Distributed Nonmonotonic Multi-Context Systems

Minh Dao-Tran   Thomas Eiter
Michael Fink   Thomas Krennwallner

KBS Group, Institute of Information Systems, Vienna University of Technology

KR 2010 - May 13, 2010

Knowledge-Based Systems Group

FAKULTÄT FÜR !NFORMATIK

Overview

Multi-Context Systems

Distributed Algorithm for Evaluating Nonmonotonic MCS

Loop Formulas for Multi-Context Systems

Experiments

Conclusions

Multi-Context Systems (MCS)

- MCSen introduced by [Giunchiglia and Serafini, 1994]:
  - represent inter-contextual information flow
  - express reasoning w.r.t. contextual information
  - allow decentralized, pointwise information exchange
  - monotonic, homogeneous logic
- Framework extended for integrating heterogeneous and nonmonotonic logics [Brewka and Eiter, 2007]

Syntax of Multi-Context Systems

- multi-context system
  - a collection $M = (C_1, \ldots, C_n)$ of contexts
- context $C_i = (L_i, kb_i, br_i)$
  - L_i: a logic
  - kb_i: a knowledge base of logic L_i
  - br_i: a set of bridge rules

Syntax of Multi-Context Systems

- multi-context system
  - a collection $M = (C_1, \dots, C_n)$ of contexts
- context $C_i = (L_i, kb_i, br_i)$
  - L_i: a logic
  - kb_i: a knowledge base of logic L_i
  - br_i: a set of bridge rules
- logic $L = (\mathbf{KB}_L, \mathbf{BS}_L, \mathbf{ACC}_L)$
  - $\mathbf{KB}_L$: set of well-formed knowledge bases
  - $\mathbf{BS}_L$: is the set of possible belief sets
  - $\mathbf{ACC}_L$: acceptability function $\mathbf{KB}_L \mapsto 2^{\mathbf{BS}_L}$
    Which belief sets are accepted by a knowledge base?

Semantics of Multi-Context Systems (2)

- *multi-context system*

$$M = (C_1, \ldots, C_n)$$

- *context*

$$C_i = (L_i, kb_i, br_i)$$

- *logic*

$$L = (\mathbf{KB}_i, \mathbf{BS}_i, \mathbf{ACC}_i)$$

Syntax of Multi-Context Systems (bridge rules)

- *multi-context system*
$$M = (C_1, \ldots, C_n)$$
- *context*
$$C_i = (L_i, kb_i, br_i)$$
- *logic*
$$L_i = (\mathbf{KB}_i, \mathbf{BS}_i, \mathbf{ACC}_i)$$

- Bridge rule $r \in br_i$ of a context C_i

$$\begin{aligned} s \leftarrow (c_1 : p_1), \ldots, (c_j : p_j), \\ \mathit{not}\,(c_{j+1} : p_{j+1}), \ldots, \mathit{not}\,(c_m : p_m) \end{aligned}$$

  - $(c_k : p_k)$ looks at belief p_k in context C_{c_k}
  - r is applicable :⇔ positive/negative beliefs are present/absent
  - we add the head s to kb_i if r is applicable

Semantics of Multi-Context Systems

- *multi-context system*
$$M = (C_1, \ldots, C_n)$$
- *context*
$$C_i = (L_i, kb_i, br_i)$$
- *logic*
$$L_i = (\mathbf{KB}_i, \mathbf{BS}_i, \mathbf{ACC}_i)$$

- *knowledge base* of a context C_i
$$kb_i \in \mathbf{KB}_i$$

- *set of bridge rules* br_i of a context C_i of form
$$s \leftarrow (c_1 : p_1), \ldots, (c_j : p_j), not\,(c_{j+1} : p_{j+1}), \ldots, not\,(c_m : p_m)$$

- Contexts $C_1, \ldots, C_n$ are knowledge bases with semantics in terms of accepted belief sets

- $S = (S_1, \ldots, S_n)$ is a belief state of M with each $S_i \in \mathbf{BS}_i$

Semantics of Multi-Context Systems

- *multi-context system*

$$M = (C_1, \ldots, C_n)$$

- *context*

$$C_i = (L_i, kb_i, br_i)$$

- *logic*

$$L_i = (\mathbf{KB}_i, \mathbf{BS}_i, \mathbf{ACC}_i)$$

- Equilibrium semantics
  - A belief state $S = (S_1, \ldots, S_n)$ with $S_i \in \mathbf{BS}_i$
    ...makes certain bridge rules applicable,
    ...add applicable bridge heads to kb_i

$\Rightarrow$ S is an equilibrium :$\Leftrightarrow$
each kb_i plus acceptable bridge heads from br_i accepts S_i

$$S_i \in \mathbf{ACC}_i(kb_i \cup \{head(r) \mid r \in app(br_i, S)\})$$

Toward Distributed Equilibria building for MCS

Obstacles:

- abstraction of contexts
- information hiding and security aspects
- lack of system topology
- cycles between contexts

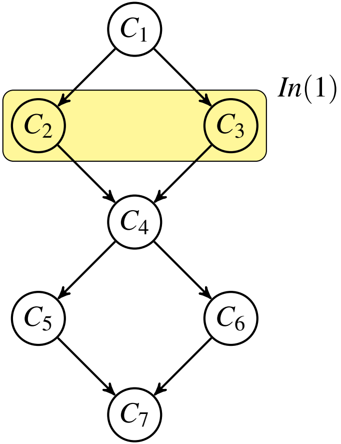
We need to capture:

- dependencies between contexts
- representation of partial knowledge
- combination/join of local results

Import Closure

Import neighborhood of C_k

$$In(k) = \{c_i \mid (c_i : p_i) \in B(r), r \in br_k\}$$



Import Closure

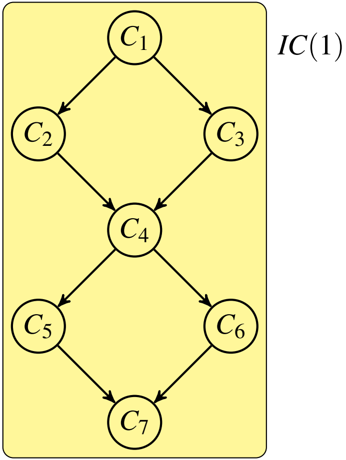
Import neighborhood of C_k

$$In(k) = \{c_i \mid (c_i : p_i) \in B(r), r \in br_k\}$$

Import closure $IC(k)$ of C_k is the smallest set S such that
(i) $k \in S$ and
(ii) for all $i \in S$, $In(i) \subseteq S$.
Alternatively,

$$IC(k) = \{k\} \cup \bigcup_{j \geq 0} IC^j(k) \quad ,$$

where
$IC^0(k) = In(k)$, and
$IC^{j+1}(k) = \bigcup_{i \in IC^j(k)} In(i)$.



Partial Belief States and Equilibria

Let $M = (C_1, \ldots, C_n)$ be an MCS, and let $\epsilon \notin \bigcup_{i=1}^{n} \mathbf{BS}_i$

Partial Belief States and Equilibria

Let $M = (C_1, \ldots, C_n)$ be an MCS, and let $\epsilon \notin \bigcup_{i=1}^{n} \mathbf{BS}_i$

A partial belief state of M is a sequence $S = (S_1, \ldots, S_n)$, where $S_i \in \mathbf{BS}_i \cup \{\epsilon\}$, for $1 \leq i \leq n$

Partial Belief States and Equilibria

Let $M = (C_1, \ldots, C_n)$ be an MCS, and let $\epsilon \notin \bigcup_{i=1}^{n} \mathbf{BS}_i$

A partial belief state of M is a sequence $S = (S_1, \ldots, S_n)$, where $S_i \in \mathbf{BS}_i \cup \{\epsilon\}$, for $1 \leq i \leq n$

$S = (S_1, \ldots, S_n)$ is a partial equilibrium of M w.r.t. a context C_k iff for $1 \leq i \leq n$,

- if $i \in IC(k)$ then $S_i \in \mathbf{ACC}_i(kb_i \cup \{head(r) \mid r \in app(br_i, S)\})$
- otherwise, $S_i = \epsilon$

Joining Partial Belief States

Join $S \bowtie T$ of belief sets S and T: like join of tuples in a database.

$S =$	S_1	$\cdots$	ϵ	$\cdots$	ϵ	$\cdots$	S_n
$T =$	ϵ	$\cdots$	ϵ	$\cdots$	T_i	$\cdots$	T_n
$S \bowtie T =$	S_1	$\cdots$	ϵ	$\cdots$	T_i	$\cdots$	$S_n = T_n$

$S \bowtie T$ is undefined, if $\epsilon \neq S_j \neq T_j \neq \epsilon$ for some j.

$$\mathcal{S} \bowtie \mathcal{T} = \{S \bowtie T \mid S \in \mathcal{S}, T \in \mathcal{T}\}$$

Algorithm DMCS

Input: an MCS M and a starting context C_k
Output: all partial equilibria of M w.r.t. C_k

Algorithm DMCS

Input: an MCS M and a starting context C_k
Output: all partial equilibria of M w.r.t. C_k

Requirement: solver lsolve(S) for each context C_k is available which computes $\mathbf{ACC}_k(kb_k \cup app_k(S))$

Algorithm DMCS

Input: an MCS M and a starting context C_k
Output: all partial equilibria of M w.r.t. C_k

Requirement: solver lsolve(S) for each context C_k is available which computes $\mathbf{ACC}_k(kb_k \cup app_k(S))$

Input parameters for DMCS:

- V: set of “interesting” variables (to project the partial equilibria)
- $hist$: visited path

Algorithm DMCS

Input: an MCS M and a starting context C_k
Output: all partial equilibria of M w.r.t. C_k

Requirement: solver lsolve(S) for each context C_k is available which computes $\mathbf{ACC}_k(kb_k \cup app_k(S))$

Input parameters for DMCS:

- V: set of "interesting" variables (to project the partial equilibria)
- $hist$: visited path

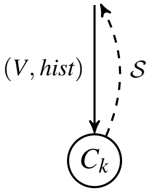
Strategy: DFS-traversal of M starting with C_k, visiting all C_i for $i \in IC(k)$

Instances of DMCS

- running at each context node,
- communicating with each other for exchanging sets of belief states

Acyclic case

Leaf context C_k ($br_k = \emptyset$)

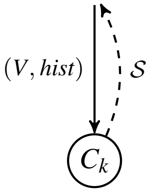


lsolve$((\epsilon, \dots, \epsilon)) = \mathcal{S}$

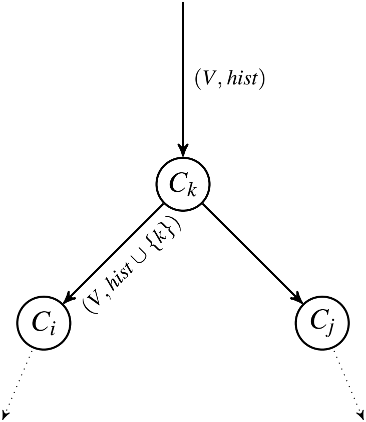
Acyclic case

Intermediate context C_k
($(i : p), (j : q)$ appear in br_k)

Leaf context C_k ($br_k = \emptyset$)



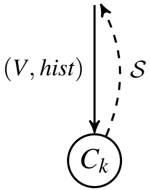
Isolve$((\epsilon, \ldots, \epsilon)) = \mathcal{S}$



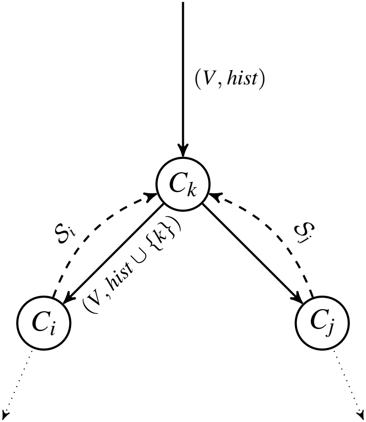
Acyclic case

Intermediate context C_k
($(i : p), (j : q)$ appear in br_k)

Leaf context C_k ($br_k = \emptyset$)



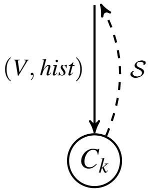
Isolve$((\epsilon, \dots, \epsilon)) = \mathcal{S}$



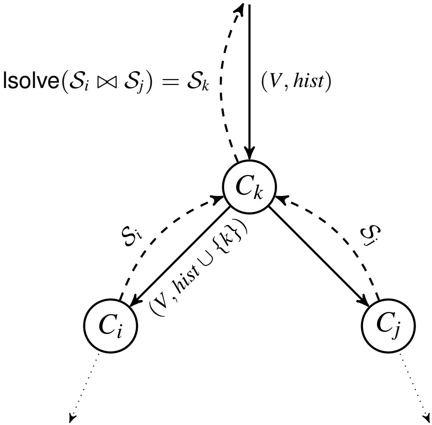
Acyclic case

Intermediate context C_k
($(i : p), (j : q)$ appear in br_k)

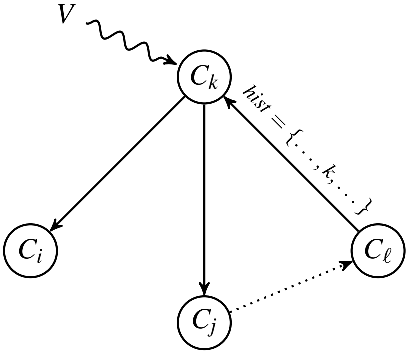
Leaf context C_k ($br_k = \emptyset$)



lsolve$((\epsilon, \ldots, \epsilon)) = \mathcal{S}$

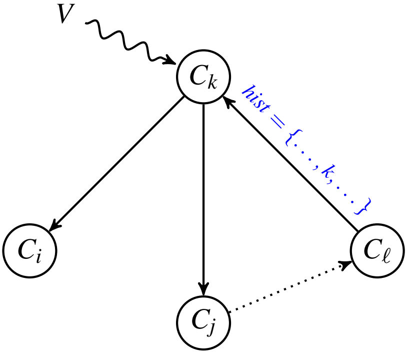


Cycle breaking



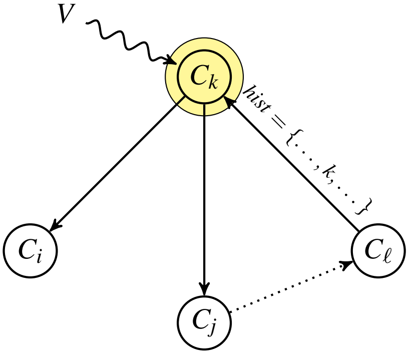
Cycle breaking

C_k detects a cycle in *hist*



Cycle breaking

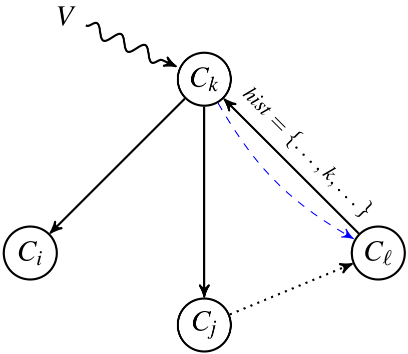
C_k detects a cycle in *hist*



- Guessing local belief sets

Cycle breaking

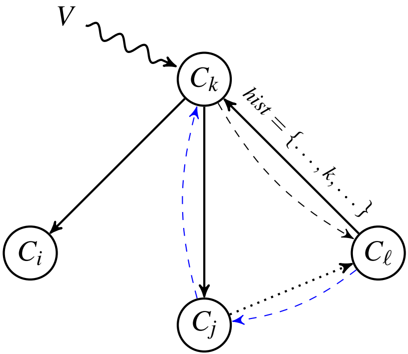
C_k detects a cycle in *hist*



- Guessing local belief sets
- return them to invoking context

Cycle breaking

C_k detects a cycle in $hist$

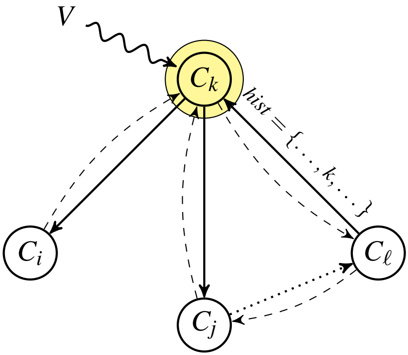


- Guessing local belief sets
- return them to invoking context

- on the way back, partial belief states w.r.t. bad guesses will be pruned by $\bowtie$

Cycle breaking

C_k detects a cycle in $hist$

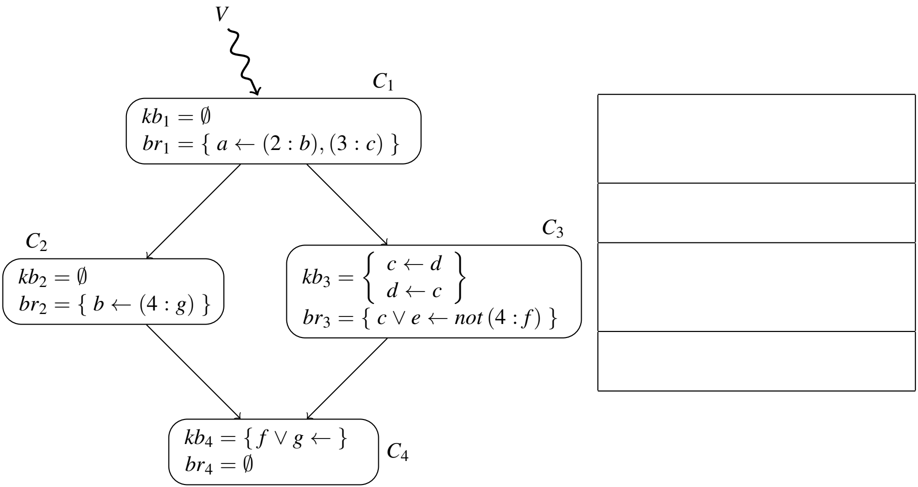


- Guessing local belief sets
- return them to invoking context

- on the way back, partial belief states w.r.t. bad guesses will be pruned by $\bowtie$
- eventually, C_k will remove wrong guesses by calling lsolve on each received partial belief state

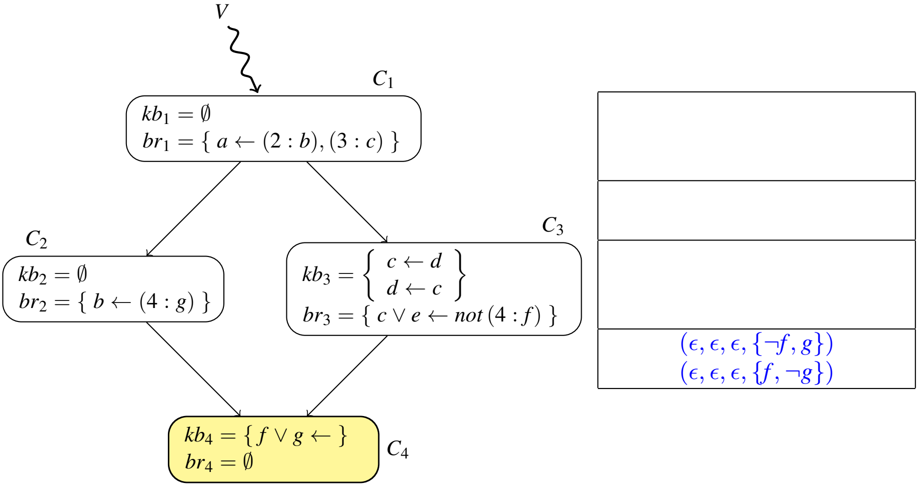
Example

A run with C_1.DMCS$(V, \emptyset)$, where $V = \{a, b, c, f, g\}$.



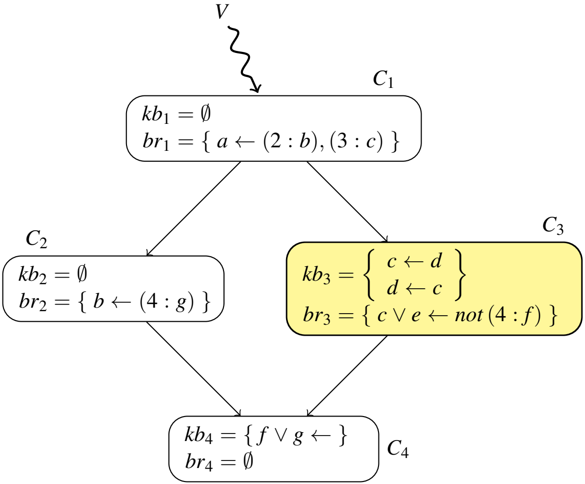
Example

A run with C_1.DMCS$(V, \emptyset)$, where $V = \{a, b, c, f, g\}$.



Example

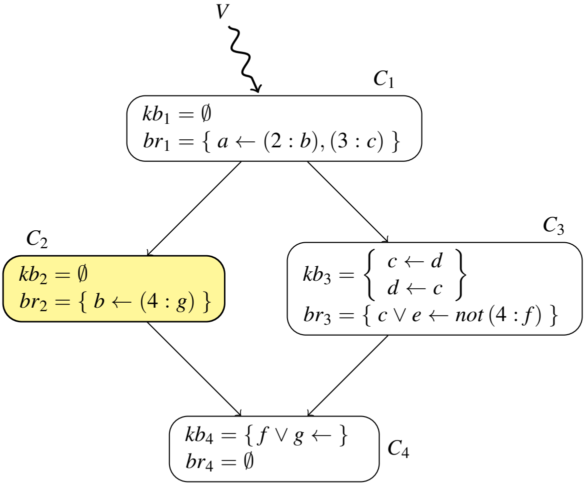
A run with C_1.DMCS$(V, \emptyset)$, where $V = \{a, b, c, f, g\}$.



$(\epsilon, \epsilon, \{\neg c, \neg d, e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{c, d, \neg e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{\neg c, \neg d, \neg e\}, \{f, \neg g\})$
$(\epsilon, \epsilon, \epsilon, \{\neg f, g\})$ $(\epsilon, \epsilon, \epsilon, \{f, \neg g\})$

Example

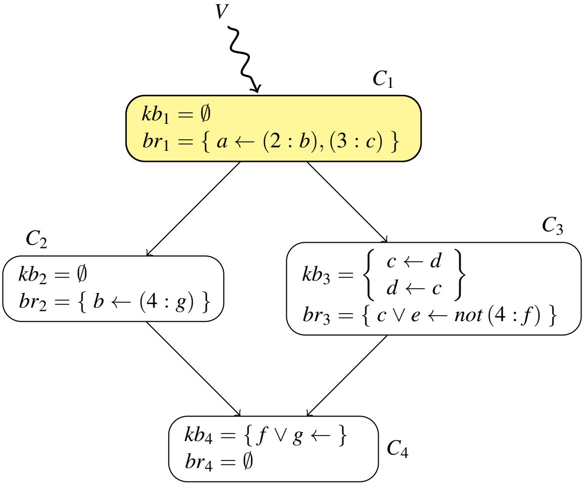
A run with C_1.DMCS$(V, \emptyset)$, where $V = \{a, b, c, f, g\}$.



$(\epsilon, \{b\}, \epsilon, \{\neg f, g\})$ $(\epsilon, \{\neg b\}, \epsilon, \{f, \neg g\})$
$(\epsilon, \epsilon, \{\neg c, \neg d, e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{c, d, \neg e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{\neg c, \neg d, \neg e\}, \{f, \neg g\})$
$(\epsilon, \epsilon, \epsilon, \{\neg f, g\})$ $(\epsilon, \epsilon, \epsilon, \{f, \neg g\})$

Example

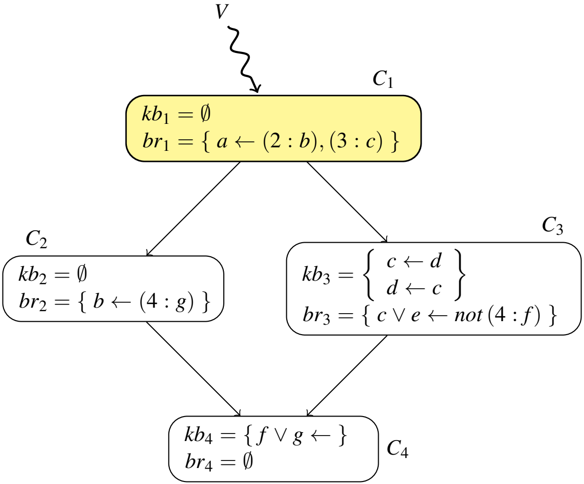
A run with C_1.DMCS$(V, \emptyset)$, where $V = \{a, b, c, f, g\}$.



$(\epsilon, \{b\}, \{c\}, \{\neg f, g\})$ $(\epsilon, \{b\}, \{\neg c\}, \{\neg f, g\})$ $(\epsilon, \{\neg b\}, \{\neg c\}, \{f, \neg g\})$
$(\epsilon, \{b\}, \epsilon, \{\neg f, g\})$ $(\epsilon, \{\neg b\}, \epsilon, \{f, \neg g\})$
$(\epsilon, \epsilon, \{\neg c, \neg d, e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{c, d, \neg e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{\neg c, \neg d, \neg e\}, \{f, \neg g\})$
$(\epsilon, \epsilon, \epsilon, \{\neg f, g\})$ $(\epsilon, \epsilon, \epsilon, \{f, \neg g\})$

Example

A run with C_1.DMCS$(V, \emptyset)$, where $V = \{a, b, c, f, g\}$.

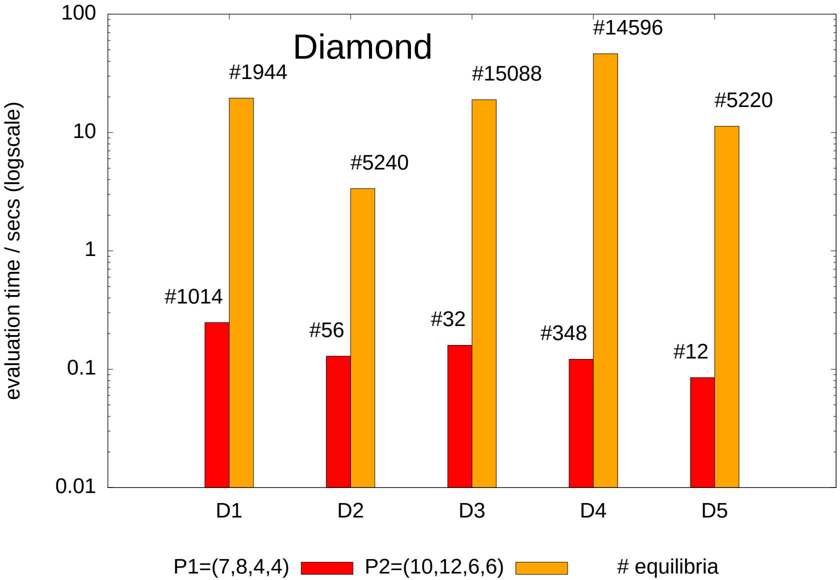


$(\{a\}, \{b\}, \{c\}, \{\neg f, g\})$ $(\{\neg a\}, \{b\}, \{\neg c\}, \{\neg f, g\})$ $(\{\neg a\}, \{\neg b\}, \{\neg c\}, \{f, \neg g\})$
$(\epsilon, \{b\}, \epsilon, \{\neg f, g\})$ $(\epsilon, \{\neg b\}, \epsilon, \{f, \neg g\})$
$(\epsilon, \epsilon, \{\neg c, \neg d, e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{c, d, \neg e\}, \{\neg f, g\})$ $(\epsilon, \epsilon, \{\neg c, \neg d, \neg e\}, \{f, \neg g\})$
$(\epsilon, \epsilon, \epsilon, \{\neg f, g\})$ $(\epsilon, \epsilon, \epsilon, \{f, \neg g\})$

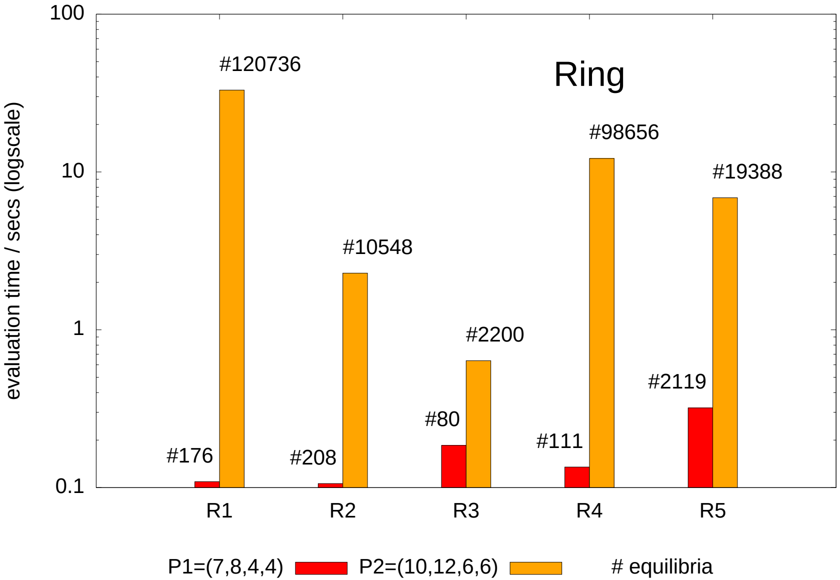
Loop Formulas for Multi-Context Systems

- DMCS is using lsolve() to incorporate the bridge rules into the local knowledge base: this must be done for every intermediate result
- Some logics allow to combine br_i and kb_i:
  - contexts with answer set programs, or
  - contexts with propositional formulas
- Benefit: a single call to a SAT solver is sufficient to compute the local semantics of a context
- This is used to adapt DMCS and provide a prototype implementation

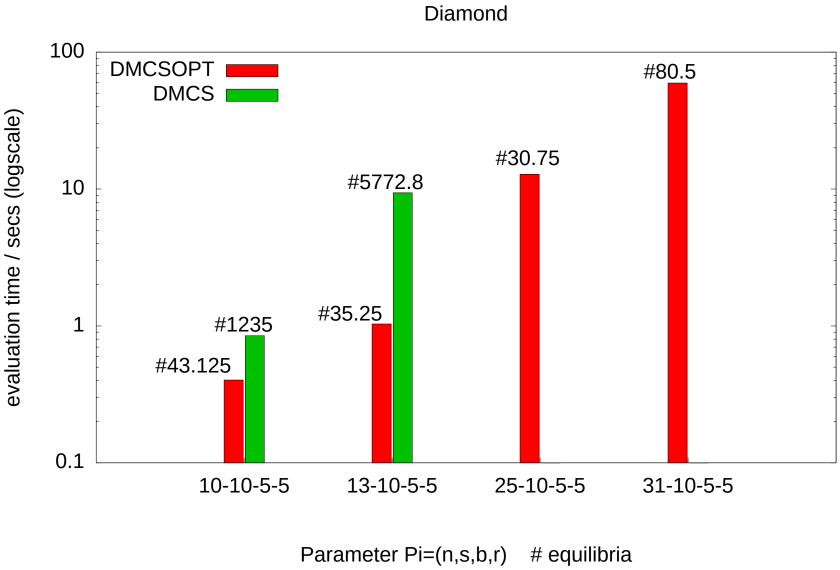
Experiments



Experiments



Experiments



Conclusions

- MCS is a general framework for integrating diverse formalisms
- First attempt for distributed MCS evaluation
- In certain settings, we can compile bridge rules away and use SAT solvers locally to generate partial equilibria (loop formulas for MCS)
- Initial experiments with a prototype implementation

Conclusions

- MCS is a general framework for integrating diverse formalisms
- First attempt for distributed MCS evaluation
- In certain settings, we can compile bridge rules away and use SAT solvers locally to generate partial equilibria (loop formulas for MCS)
- Initial experiments with a prototype implementation

Future work:

- improve scalability
  - move away from “knowing-nothing” to “knowing-something”
  - approximation semantics
  - syntactic restrictions
  - specialized algorithms for some types of topologies
- how to deal with dynamic setting?

Related work

Frameworks/Platforms

- Framework for P2P inference systems [Hirayama and Yokoo, 2005]: consequence finding v.s. model building
- MWeb [Analyti *et al.*, 2008]: scope and context for modular web rule bases on the Web

Distributed Reasoning

- Satisfiability checking for homogeneous, monotonic MCS [Roelofsen *et al.*, 2004]: (co-inductive) fixpoint strategy, not truly distributed
- DisSAT [Hirayama and Yokoo, 2005]: finding single models (randomize)
- Distributed Description Logic [Serafini and Tamilin, 2005], [Serafini *et al.*, 2005]
  - reasoning v.s. (distributed) model building
  - loose v.s. tight integration (signatures, meaning of symbols)

References I

- Anastasia Analyti, Grigoris Antoniou, and Carlos Viegas Damásio.
A principled framework for modular web rule bases and its semantics.
In *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR2008)*. AAAI Press, September 2008.

- Gerhard Brewka and Thomas Eiter.
Equilibria in heterogeneous nonmonotonic multi-context systems.
In *AAAI'07*, pages 385–390. AAAI Press, 2007.

- Fausto Giunchiglia and Luciano Serafini.
Multilanguage hierarchical logics or: How we can do without modal logics.
Artificial Intelligence, 65(1):29–70, 1994.

References II

Katsutoshi Hirayama and Makoto Yokoo.
The distributed breakout algorithms.
Artif. Intell., 161(1–2):89–115, January 2005.

Floris Roelofsen, Luciano Serafini, and Alessandro Cimatti.
Many Hands Make Light Work: Localized Satisfiability for Multi-Context Systems.
In Ramon López de Mántaras and Lorenza Saitta, editors, *16th Eureopean Conference on Artificial Intelligence (ECAI'04)*, pages 58–62. IOS Press, 2004.

Luciano Serafini and Andrei Tamilin.
Drago: Distributed reasoning architecture for the semantic web.
In *Second European Semantic Web Conference (ESWC 2005)*, pages 361–376. Springer, 2005.

References III

Luciano Serafini, Alexander Borgida, and Andrei Tamilin.
Aspects of distributed and modular ontology reasoning.
In *Nineteenth International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 570–575. AAAI Press, 2005.