrow E_{q2}$

$M_3 \rightarrow O_3 \rightarrow E_{q3}$

$\ldots$
Definition (Equilibrium)

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

$$B_i \in acc_i(kb'), \text{ where } kb' = mng_i(app^\text{now}_i(I, B), kb_i).$$
Definition (Equilibria Stream)

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

- $B^t$ is an equilibrium of $M$ given $KB^t$ and $I^t$, where $KB^t$ is
  - $KB^1 = KB$
  - $KB^{t+1} = \text{upd}_M(KB^t, I^t, B^t)$, where
    - $\text{upd}_M(KB, I, B) = \langle kb'_1, \ldots, kb'_n \rangle$, such that $kb'_i = \text{mng}_i(\text{app}^n_{i\text{ext}}(I, B), kb_i)$
Consistent rMCS

Definition

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

- $M$ is **consistent** with respect to $KB$ and $I$ if there exists an equilibria stream of $M$ given $KB$ and $I$.

- $M$ is **strongly consistent** with respect to $KB$ if, for every input stream $I$ for $M$, $M$ is consistent with respect to $KB$ and $I$. 
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 $\text{acc}_i(kb) \neq \emptyset$, for every $kb \in KB_i$.
Question
Can we ensure strong consistency of a rMCS?

Definition
A context $C_i$ is **totally coherent** if $\text{acc}_i(kb) \neq \emptyset$, 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 $\text{acc}_i(kb) \neq \emptyset$, 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 $M = \langle \langle C_1, \ldots, C_n \rangle, IL, BR \rangle$ 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.
Question
What if there are no equilibria streams?
Recovering Equilibria Streams

Question
What if there are no equilibria streams?

Definition (Repair)
Let $M = \langle C, IL, BR \rangle$ be an rMCS, $KB$ a configuration of knowledge bases for $M$, and $I$ an input stream for $M$ until $\tau$. 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 $I$ is a function $R : [1..\tau] \rightarrow 2^{br_M}$ such that there exists a function $B : [1..\tau] \rightarrow Bel_M$ such that

- $B^t$ is an equilibrium of $M[R^t]$ given $KB^t$ and $I^t$, with $KB^t$ inductively defined as
  - $KB^1 = KB$
  - $KB^{t+1} = upd_{M[R^t]}(KB^t, I^t, B^t)$,
On repairs of rMCS composed of totally coherent contexts

**Proposition**

Let $M = \langle \langle C_1, \ldots, C_n \rangle, IL, BR \rangle$ be an rMCS such that each $C_i$ is totally coherent, $KB$ a configuration of knowledge bases for $M$, and $I$ an input stream for $M$ until $\tau$. Then, there exists $R : [1..\tau] \rightarrow 2^{br_M}$ and $B : [1..\tau] \rightarrow Bel_M$ such that $B$ is a repaired equilibria stream given $KB$, $I$ and $R$. 
Question
Are all the repairs equally good?
Types of Repairs

Question
Are all the repairs equally good?

Definition
For two repairs $R_a$ and $R_b$, we say that $R_a \leq R_b$ if $R_a^i \subseteq R_b^i$ for every $i \leq \tau$, and that $R_a < R_b$ if $R_a \leq R_b$ and $R_a^i \subset R_b^i$ 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 $I$. We say that $\mathcal{R}$ is a:

- **Minimal Repair** if there is no repair $\mathcal{R}_a$ for $M$ given $KB$ and $I$ 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 $I$ 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 $\mathcal{R}^i_a \subset \mathcal{R}^i$ for every $i \leq j$. 
**Partial Equilibria Stream**

**Question**

What if there are no repairs?

---

**Definition (Partial Equilibria Stream)**

Let $M = \langle C, IL, BR \rangle$ be an rMCS, $\text{KB}$ a configuration of knowledge bases for $M$, and $I$ an input stream for $M$ until $\tau$. A partial equilibria stream of $M$ given $\text{KB}$ and $I$ is a partial function $B: [1..\tau] \not\rightarrow \text{Bel}_M$ such that $B_t$ is an equilibrium of $M$ given $\text{KB}_t$ and $I_t$, or $B_t$ is undefined otherwise.

$\text{KB}_t$ inductively defined as $\text{KB}_1 = \text{KB}$ and $\text{KB}_{t+1} = \{ \text{upd}_M(\text{KB}_t, I_t, B_t) \text{, if } B_t \text{ is not undefined.}\}$.
Partial Equilibria Stream

Question

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

Definition (Partial Equilibria Stream)

Let $M = \langle C, IL, BR \rangle$ be an rMCS, $KB$ a configuration of knowledge bases for $M$, and $I$ an input stream for $M$ until $\tau$. A partial equilibria stream of $M$ given $KB$ and $I$ is a partial function $B: [1..\tau] \not\rightarrow Bel$ $M$ such that $B_t$ is an equilibrium of $M$ given $KB_t$ and $I_t$, or $B_t$ is undefined otherwise.

$KB_t$ inductively defined as $KB_1 = KB_1$, $KB_{t+1} = \{ upd_M(KB_t, I_t, B_t), \text{if } B_t \text{ is not undefined} \}$. $KB_t$, otherwise.
Partial Equilibria Stream

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

Definition (Partial Equilibria Stream)

Let $M = \langle C, IL, BR \rangle$ be an rMCS, $KB$ a configuration of knowledge bases for $M$, and $I$ an input stream for $M$ until $\tau$. A partial equilibria stream of $M$ given $KB$ and $I$ is a partial function $B : [1..\tau] \not\rightarrow Bel_M$ such that

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

$KB^t$ inductively defined as

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

**Proposition**

Every equilibria stream of $M$ given $KB$ and $I$ is a partial equilibria stream of $M$ given $KB$ and $I$. 
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 $\tau$. Then, there exists $B : [1..\tau] \rightarrow Bel_M$ such that $B$ 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!