Distributed Reasoning with Conflicts in an Ambient Peer-to-Peer Setting

Antonis Bikakis, Grigoris Antoniou

Institute of Computer Science, FORTH, Greece

Artificial Intelligence Methods for Ambient Intelligence

Aim of this Study

- An algorithm for reasoning with distributed rule theories in an ambient setting
- The algorithm
  - models the participating agents as nodes in a P2P system
  - takes into account some special characteristics of context knowledge and ambient agents
  - considers the potential conflicts that may arise during the integration of the distributed knowledge

Talk Outline

- Context Knowledge and Ambient Agents
  - Notion of Context
  - Special Characteristics of Ambient Agents
  - Challenges of Reasoning in an Ambient Setting
- Related Work
- Algorithm Description
  - General Approach
  - Problem Statement - Definitions
  - Steps of the *P2P_DR* Algorithm
  - The Algorithm in Action
  - Algorithm Properties
- Future Work

Notion of Context

- Context can be described as

".. any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and application themselves.." [Dey and Abowd, 1999]

- Context Characteristics
  - Multiple heterogeneous formats
  - Dynamic
  - Unknown
  - Ambiguous
  - Imprecise
  - Erroneous

Ambient Agents

- Diverse goals, experiences and perceptive capabilities
- Distinct vocabularies
- Different levels of sociality
- Dynamic behavior
  - Nodes join and leave the system randomly
    - →An ambient agent is not able to know a priori all other entities that are present at a specific time instance
    - →It cannot communicate directly with all other ambient agents

Challenges

- Reasoning with the highly dynamic and ambiguous context
- Managing the potentially huge piece of context data, in a real-time fashion, taking into account the restricted computational, storage and communication capabilities of some mobile devices
- Collective intelligence, by supporting information sharing, and distributed reasoning between the entities of the ambient environment.

Centralized reasoning is not a good solution, as it is too costly and cannot handle the system dynamics

Talk Outline

- Context Knowledge and Ambient Agents
  - Notion of Context
  - Special Characteristics of Ambient Agents
  - Challenges of Reasoning in an Ambient Setting
- Related Work
- Algorithm Description
  - General Approach
  - Problem Statement - Definitions
  - Steps of the *P2P_DR* Algorithm
  - The Algorithm in Action
  - Algorithm Properties
- Future Work

Reasoning in PDMS (1/2)

- First Order Logic Interpretations of P2Ps
  - proposed by [*Bernstein et al.,2002; Halevy et al.,2003*]
  - problems regarding modularity, generality and decidability
- Semantics based on Epistemic Logic
  - proposed by [*Calvanese et al.,2004*]
  - does not deal with inconsistencies
- Autoepistemic Semantics
  - proposed by [*Franconi et al.,2003*]
  - formalizes local inconsistency
  - guarantees that a locally inconsistent database base will not render the entire knowledge base inconsistent

Reasoning in PDMS (2/2)

- Non-monotonic Epistemic Logic Semantics
  - proposed by [*Calvanese et al.,2005*]
  - enables isolating local inconsistency
  - handles peers that provide mutually inconsistent data
- Propositional P2P Inference System
  - proposed by [*Chatalic et al.,2006*]
  - detects mutually inconsistent data and reasons without them
- Common Deficiencies
  - Conflicts are not actually resolved; they are rather isolated
  - Trust is not part of the model
  - In most cases, the participating peers share a common alphabet

Non-monotonic Reasoning in MCS

- Rule-based MCS with Default Negation
  - proposed by [*Roelofsen & Serafini, 2005*]
- Contextual Default Reasoning
  - proposed by [*Brewka et al.,2007*]
  - models bridging rules between different contexts as default rules
  - closer to implementation due to the well-studied relation between Default Logic and Logic Programming
  - does not provide reasoning algorithms, leaving some practical issues unanswered

Talk Outline

- Context Knowledge and Ambient Agents
  - Notion of Context
  - Special Characteristics of Ambient Agents
  - Challenges of Reasoning in an Ambient Setting
- Related Work
- Algorithm Description
  - General Approach
  - Problem Statement - Definitions
  - Steps of the *P2P_DR* Algorithm
  - The Algorithm in Action
  - Algorithm Properties
- Future Work

Our Approach

- A P2P rule-based model that captures
  - local knowledge
  - bridging rules
  - trust
- Why P2P?
  - Each different peer independently collects and processes in its own way the available context information.
  - Each peer may not have (immediate) access to all information sources.
  - The peers share their knowledge through messages with their neighboring nodes.
  - Each peer may not trust all the other peers at the same level.
  - Peers join and leave the system randomly.

Definitions

- We assume a peer-to-peer system P as a collection of peer local theories

  $P=\{P_i\}$, i=1,2,…,n

- Each peer has a proper distinct vocabulary V_{Pi} and a unique identifier i.
- Each local theory is a set of rules that contain only local literals

  $r_i : a_i , b_i ,…, k_i \rightarrow x_i$

- Each peer also defines mappings that associate local literals with literals from the vocabulary of other peers (remote literals):

  $m_i : a_i , b_j ,…, z_k \rightarrow x$

Problem Statement

- ***Given a peer-to-peer system P and a query about a literal x_i issued at peer P_i, find the truth value of x_i considering P_i's local theory, its mappings and the theories of other system nodes.***
  - We assume that the local theories are consistent,
  - ..but this is not necessarily true for the unification of the system peer theories (local rules and mappings).
  - The inconsistencies result from interactions between local theories and are caused by mappings.

P_1	P_2	P_3
$r_{11} : a_1 \rightarrow x_1$	$r_{21} : \rightarrow a_2$	$r_{31} :\rightarrow a_3$
$m_{11} : a_2 \rightarrow a_1$		
$m_{12} : a_3 \rightarrow \neg a_1$		

Steps of the *P2P_DR* Algorithm (1/3)

- **Step 1**
  - Use P_i's local theory to prove x_i
- **Step 2**
  - Collect P_i's local and mapping rules that support x_i .
  - For each such rule, check the truth value of the literals in its body. For each local / remote literal, issue similar queries (*recursive calls of the algorithm*) to P_i (local literals) or to the appropriate P_i 's acquaintances (remote literals).
  - To avoid *circles*, before each new call, check if the same query has been issued before during the same call of the algorithm.
  - Build the *supportive set of x_i* ; this contains the 'strongest' set of mapping rules (defined either locally or remotely) that can be used to prove x_i in the absence of contradictions.

Steps of the *P2P_DR* Algorithm (2/3)

- **Step 3**
  - Collect P_i's local and mapping rules that support $\neg x_i$ (contradict x_i).
  - In the same way with Step 2, build the *supportive set of* $\neg x_i$ (*conflicting set of x_i*)
- **Step 4**
  - Compare the *supportive* with the *conflicting* set of x_i .
  - If the *supportive set* is stronger set than the *conflicting set*, return *Yes* and terminate. Otherwise, return *No* and terminate.

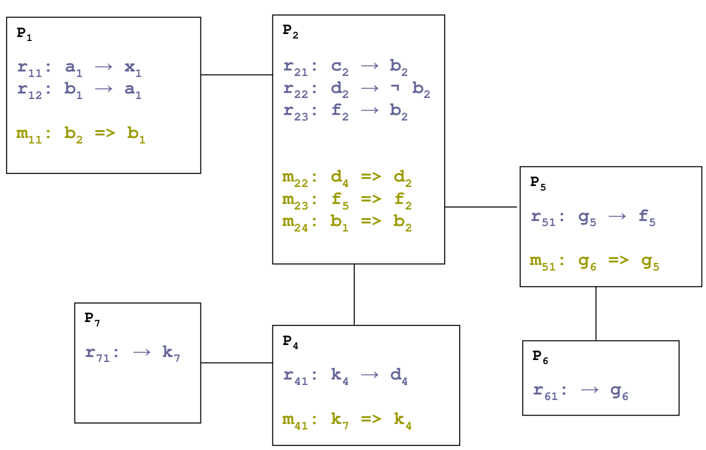
Steps of the *P2P_DR* Algorithm (3/3)

- How to compare two mapping sets
  - Each peer defines an *order* of the system peers, based on the *trust* it has on each one of them. According to this ordering, for two mapping rules, m_k and m_l

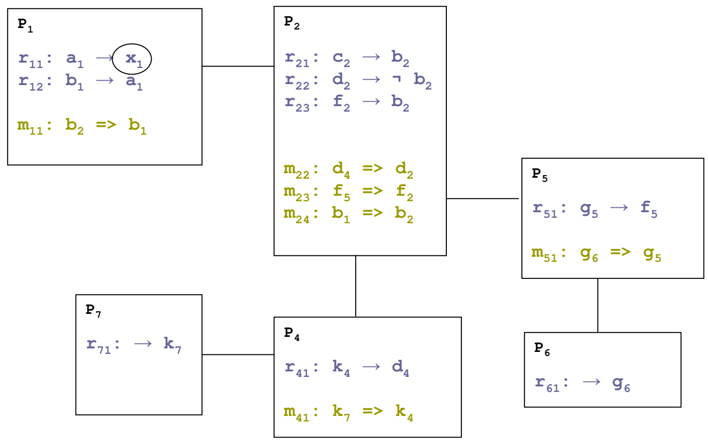
    m_k is stronger than m_l from P_i's viewpoint if P_k precedes P_l in P_i's order

  - The strength of a mapping set is determined by the strength of the weakest rule in this set.

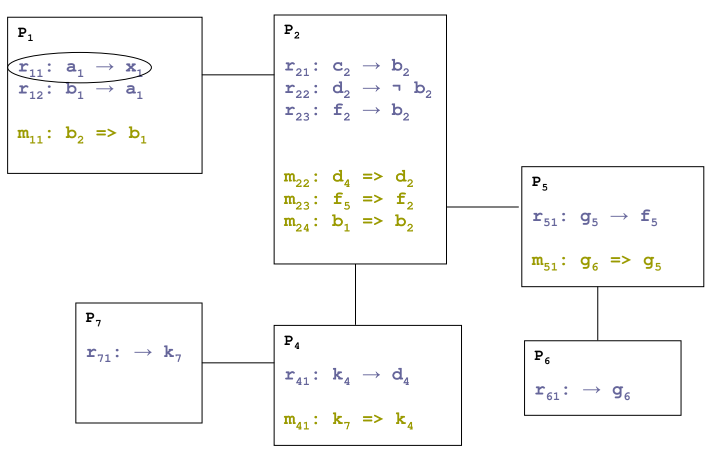
Assume that we issue a query about x_1 to P_1



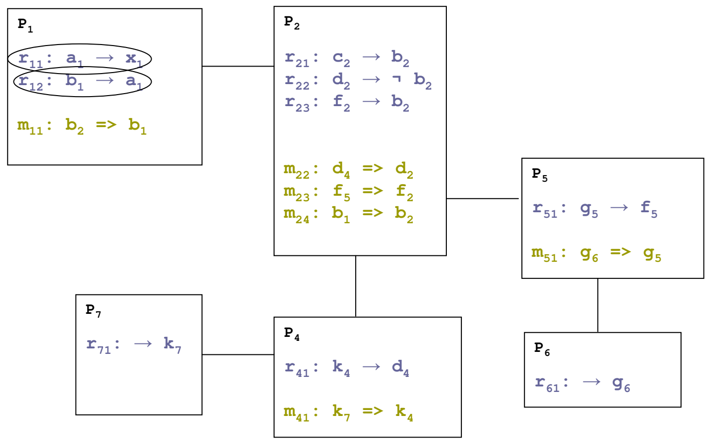
Neither x_1 nor $\neg x_1$ derive from P_1's local theory



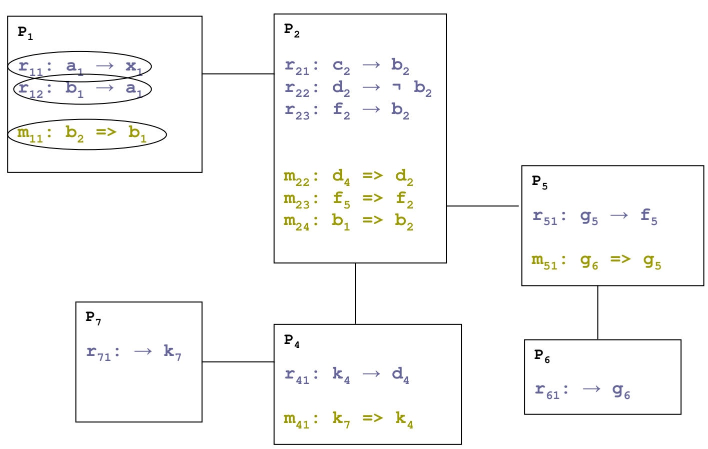
r_{11} is a supportive rule for x_1, which has a_1 as its only premise.



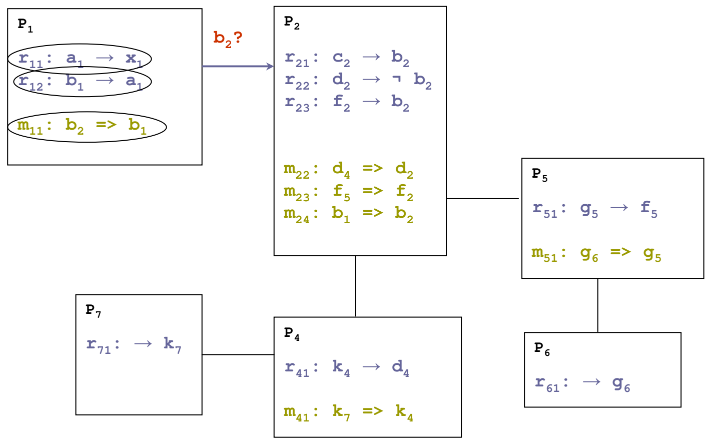
r_{12} is a supportive rule for a_1, which has b_1 as its only premise.



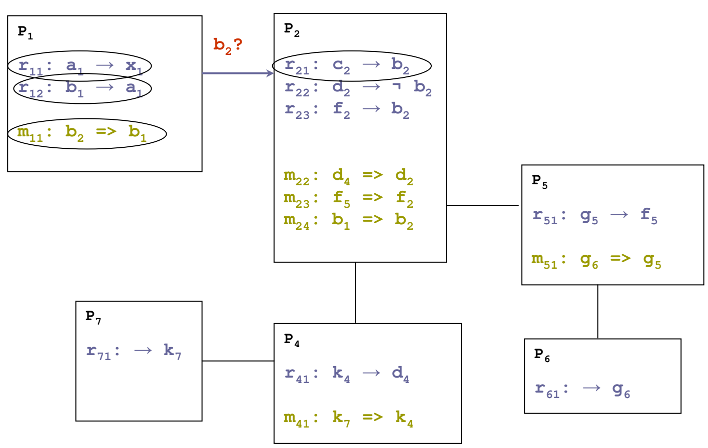
m_{11} is a supportive rule for b_1, which has b_2 as its only premise



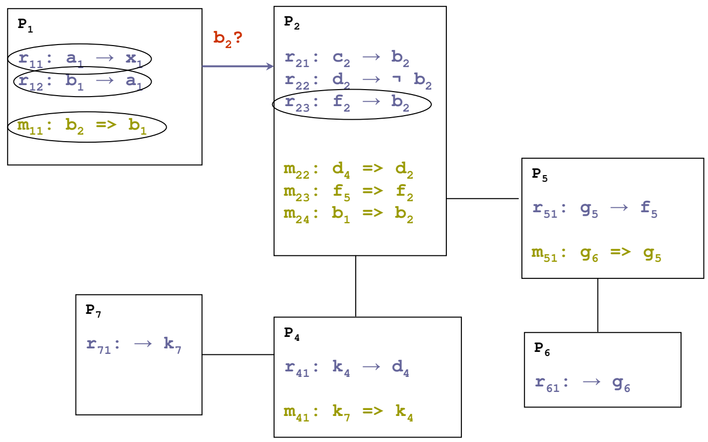
b_2 belongs to P_2's published theory, so P_1 queries P_2 about b_2



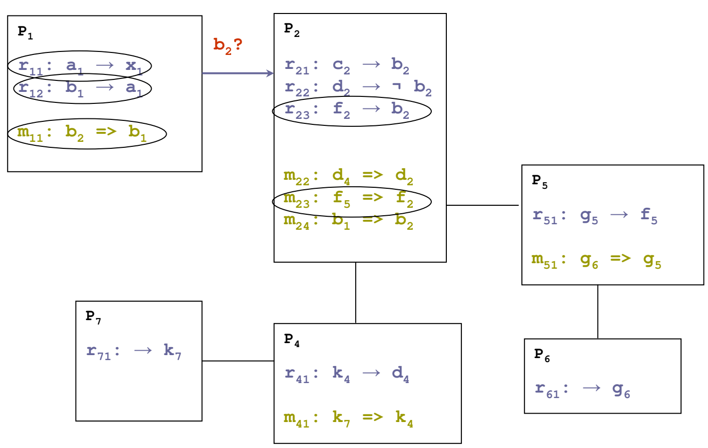
r_{21} is a supportive rule for b_2, which has c_2 as its only premise. There is no supportive rule for c_2, so r_{21} cannot be used to prove b_2



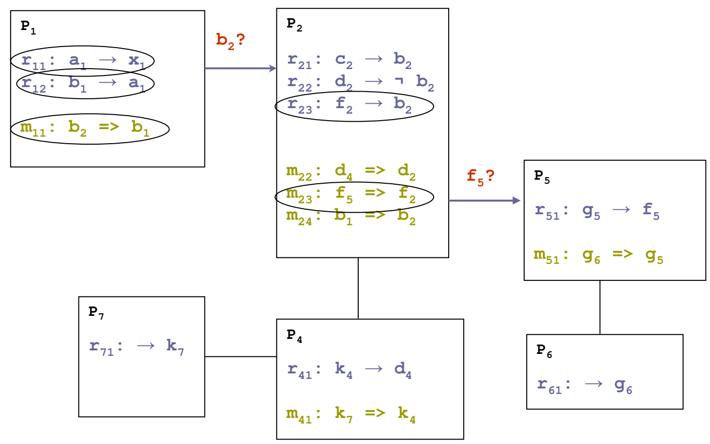
r_{23} *is another supportive rule for* b_2*, which has* f_2 *as its only premise.*



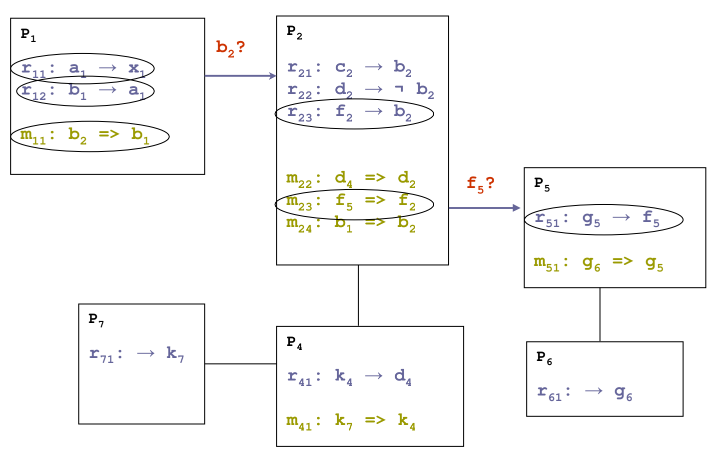
m_{23} *is a supportive rule for* f_2*, which has* f_5 *as its only premise.*



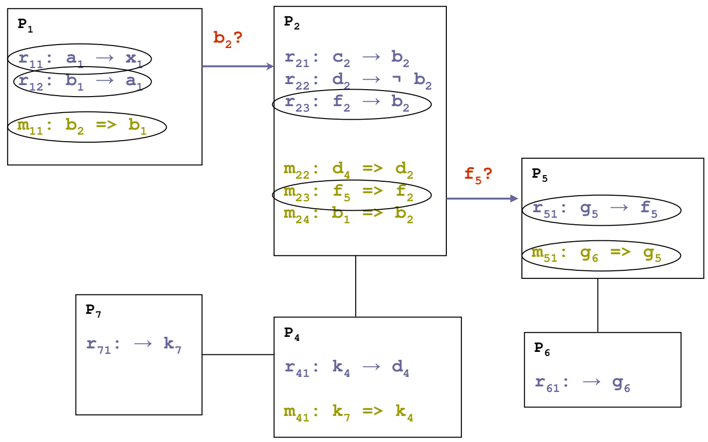
f_5 belongs to P_5's published theory, so P_2 queries P_5 about f_5



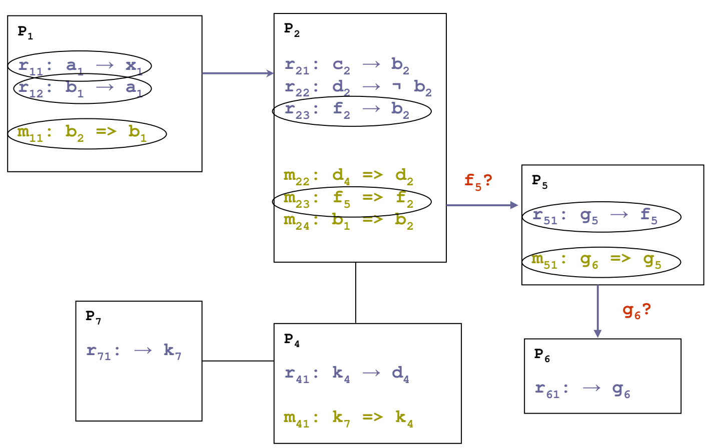
r_{51} is a supportive rule for f_5, which has g_5 as its only premise.



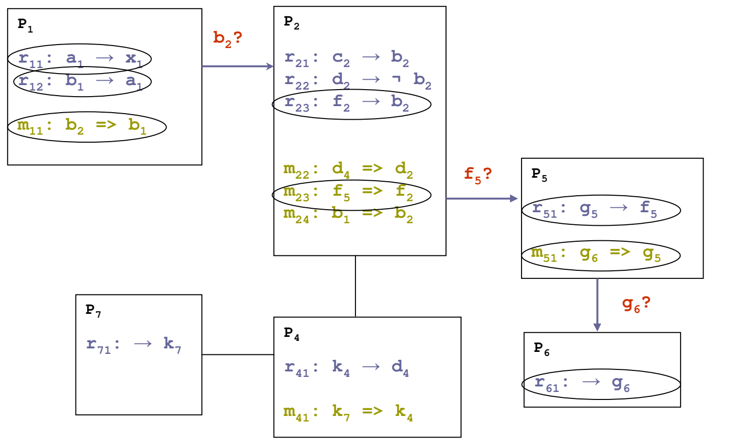
m_{51} is a supportive rule for g_5, which has g_6 as its only premise.



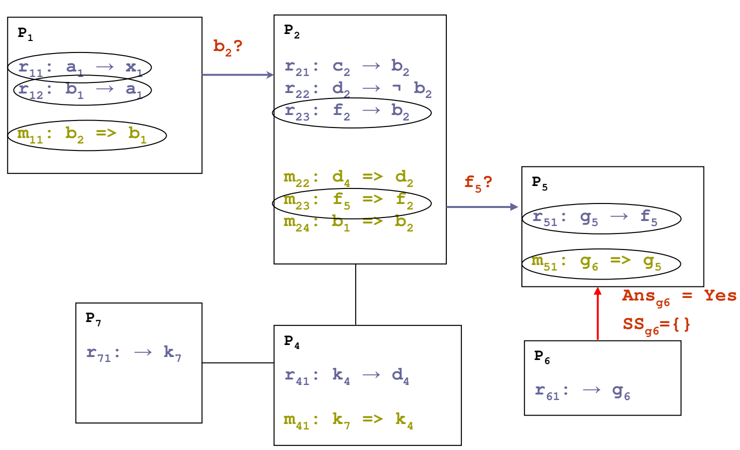
g_6 belongs to P_6's published theory, so P_5 queries P_6 about g_6



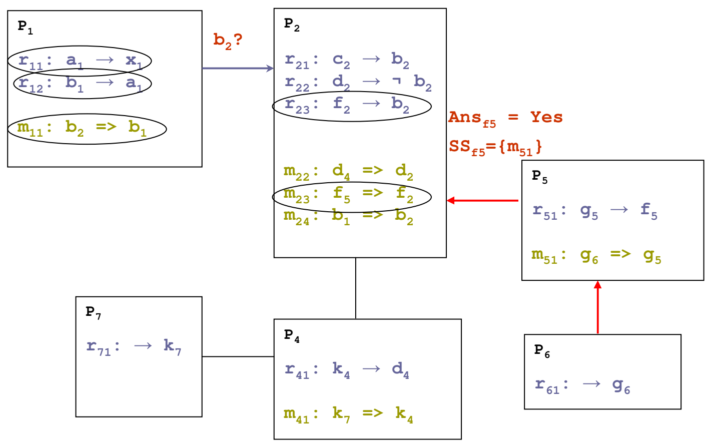
g_6 derives from P_6's local theory



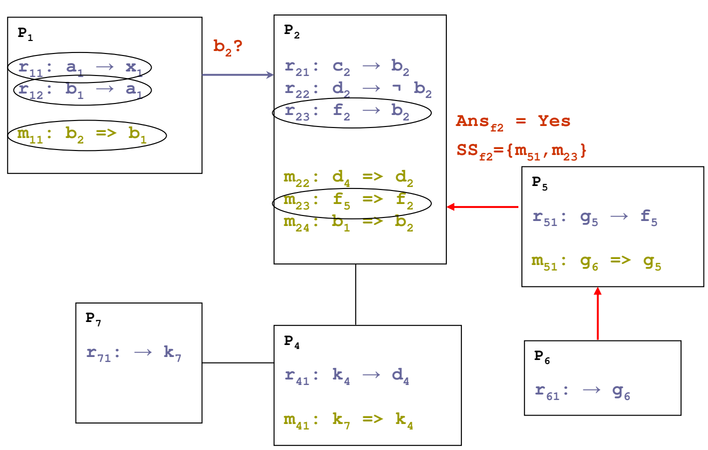
*P_6 returns the answer **Yes** to P_5 about g_6 with an empty set of supportive mappings (it was proved locally)*



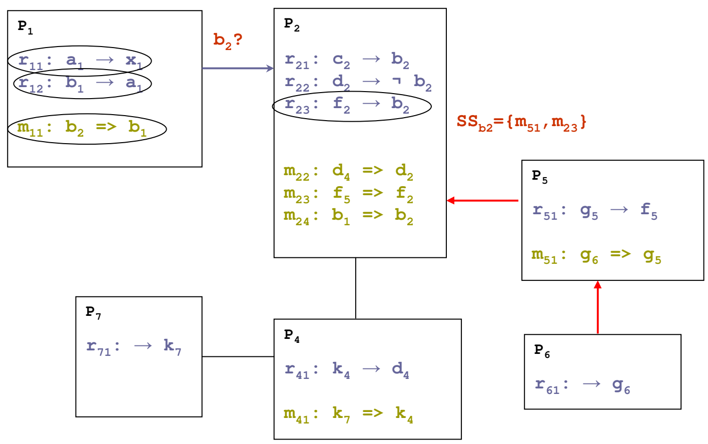
*P_5 returns the answer **Yes** to P_2 about f_5 with a supportive set $SS_{g5}=\{m_{51}\}$*



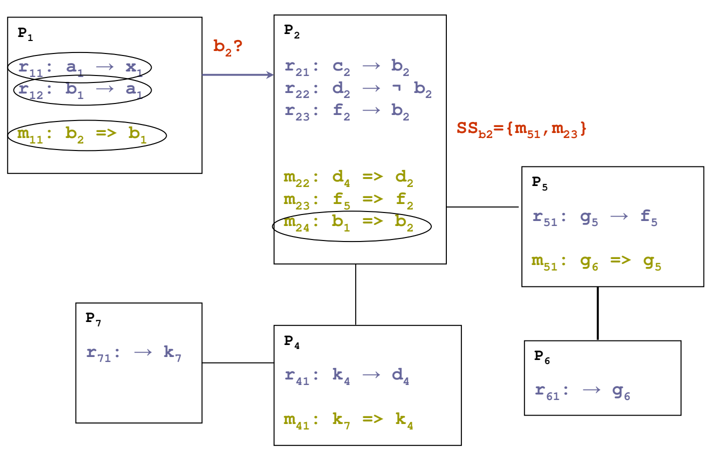
*P_2 returns the answer **Yes** for f_2 with a supportive set $SS_{f2}=\{m_{51},m_{23}\}$*



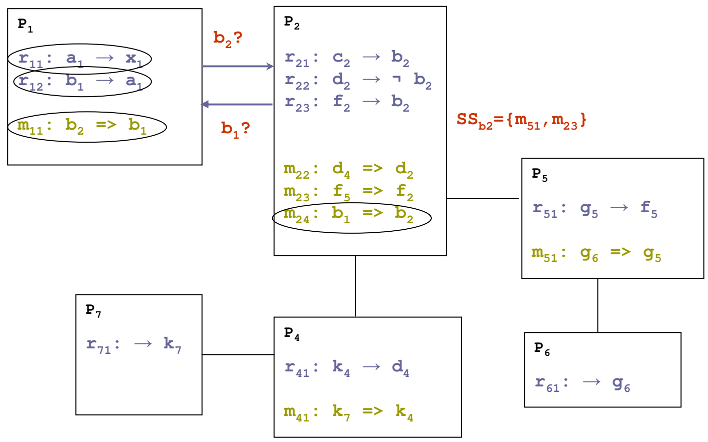
P_2 builds an initial supportive set for b_2, $SS_{b2}=\{m_{51},m_{23}\}$



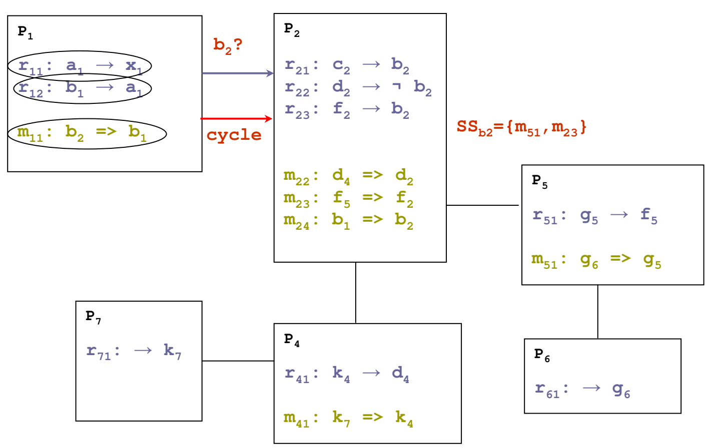
m_{24} is another supportive rule for b_2, which has b_1 as its only premise.



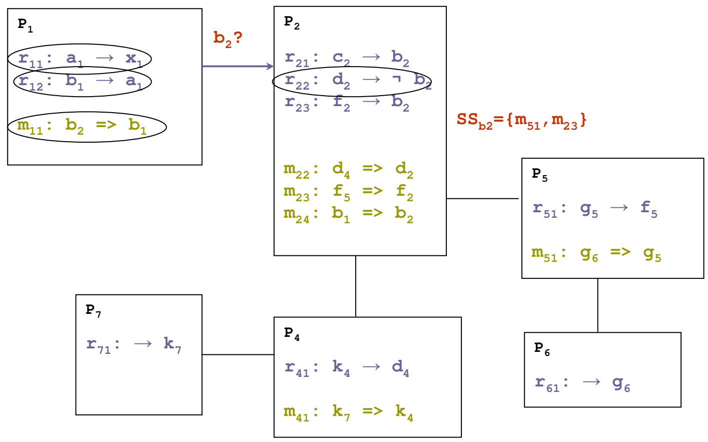
b_1 belongs to P_1's published theory, so P_2 queries P_1 about b_1



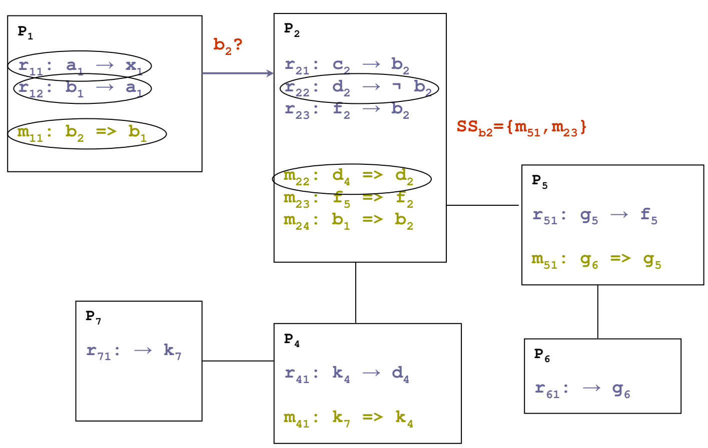
A cycle is detected, so P_2 abandons m_{24}



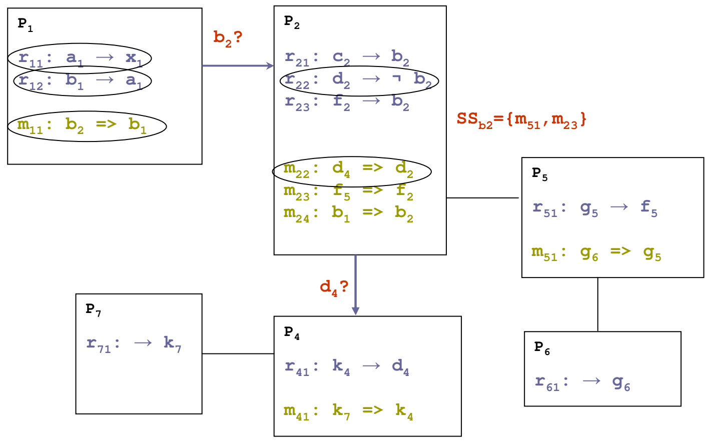
r_{22} *is a conflicting rule for* b_2*, which has* d_2 *as its only premise*



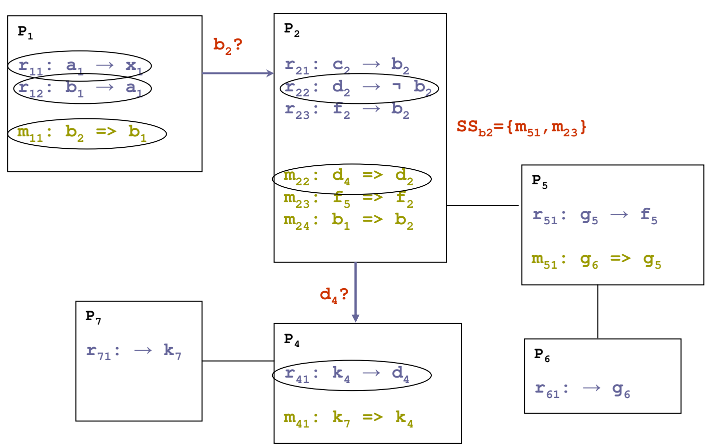
m_{22} *is a supportive rule for* d_2*, which has* d_4 *as its only premise.*



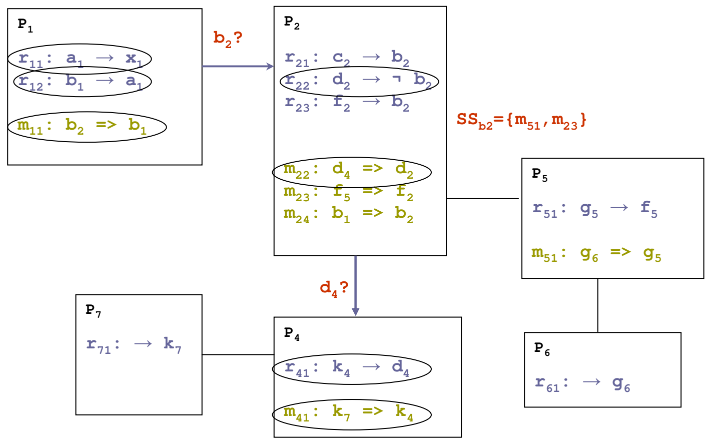
d_4 belongs to P_4's published theory, so P_5 queries P_4 about d_4



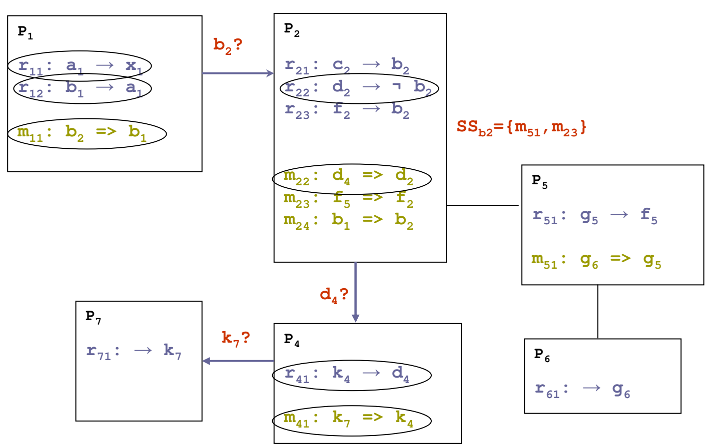
r_{41} is a supportive rule for d_4, which has k_4 as its only premise



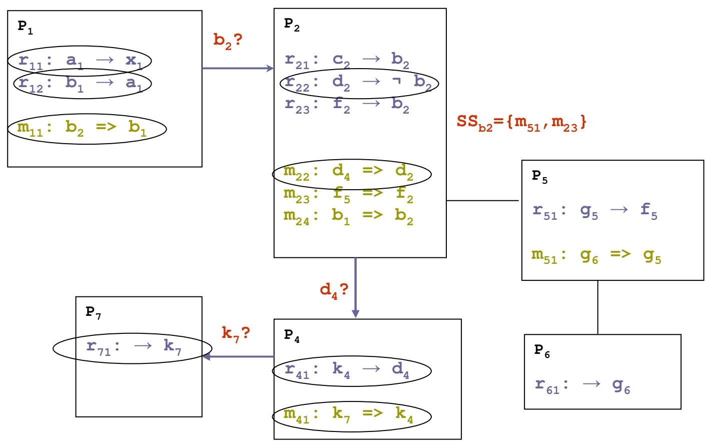
m_{41} is a supportive rule for k_4, which has k_7 as its only premise



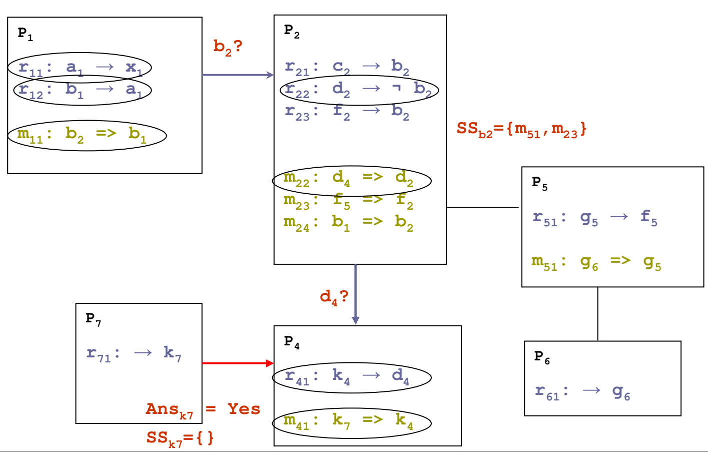
k_7 belongs to P_7's published theory, so P_4 queries P_7 about k_7



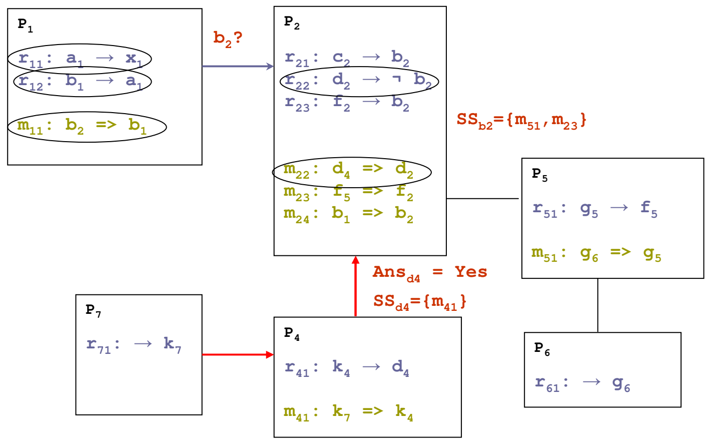
k_7 derives from P_7's local theory



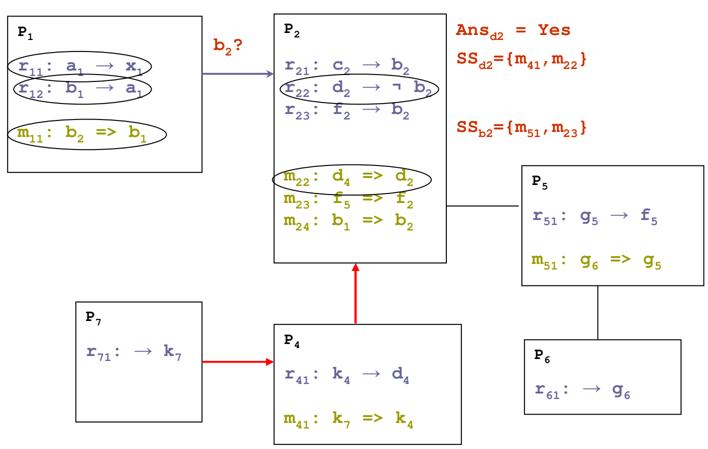
*P_7 returns the answer **Yes** to P_4 about k_7 with an empty set of supportive mappings (it was proved locally)*



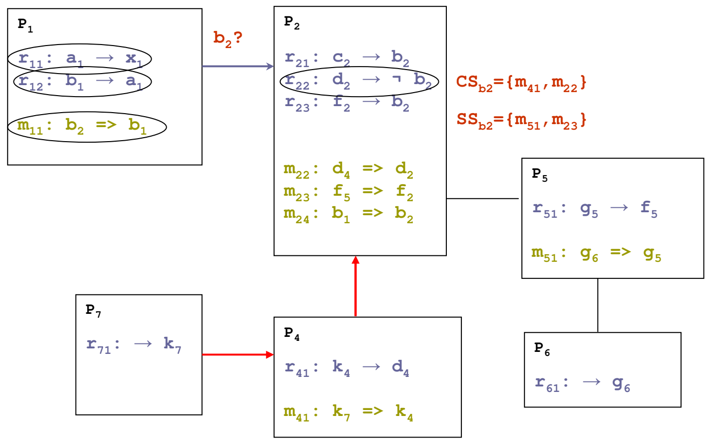
*P_4 returns the answer **Yes** to P_2 about d_4 with a supportive set*



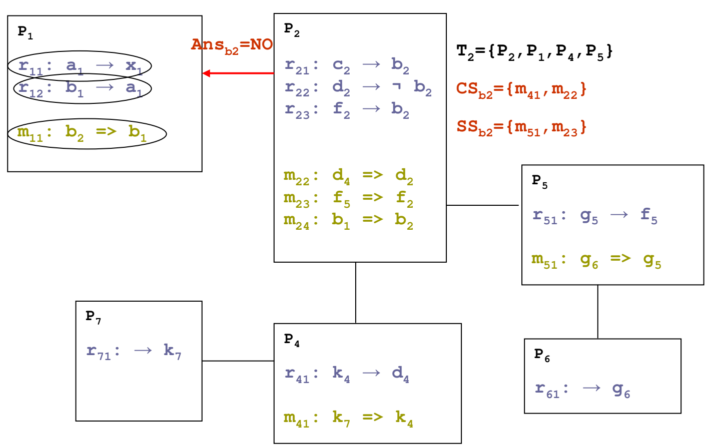
*P_2 returns the answer **Yes** for d_2 with a supportive set $SS_{d2}=\{m_{41},m_{22}\}$*



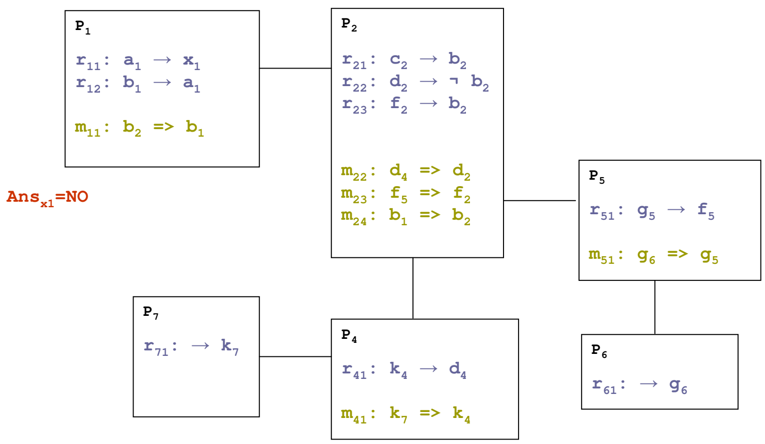
P_2 builds the conflicting set of b_2, $CS_{b2}=\{m_{41},m_{22}\}$



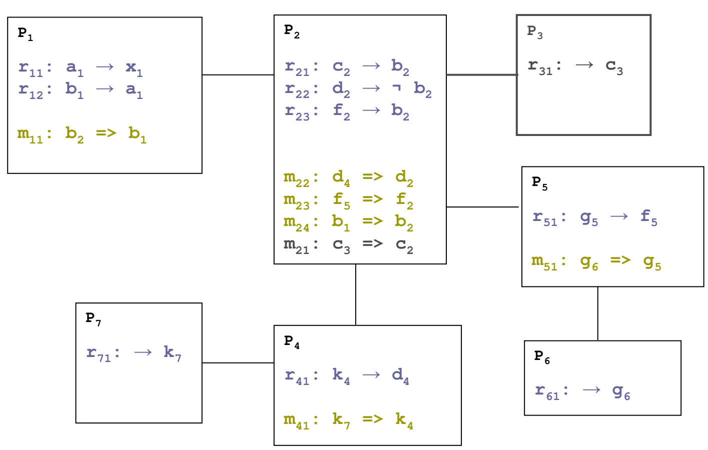
*Assuming that P_2 trusts P_4 more than P_5, SS_{b2} is not stronger than CS_{b2} so P_2 cannot prove b_2 and returns **NO** to the query issued by P_1*



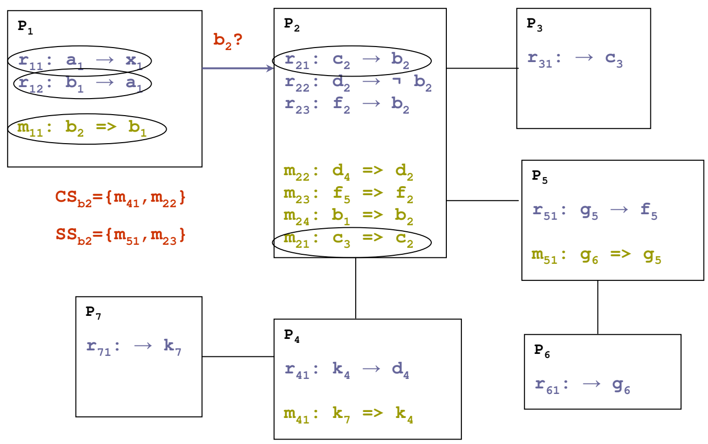
*P_1 returns successively **NO** for b_1, a_1, and finally for x_1*



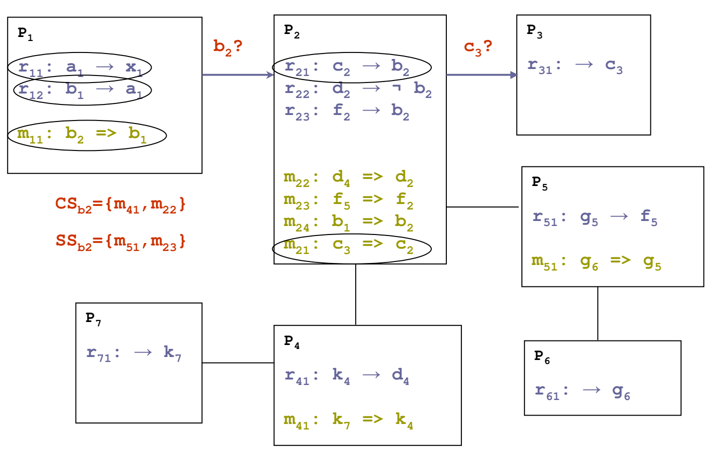
Imagine the case that a new peer (P_3) joins the system, and P_2 establishes a connection with the new peer through the mapping rule m_{21}



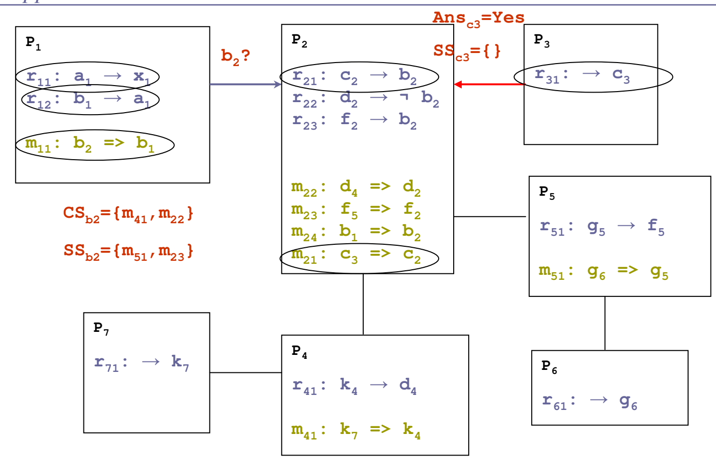
m_{21} supports c_2, so rule r_{21} is now applicable



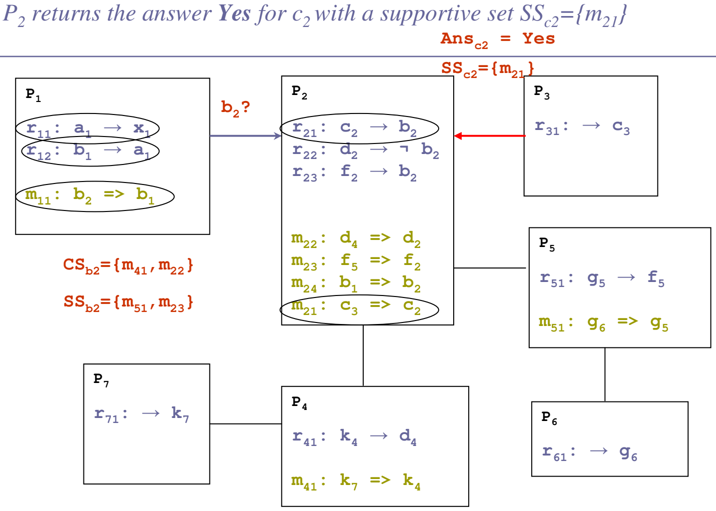
m_{21} has c_3 as its only premise, and P_2 issues a query about c_3 to P_3



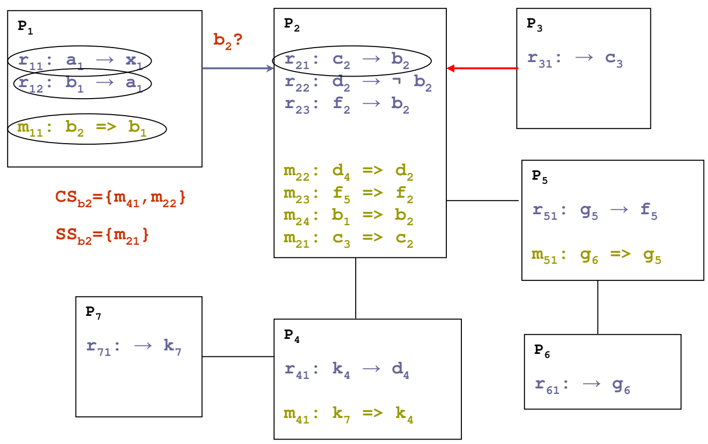
*c_3 derives from P_3's local theory , and P_3 returns Ans_{c3} = **Yes** with an empty supportive set*



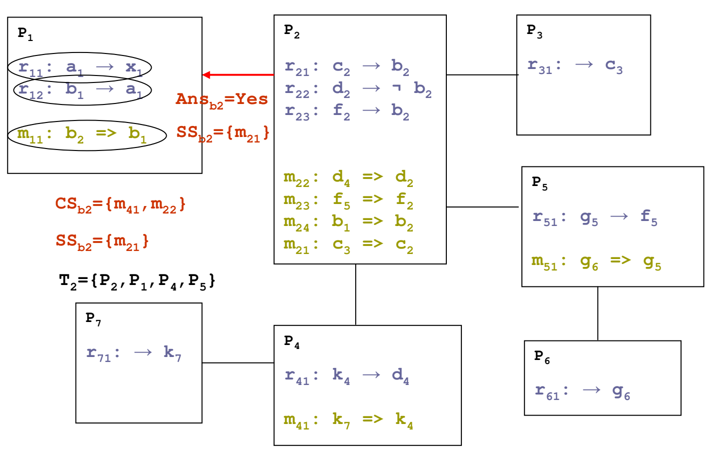
*P_2 returns the answer **Yes** for c_2 with a supportive set SS_{c2}={m_{21}}*



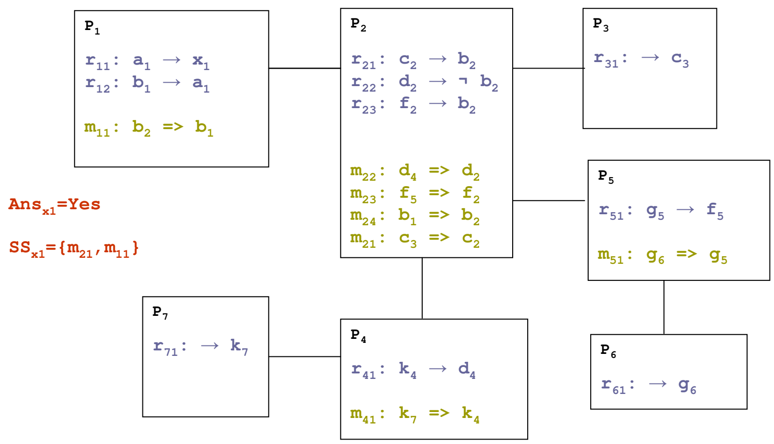
m_{21} is stronger than $\{m_{41}, m_{22}\}$, so $SS_{b2} = \{m_{21}\}$



*Now SS_{b2} is stronger than CS_{b2} as m_{21} is stronger than $\{m_{41}, m_{22}\}$, and P_2 returns Ans_{b2} = **Yes**, with $SS_{b2} = \{m_{21}\}$*



*P_1 returns successively **Yes** for b_1, a_1, and finally for x_1, adding SS_{b2} and m_{11} to SS_{x1}*



Properties of the Algorithm

- The algorithm is guaranteed to *terminate*
- The total *number of messages* that need to be exchanged for the evaluation of a single query is in the worst case $O(n^2)$ (n is the total number of system nodes)
- There is a *defeasible theory* that derives from the unification of the distributed theories and derives the same conclusions.

Talk Outline

- Context Knowledge and Ambient Agents
  - Notion of Context
  - Special Characteristics of Ambient Agents
  - Challenges of Reasoning in an Ambient Setting
- Related Work
- Algorithm Description
  - General Approach
  - Problem Statement - Definitions
  - Steps of the *P2P_DR* Algorithm
  - The Algorithm in Action
  - Algorithm Properties
- Future Work

Future Work

- Extend the Algorithm to Support
  - Overlapping vocabularies
  - Defeasible Logic Local Theories
  - Non-Boolean queries
- Study Applications in the AmI Domain
  - Rules may represent ontological knowledge, policies, or regulations

Distributed Reasoning with Conflicts in an Ambient Peer-to-Peer Setting

Thank You!

Questions?