Advancing Multi-Context Systems by Inconsistency Management

Antonius Weinzierl

Knowledge-based Systems Group
Vienna University of Technology

July 19, 2011 @ RuleML

Supported by the Vienna Science and Technology Fund (WWTF) under grant ICT 08-020.
Motivation

- **Large variety** of languages/formats/tools for knowledge representation:
  - Databases, triple-stores, ontologies, temporal and modal logics, nonmonotonic logics, answer-set programs, …

- **How to benefit from diversity?**
- **How to access heterogeneous knowledge sources?**
Motivation

- **Large variety** of languages/formats/tools for knowledge representation:
  - Databases, triple-stores, ontologies, temporal and modal logics, nonmonotonic logics, answer-set programs, . . .

- **How to benefit from diversity?**

- **How to access heterogeneous knowledge sources?**

- **Multi-Context Systems (MCS) framework** for interlinking heterogeneous knowledge bases.
Motivation

- **Large variety** of languages/formats/tools for knowledge representation:
  - Databases, triple-stores, ontologies, temporal and modal logics, nonmonotonic logics, answer-set programs, ...

- How to **benefit from diversity**?

- How to access heterogeneous knowledge sources?

- Multi-Context Systems (MCS) framework for **interlinking heterogeneous knowledge bases**.

- Knowledge exchange between (previously unrelated) sources.
  - But: Unforeseen side-effects, e.g., violation of constraints.
  - Inconsistent system useless.
  - Inconsistency management needed.
Hospital Example

Example

- Patient records (relDB), disease ontology, expert system.
- Patient Sue: X-Ray indicates pneumonia, blood marker present, and allergic to strong antibiotics.
- Bridge rules for ontology:
  
  \[ (C_{onto} : xray(Sue)) \leftarrow (C_{patients} : labresult(Sue, xray)). \]
  
  \[ (C_{onto} : marker(Sue)) \leftarrow (C_{patients} : labresult(Sue, marker)). \]

- Ontology: \( \{xray \sqcap marker \sqsubseteq atyp\_pneu\} \), concludes: \( atyp\_pneu(Sue) \).

- Expert system (logic program):

  \[ give\_weak \lor give\_strong : \neg pneumonia. \]
  
  \[ give\_strong : \neg atyp\_pneumonia. \]
  
  \[ : \neg give\_strong, not\ allowed\_strong. \]
Hospital Example

Example (ctd.)

- Further bridge rules for expert system:
  \[
  (C_{\text{expert}} : pneumonia) \leftarrow (C_{\text{onto}} : pneumonia(Sue)) \\
  (C_{\text{expert}} : atyp\_pneumonia) \leftarrow (C_{\text{onto}} : atyp\_pneu(Sue)) \\
  (C_{\text{expert}} : allowed\_strong) \leftarrow \neg (C_{\text{patients}} : allergy(Sue, strong\_ab))
  \]


\[\Rightarrow\] No answer (acceptable belief set) for program:

\[
give\_weak \lor give\_strong : \neg pneumonia. \\
give\_strong : \neg atyp\_pneumonia. \\
\quad : \neg give\_strong, not allowed\_strong.
\]

\[\Rightarrow\] MCS is inconsistent (no equilibrium).

- Multiple formalisms (contexts) involved.
- No obvious “right” repair.
Research Issues

- Nonmonotonic, heterogeneous setting.
- Need uniform description of inconsistency.
Research Issues

- Nonmonotonic, heterogeneous setting.
- Need uniform description of inconsistency.
- Multiple repairs likely $\Rightarrow$ assess them, find most preferred ones.
  - Without committing to specific preference formalism?
  - Different formalisms for parts of an MCS?

Resolve inconsistency (purely) locally?
- Consistency guaranteed?
  - Integrate formalism-specific inconsistency methods (belief revision, paraconsistent semantics, etc).

Besides inconsistency: test versatility of MCS, e.g., knowledge exchange with SPARQL?
- Foundational perspective.

Research started 2 years ago, we have some answers!
Research Issues

- Nonmonotonic, heterogeneous setting.
- Need uniform description of inconsistency.
- Multiple repairs likely ⇒ assess them, find most preferred ones.
  - Without committing to specific preference formalism?
  - Different formalisms for parts of an MCS?
- Resolve inconsistency (purely) locally?
  - Consistency guaranteed?
- Integrate formalism-specific inconsistency methods (belief revision, paraconsistent semantics, etc).
Research Issues

- Nonmonotonic, heterogeneous setting.
- Need uniform description of inconsistency.
- Multiple repairs likely $\Rightarrow$ assess them, find most preferred ones.
  - Without committing to specific preference formalism?
  - Different formalisms for parts of an MCS?
- Resolve inconsistency (purely) locally?
  - Consistency guaranteed?
- Integrate formalism-specific inconsistency methods (belief revision, paraconsistent semantics, etc).
- Besides inconsistency: test versatility of MCS, e.g., knowledge exchange with SPARQL?
Research Issues

- Nonmonotonic, heterogeneous setting.
- Need uniform description of inconsistency.
- Multiple repairs likely ⇒ assess them, find most preferred ones.
  - Without committing to specific preference formalism?
  - Different formalisms for parts of an MCS?
- Resolve inconsistency (purely) locally?
  - Consistency guaranteed?
- Integrate formalism-specific inconsistency methods (belief revision, paraconsistent semantics, etc).
- Besides inconsistency: test versatility of MCS, e.g., knowledge exchange with SPARQL?
  ⇒ Foundational perspective.
- Research started 2 years ago, we have some answers!
Related Work

- History of MCS:

Defeasible rules in MCS (Bikakis and Antoniou, 2009):
- preference-based inconsistency removal,
- provenance-based,
- no deeper inconsistency analysis,
- no information hiding.

Peer-to-Peer systems (e.g., Calvanese et al., 2008, Serafini et al., 2003):
- isolate faulty peers,
- ignore their information,
- no overall consistency,
- no heterogeneity.

Information integration (e.g., Bleiholder and Naumann, 2007):
- Single database as result, usually relational,
- no heterogeneous framework.

Inconsistency handling for specific formalisms:
- belief revision,
- possibilistic reasoning,
- works only for certain formalism.
Related Work

- History of MCS:


- Peer-to-Peer systems (e.g., Calvanese et al., 2008, Serafini et al., 2003): isolate faulty peers, ignore their information, No overall consistency, no heterogeneity.
Related Work

- **History of MCS:**

- **Defeasible rules in MCS** (Bikakis and Antoniou, 2009):
  preference-based inconsistency removal, provenance-based, no deeper inconsistency analysis, no information hiding.

- **Peer-to-Peer systems** (e.g., Calvanese et al., 2008, Serafini et al., 2003):
  isolate faulty peers, ignore their information, No overall consistency, no heterogeneity.

- **Information integration** (e.g., Bleiholder and Naumann, 2007):
  Single database as result, usually relational, no heterogeneous framework.

- **Inconsistency handling for specific formalisms:** belief revision, possibilistic reasoning, works only for certain formalism.
Multi-Context Systems

- **Logic**: \( L = (KB_L, BS_L, ACC_L) \), where
  - \( KB_L \) set of knowledge bases (sets of “wf formulas”),
  - \( BS_L \) set of possible belief sets (“accepted theorems”), and
  - \( ACC_L : KB_L \rightarrow 2^{BS_L} \) semantics.

- **Multi-Context System (MCS)** \( M = (C_1, \ldots, C_n) \) collection of contexts \( C_i = (L_i, kb_i, br_i) \), where
  - \( L_i \) a logic,
  - \( kb_i \in KB_{L_i} \) a knowledge base, and
  - \( br_i \) a set of bridge rules.

- **Bridge rules** of form:
  \[
  (k: s) \leftarrow (c_1: p_1), \ldots, (c_j: p_j), \text{not}(c_{j+1}: p_{j+1}), \ldots, \text{not}(c_m: p_m).
  \]
  - such that \( kb \cup \{s\} \) is an element of \( KB_{L_k} \),
  - \( c_\ell \in \{1, \ldots, n\} \), and
  - \( p_\ell \) is element of some belief set of \( BS_{c_\ell} \), for all \( 1 \leq \ell \leq m \).
MCS Semantics

- Given MCS $M = (C_1, \ldots, C_n)$.
- Belief state: $S = (S_1, \ldots, S_n)$ belief set for each context, $S_i \in BS_i$ for $i = 1, \ldots, n$.
- $(k: s) \leftarrow (c_1: p_1), \ldots, (c_j: p_j)$, $\neg(c_{j+1}: p_{j+1}), \ldots, \neg(c_m: p_m)$.
  Applicability: $S \models body(r)$ iff $p_\ell \in S_{c_\ell}$ for $1 \leq \ell \leq j$ and $p_\ell \notin S_{c_\ell}$ for $j < \ell \leq m$.
- Heads of all applicable bridge rules of $C_i$:
  $$\text{app}_i(S) = \{hd(r) \mid r \in br_i \land S \models body(r)\}$$
- Equilibrium: $S = (S_1, \ldots, S_n)$ such that $\forall i \in \{1, \ldots, n\}$:
  $$S_i \in ACC_i(kb_i \cup \text{app}_i(S))$$
Methodology

- Analogy to existing notions: diagnosis/explanation inspired by Reiter.
- Algorithms: reduction to computational logic, meta-reasoning, e.g., for evaluating prototypes or preference handling.
- Open notions: enable user to instantiate with best fitting formalism, e.g., for local inconsistency management.
- Prototypes: extensive (random) benchmarks.
Explanations of Inconsistency

- Characterize inconsistency by involved bridge rules.
- Explanation: indicate sources of inconsistency (separates multiple).
- Diagnosis: indicates possible repairs.

Example (ctd.)

Intuitively, inconsistency caused by information flow of $r_1$, $r_2$, $r_4$ and $r_5$ not firing.

$r_1: (C_{onto}: \text{xray}(Sue)) \leftarrow (C_{patients}: \text{labresult}(Sue, xray))$.

$r_2: (C_{onto}: \text{marker}(Sue)) \leftarrow (C_{patients}: \text{labresult}(Sue, marker))$.

$r_3: (C_{expert}: \text{pneumonia}) \leftarrow (C_{onto}: \text{pneumonia}(Sue))$

$r_4: (C_{expert}: \text{atyp}_p\text{neumonia}) \leftarrow (C_{onto}: \text{atyp}_p\text{neu}(Sue))$

$r_5: (C_{expert}: \text{allowed}_\text{strong}) \leftarrow \text{not}(C_{patients}: \text{allergy})$

- Minimal diagnoses: ($\{r_1\}, \emptyset$) ignore x-ray, ($\{r_4\}, \emptyset$) ignore atypical pneumonia, ($\emptyset, \{r_5\}$) ignore allergy, ...
- Minimal explanation: ($\{r_1, r_2, r_4\}, \{r_5\}$).
Inconsistency Assessment

Example (ctd.)

- MCS is extended by accounting.
- Let reason for absence of \textit{allowed\_strong} be at accounting.
- Goal 1: Forbid diagnoses ignoring patient allergies.
- Goal 2: Prefer healthy patients over correct accounting.
Contributions

Inconsistency Assessment

Example (ctd.)

- MCS is extended by accounting.
- Let reason for absence of *allowed_strong* be at accounting.
- Goal 1: Forbid diagnoses ignoring patient allergies.
- Goal 2: Prefer healthy patients over correct accounting.

- Focus on subset-minimal diagnoses.
- Meta-reasoning transformation: observe applied diagnoses.
  \[\Rightarrow\] Multiple observer contexts (arbitrary/best fitting formalism).
- Filter undesired diagnoses (making observer inconsistent).
- Apply (arbitrary) preference formalism (map preference to bridge rules).
Contributions

Local Inconsistency Management

- Extend each context with general management function $mng_i$ ⇒ managed Multi-Context Systems.
- Arbitrary manipulation of knowledge base (wrt. applicable rules).

Sketch

Belief state $S = (S_1, \ldots, S_n)$ is equilibrium iff for all $1 \leq i \leq n$ there exists $(kb'_i, ACC_i) \in mng_i(app_i(S), kb_i)$ such that $S_i \in ACC_i(kb'_i)$.

- Covers belief revision, logic program updates, database manipulation, switching to paraconsistent semantics (each per context).
Local Inconsistency Management

- Extend each context with general management function $mng_i$ ⇒ managed Multi-Context Systems.
- Arbitrary manipulation of knowledge base (wrt. applicable rules).

**Sketch**

Belief state $S = (S_1, \ldots, S_n)$ is equilibrium iff for all $1 \leq i \leq n$ there exists $(kb'_i, ACC_i) \in mng_i(app_i(S), kb_i)$ such that $S_i \in ACC_i(kb'_i)$.

- Covers belief revision, logic program updates, database manipulation, switching to paraconsistent semantics (each per context).

⇒ If all contexts always have acceptable belief sets, then
  - equilibrium still not guaranteed.
  - cycles are only source of inconsistency.
  - acyclic mMCS have equilibrium.
Contributions

- Uniform representation of inconsistency (Eiter et al., KR 2010):
  - Inconsistency explanation and diagnosis.
Contributions

- Uniform representation of inconsistency (Eiter et al., KR 2010):
  - Inconsistency explanation and diagnosis.

- Preference handling by meta-reasoning transformation (Eiter et al., JELIA 2010):
  - Allows arbitrary preference formalism (also for parts of an MCS).
Contributions

- Uniform representation of inconsistency (Eiter et al., KR 2010):
  - Inconsistency explanation and diagnosis.

- Preference handling by meta-reasoning transformation (Eiter et al., JELIA 2010).
  - Allows arbitrary preference formalism (also for parts of an MCS).

  - Management component at each context.
  - Employ legacy systems/methods for inconsistency handling (belief revision, updates, etc).
Contributions

- Uniform representation of inconsistency (Eiter et al., KR 2010):
  - Inconsistency explanation and diagnosis.

- Preference handling by meta-reasoning transformation (Eiter et al., JELIA 2010).
  - Allows arbitrary preference formalism (also for parts of an MCS).

  - Management component at each context.
  - Employ legacy systems/methods for inconsistency handling (belief revision, updates, etc).

- Computational complexity analysis.

- Versatility: SPARQL-MCS with SPARQL queries as bridges (Schüller and W., SSW 2011).
Acknowledgements

I am very grateful to my advisor Thomas Eiter, the principal investigator of our research project Michael Fink, and my colleague Peter Schüller who provided guidance, and helped with many fruitful discussions.
Conclusion and Future Work

- We answered several foundational questions.
- Methods for inconsistency assessment.
- Local (specialized) inconsistency handling.
- Complexity results.

Future work:

- Optimized evaluation of MCS (avoid grounding of bridge rules).
- Investigate approximations.

⇒ Write thesis.