Decentralized Reasoning with Inconsistencies in Peer-to-Peer Inference Systems

IASI-GEMO team

P. Chatalic, G.H. Nguyen, M.C. Rousset

UNIVERSITÉ PARIS-SUD 11

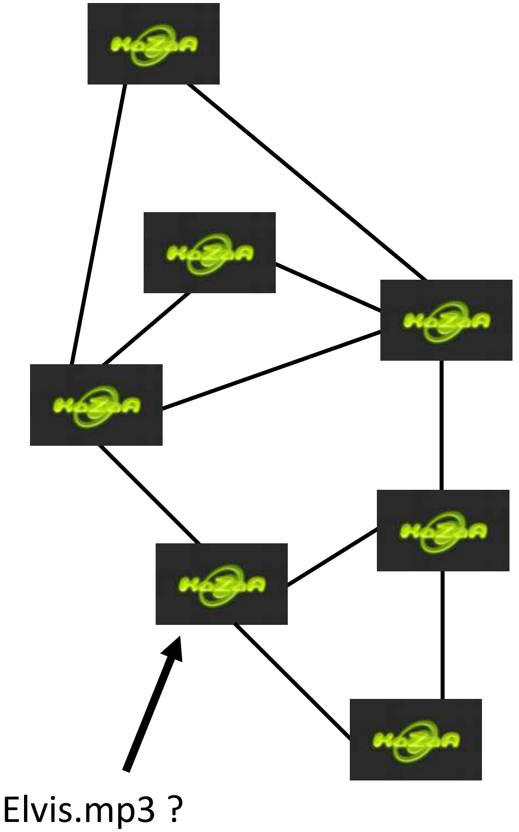
cnrs

INRIA

Outline

1. Introduction
   - Peer Data Management Systems
   - The SomeWhere platform
2. Reasoning with inconsistent peers theories
   - Detection of inconsistencies
   - Well-Founded Reasoning

Why Peer-to-Peer ?



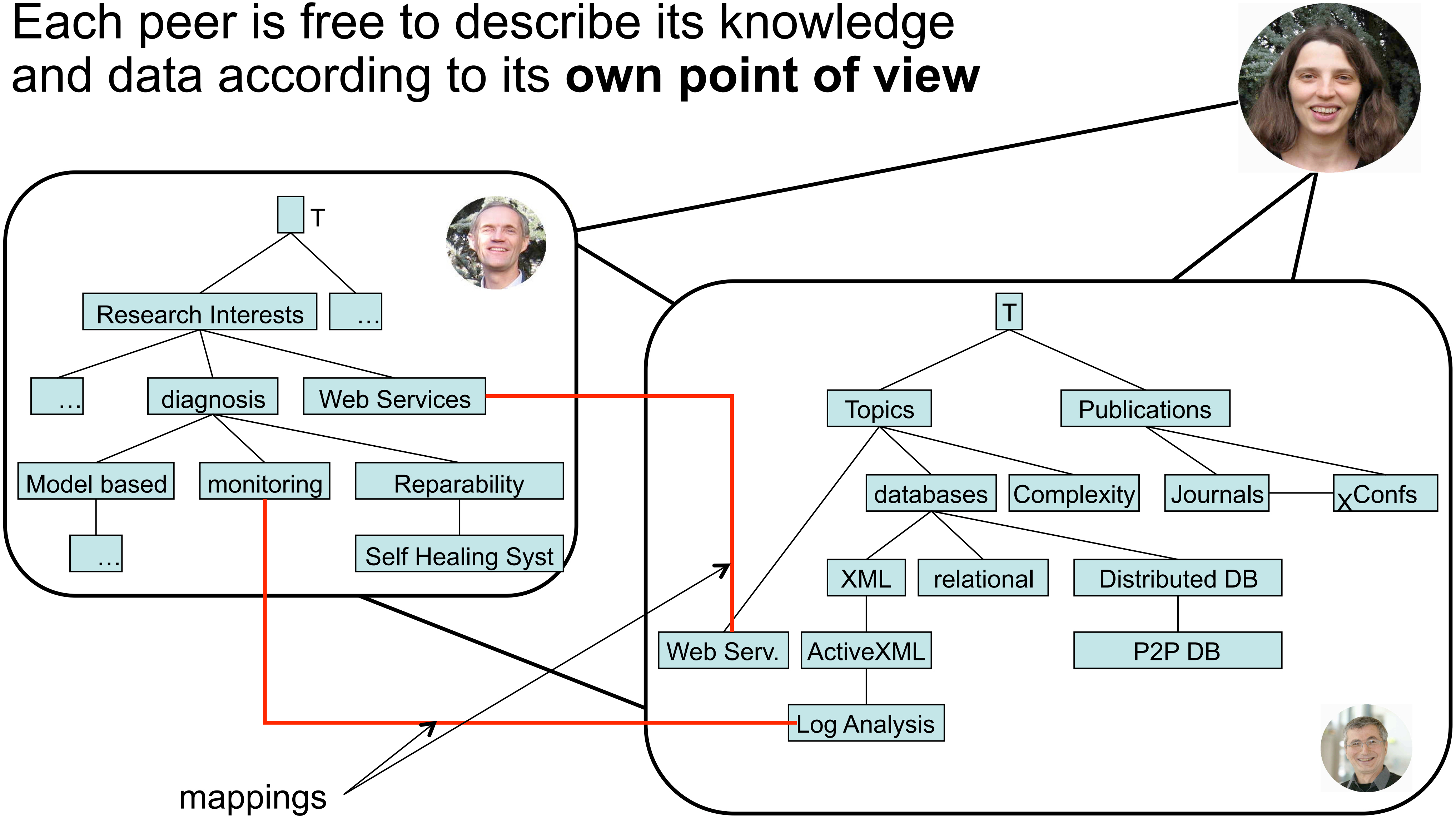
Characteristics of P2P Systems

- Networks of **independent** peers
- Each peer can be a provider and a consumer (client / server)
- Scaling up
- Dynamic architectures
- Robustness

Goal : Explore how to benefit from such nice properties in the context of semantic information integration systems

Peer Data Management Systems

Each peer is free to describe its knowledge and data according to its **own point of view**



Reasoning in a PDMS

Answering queries in a PDMS

2 steps approach

- query reformulation

  computes maximal rewritings of the query

  i.e. most general conjunctive queries that entails the initial query wrt the global theory

- evaluation of the reformulations against data

P2P Inference Systems

- **SomeRDFS**
- **SomeOWL**

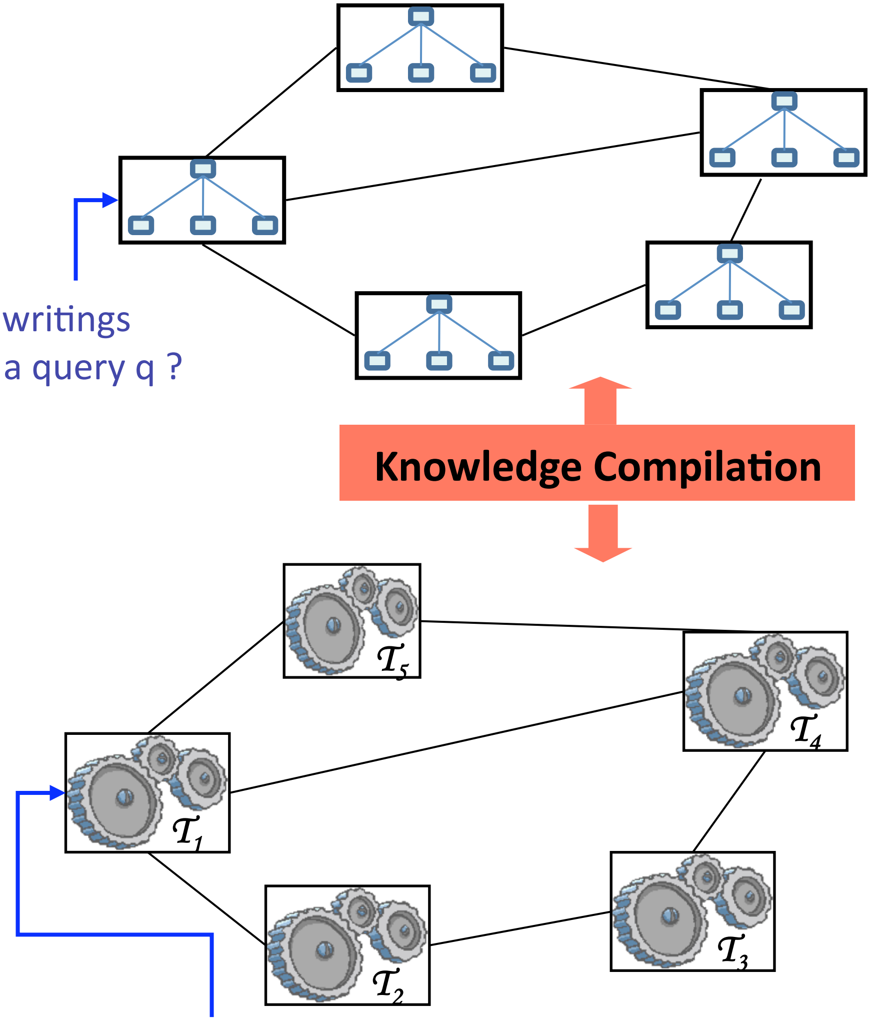
Propositional encoding

SomeWhere

Peers are autonomous agents

- own language
- local theory + mappings
- inference engine
- communication capabilities with their neighbors

Reduction to a problem of
distributed consequence finding



Rewritings
of a query q ?

Knowledge Compilation

Proper prime consequences of
c w.r.t. the global theory ?

$$\mathcal{T} = \bigcup \mathcal{T}_i$$

Propositional ?

- Propositional reasoning is already difficult
- Simplicity of the language allows for efficient algorithms implementation
- Tremendous improvements of SAT solvers over the previous decades
- Consequence Finding algorithms may benefit from some of these techniques
- The encoding may be hidden to the user

SomeWhere is a general architecture that can be used
in a much wider range of applications than
information integration

The challenge

No peer knows the global theory !

A peer only knows

- its local theory
- mappings with its direct neighbors

Many interesting inference problems have to be reconsidered in this fully distributed setting

- Can we obtain in a fully decentralized way the very same answers that would be obtained in the centralized case ?
- Does the solution scale up ?

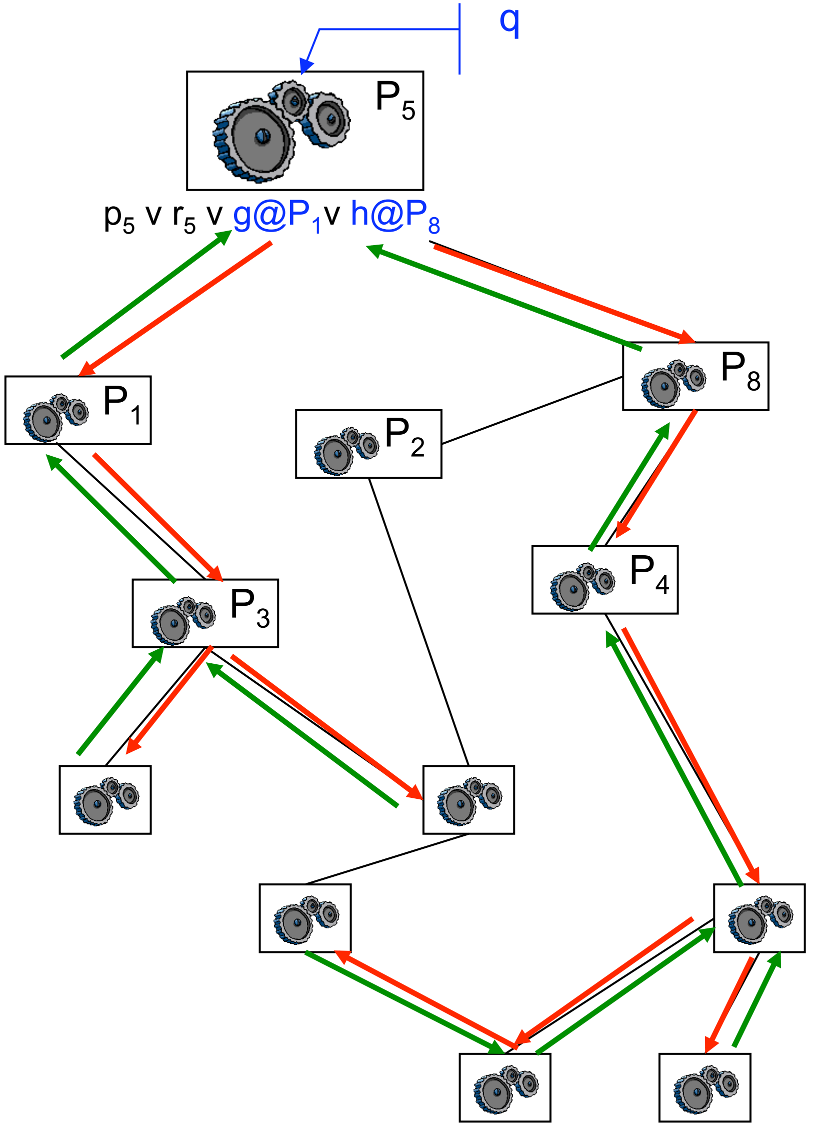
For consequence finding ?

DeCA [Adjiman & al 06]

- runs on all peers
- query a peer with its language
- computes consequences of an input clause *c*

A two step algorithm

1. Computes local consequents wrt some production field.
   Pure local consequents are returned immediately
2. Split/Recombination strategy
   - clauses involving *foreign variables* are splitted
   - Implicates of foreign litterals are recursively computed by neighbors peers
   - respective results are recombined incrementally



DeCA

Properties

- Anytime
- Termination notification
- Correctness
- Completeness

  all proper prime implicates of a clause wrt the global theory
  are returned

DeCA has been implemented in the **SomeWhere Plateform**

Experimental evaluation

- "small world" networks of 1000 peers
- random 2+k clause theories (crossover sat/unsat)
  70 variables

⇒ **Somewhere scales up** [IJCAI'05,JAIR'06]

Outline

1. Introduction
   - Peer Data Management Systems
   - The SomeWhere plateform
2. Reasoning with inconsistent peers theories
   - Detection of inconsistencies
   - Well-Founded Reasoning

In case of inconsistency ?

Problem 1

Trivialization renders DECA answers meaningless !

Problem 2

Inconsistencies **cannot be avoided** because the framework is decentralized

All peers being equal… no culprit !

New challenges :

- Can we detect the presence of inconsistencies ?
- Can we restrict the reasoning to produce only meaningful answers ?

Detection of inconsistency

A reasonable hypothesis

We assume each T_i to be consistent

Peer theories $T_i = L_i \cup M_i$

- L_i : the "local" part of T_i
- M_i : the "mapping" part of T_i

Thus $\boldsymbol{L} = \cup L_i$ is consistent

Mappings are responsible for inconsistencies

Def

A **nogood** is a subset ***ng*** of $\boldsymbol{M} = \cup M_i$ such that ***L U ng*** is inconsistent

Detection of inconsistency

It can be done in a fully decentralized way

the empty clause should be derivable
from any clause of a nogood

When a new mapping *m* is added
look for proofs of the empty clause

- with *m* as input clause
- *ms is* the **mapping support** of the proof

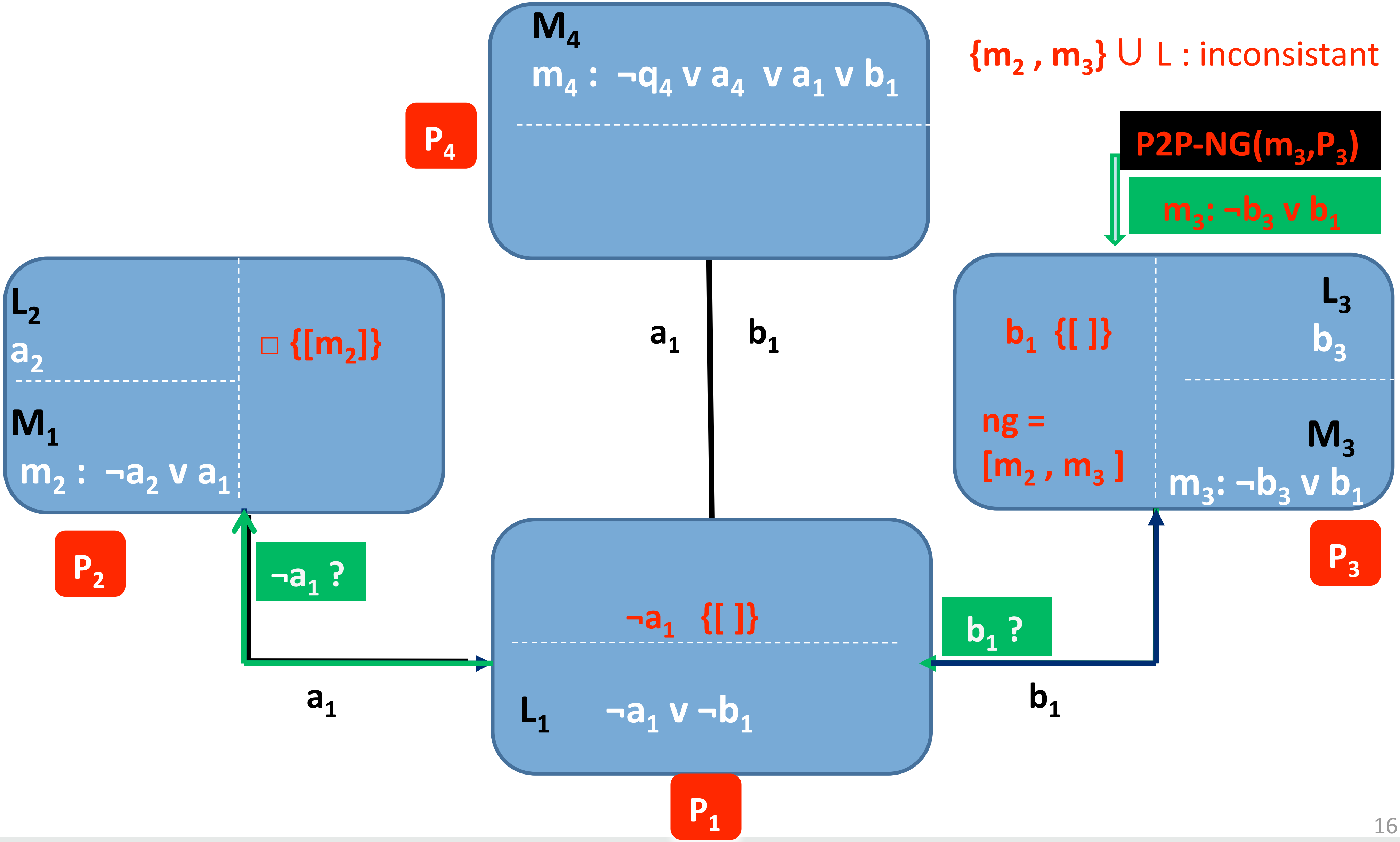
ms U {*m*} is a nogood !

The P2P-NG algorithm

P2P-NG(m,P) is a both a specialized and extended version of DeCA

- used for each addition of a mapping *m* to a peer *P*
- keeps track of mapping supports
- termination conditions are different
- looks for all possible empty clause proofs
- focuses on derivation of the empty clause

P2P-NG illustration



$\{m_2, m_3\} \cup L$: inconsistant

P2P-NG(m_3,P_3)

m_3: $\neg b_3 \vee b_1$

M_4

m_4 : $\neg q_4 \vee a_4 \vee a_1 \vee b_1$

P_4

L_2

a_2

□ {[m_2]}

M_1

m_2 : $\neg a_2 \vee a_1$

P_2

$\neg a_1$?

a_1

a_1 b_1

b_1 {[ ]}

L_3

b_3

ng = [m_2 , m_3]

M_3

m_3: $\neg b_3 \vee b_1$

P_3

b_1 ?

b_1

$\neg a_1$ {[ ]}

L_1 $\neg a_1 \vee \neg b_1$

P_1

Nogoods recording

Properties of P2P-NG

- correct
- complete

  all **minimal nogoods** are discovered
  whatever the order of mapping introduction
- nogoods are stored in a **distributed way**

  A nogood found by P2P-NG(m,P) is recorded
  by the peer P
- Peers having mappings involved in nogood are not necessarily aware of it

Well Founded consequence finding

Def *a **well founded consequent** of c wrt the (global) theory T is a consequent of c wrt a consistent subset of T*

Intuition

consequents all mapping support of which contain some nogood should be discarded

Problem

all nogoods have been detected but no peer knows where these are stored

WF-DeCA

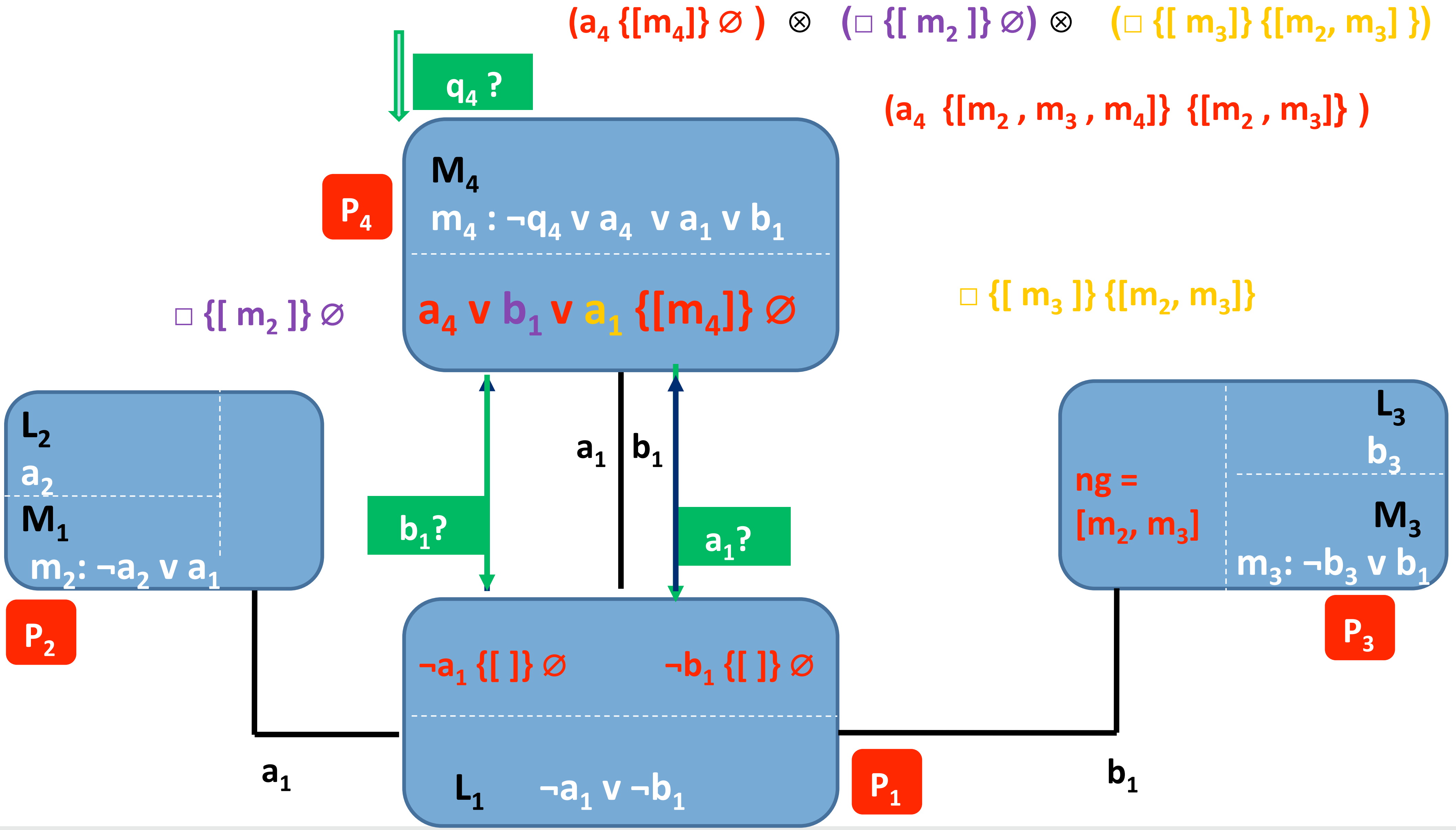
An extension of DeCA that

- computes consequents of some input clause
- keeps track of mapping supports
- collects **relevant** nogoods on visited peers
- filter out consequents that are not well founded

relevant nogoods

a nogood ng is relevant to a consequent with a set of mapping supports sms = $\{ms_1,\ldots,ms_K\}$ if it contains at least a mapping of $ms_1,\ldots,ms_K$

WF-Deca : illustration



$(a_4 \{[m_4]\} \varnothing) \otimes (\square \{[m_2]\} \varnothing) \otimes (\square \{[m_3]\} \{[m_2, m_3]\})$

$(a_4 \ \{[m_2, m_3, m_4]\} \ \{[m_2, m_3]\})$

q_4 ?

P_4

M_4

$m_4 : \neg q_4 \vee a_4 \vee a_1 \vee b_1$

$a_4 \vee b_1 \vee a_1 \{[m_4]\} \varnothing$

$\square \{[m_2]\} \varnothing$

$\square \{[m_3]\} \{[m_2, m_3]\}$

L_2

a_2

M_1

$m_2: \neg a_2 \vee a_1$

P_2

b_1?

a_1

b_1

a_1?

ng = $[m_2, m_3]$

L_3

b_3

M_3

$m_3: \neg b_3 \vee b_1$

P_3

$\neg a_1 \{[\ ]\} \varnothing$

$\neg b_1 \{[\ ]\} \varnothing$

L_1 $\neg a_1 \vee \neg b_1$

P_1

a_1

b_1

WF-DeCA Properties

Properties

- anytime
- termination notification
- guarantee that **all** relevant nogoods are collected
- correctness relies on the completeness of P2P-NG

⇒ **SomeWhere+** [experimental evaluation]

Summary

Fully decentralized reasoning in a P2P setting is possible and scales up

• Deca	consistent network
• WF-Deca	with inconsistencies

Outcome of projects Picsel3, Mediad [FT R&D]

- SomeWhere [regist. softw.] - used at Univ. Toronto
- SomeRDFS [WebContent]

Perspectives

- dynamicity of P2PIS
- rewriting evaluation strategies
  - preferences
  - cost model

Related work :

- Trust model inconsistent answers
- conservative extension
- conditional mappings / alternative semantics