



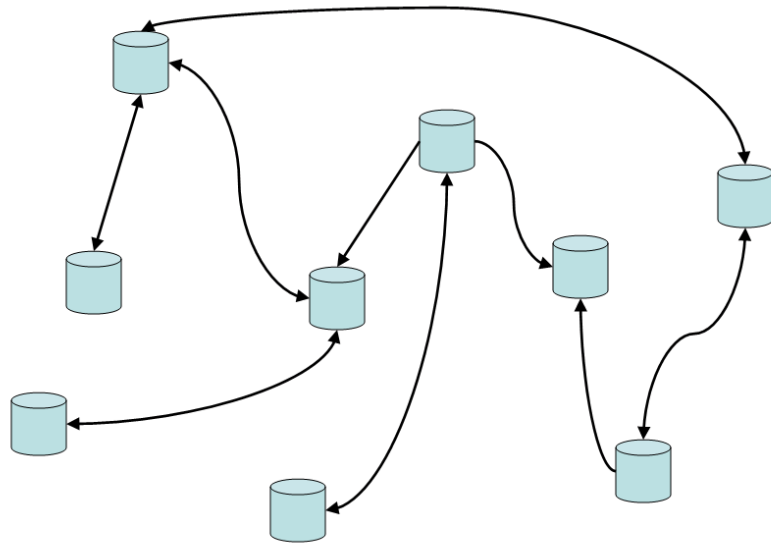
Information Agents in a Peer Data Exchange System

Leopoldo Bertossi

Carleton University
Ottawa, Canada

Join work with: Loreto Bravo, University of Edinburgh

Peer-to-Peer Data Exchange Systems



Each peer has a **local and autonomous database**

Data at two different peers may be related by **data exchange constraints** (DECs) (or data mappings)

Local queries are posed to individual peers

Peers exchange data when they answer their queries

How data is exchanged depends on the query, the DECs, the data at the relevant peers, and also on **trust relationships** between peers

A peer **does not update** its physical instance according to its DEC's and other peers' instances

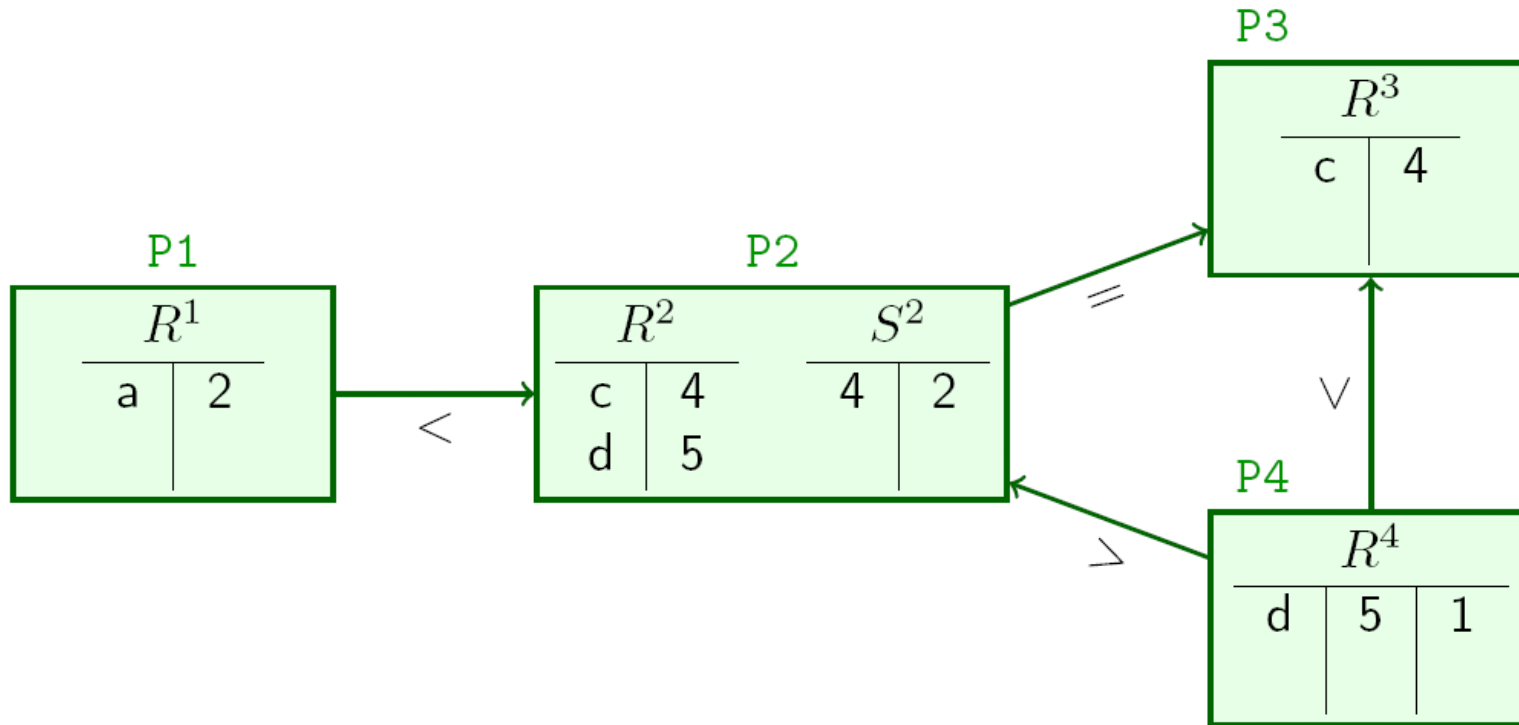
However, if a peer P is answering a (local) query Q_P , it may, **at query time**:

- ▶ Import data from other peers to complement its data
- ▶ Ignore part of its own data

All this depending upon its own DEC's and the peers' instances

But also upon the **trust relationships** that P has with other peers

Example: Peers in a systems \mathcal{P} , their schemas, instances, DECs, trust relationships:



DECs:

$$\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$$

$$\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

$$\Sigma(P4, P2) = \{\forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z))\}$$

$$\Sigma(P4, P3) = \{\forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z))\}$$

Our motivation:

- ▶ Present a formal semantics for such a system of peers who exchange data for query answering
- ▶ Role of a semantics: to **characterize in precise terms what are the intended and correct answers to a query** posed to and answered by a peer in the system
- ▶ We propose a *model-theoretic semantics*

That is, a collection of possible and admissible models over which the system is interpreted

- ▶ The expected answers from a peer to a query are those that are *certain* wrt a set of database instances associated to that peer
- ▶ The **declarative semantics can be made executable**, by using logic programs with stable model semantics that specify the intended models

- ▶ A peer data exchange system (PDES) can be seen as a *set of information agents*
- ▶ Each of them being the owner of a data source
- ▶ Each peer P can be seen as an *ontology* consisting of:
 - ▷ the database instance
 - ▷ metadata describing the *database schema* $\mathcal{R}(P)$ and local integrity constraints (ICs)
 - ▷ its set $\Sigma(P) = \bigcup_{P' \in \mathcal{P}} \Sigma(P, P')$ of DECs, and
 - ▷ its trust relationships
- ▶ These ontologies may be pairwise inconsistent due to the DECs and the database facts

It is easy to extend our framework to handle DECs that contain views, i.e. defined relational predicates

- ▶ This kind of consistency issues also emerge when aligning ontologies
- ▶ Our notion of DEC largely extends the inclusions of concepts in the ontological scenario

Our DEC's can be much more general than inclusions

- ▶ In our case, we do not make the ontologies mutually consistent

Whenever possible, inconsistencies are solved at query time

We consider **data exchange constraints (DECs)** of the following kinds:

- ▶ *Universal data exchange constraint* (UDEEC) between peers P1, P2:

$$\forall \bar{x} \left(\bigwedge_{i=1}^n R_i(\bar{x}_i) \longrightarrow \left(\bigvee_{j=1}^m Q_j(\bar{y}_j) \vee \varphi \right) \right)$$

$R_i, Q_j \in \mathcal{R}(P1) \cup \mathcal{R}(P2)$, and φ is a disjunction of built-in atoms

We can have a predicates of both peers on both sides of implication

- ▶ *Referential data exchange constraint* (RDEC) between peers P1, P2:

$$\forall \bar{x} (R(\bar{x}) \longrightarrow \exists \bar{y} Q(\bar{x}', \bar{y}))$$

$R, Q \in \mathcal{R}(P1) \cup \mathcal{R}(P2)$, and $\bar{x}' \subseteq \bar{x}$

When a peer P receives a query:

- ▶ P sends queries to its neighbors to check the satisfaction of its DECs

This can be made relative to the specific query at hand
(relevance of DECs to the query)

- ▶ If they are not satisfied, P tries to restore satisfaction of (consistency wrt) its DECs
 - ▷ A repair, that satisfies the DECs, respects the trust relationships and is “as close as possible” to P ’s original data is called a Solution for P
- ▶ There might be many solutions
 - ▷ Peer P will only return the data or query answers that are certain, i.e. that are true in all of P ’s solutions
 - ▷ These are the peer consistent answers (PCAs) from P

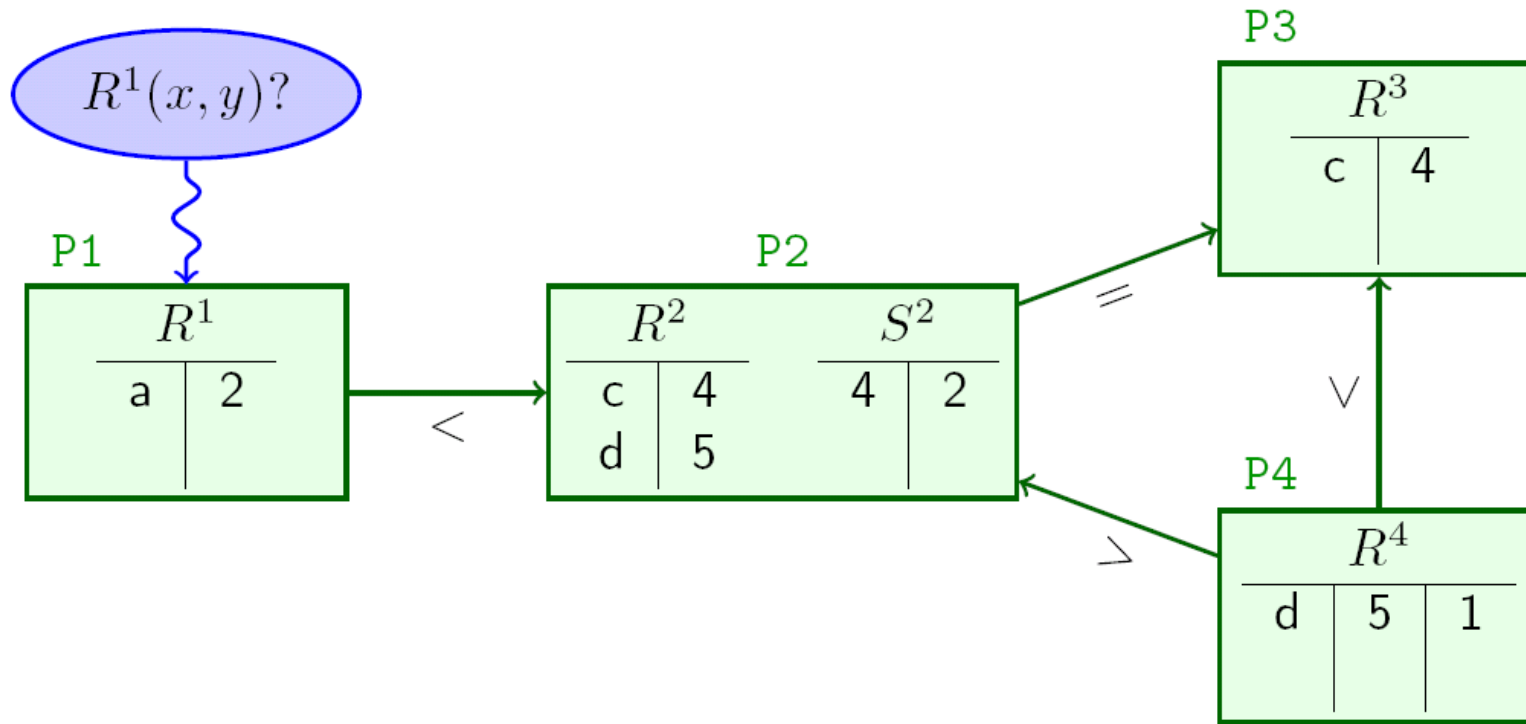
► What did P receive from a neighbor P'?

The answers from P' to its queries to P are also true in all solutions for P'

The same idea/process has to be applied to P's neighbors, and so on ...

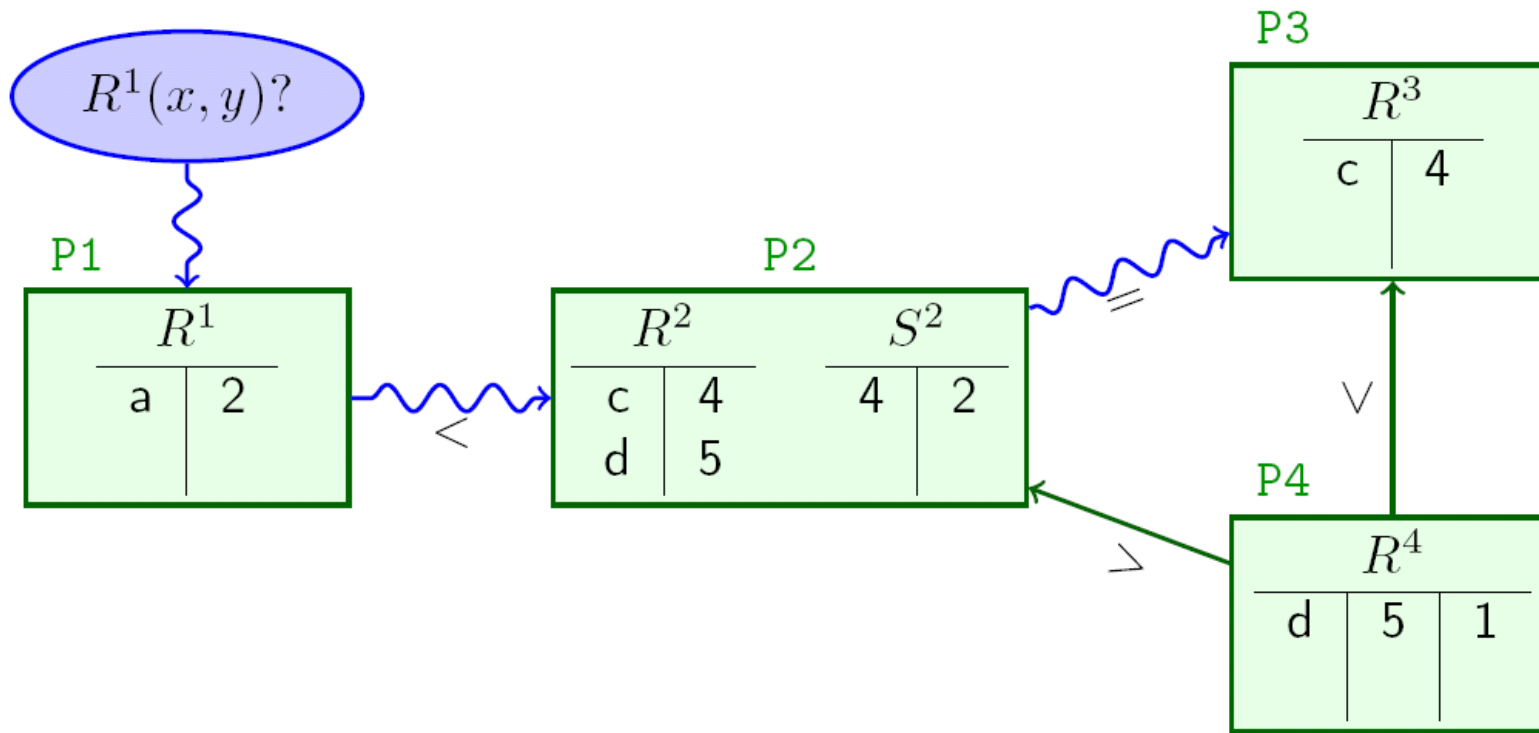
Peers pass (locally) certain data to other neighboring peers

Idea, Intuitions, “Operational” Semantics



DECs:

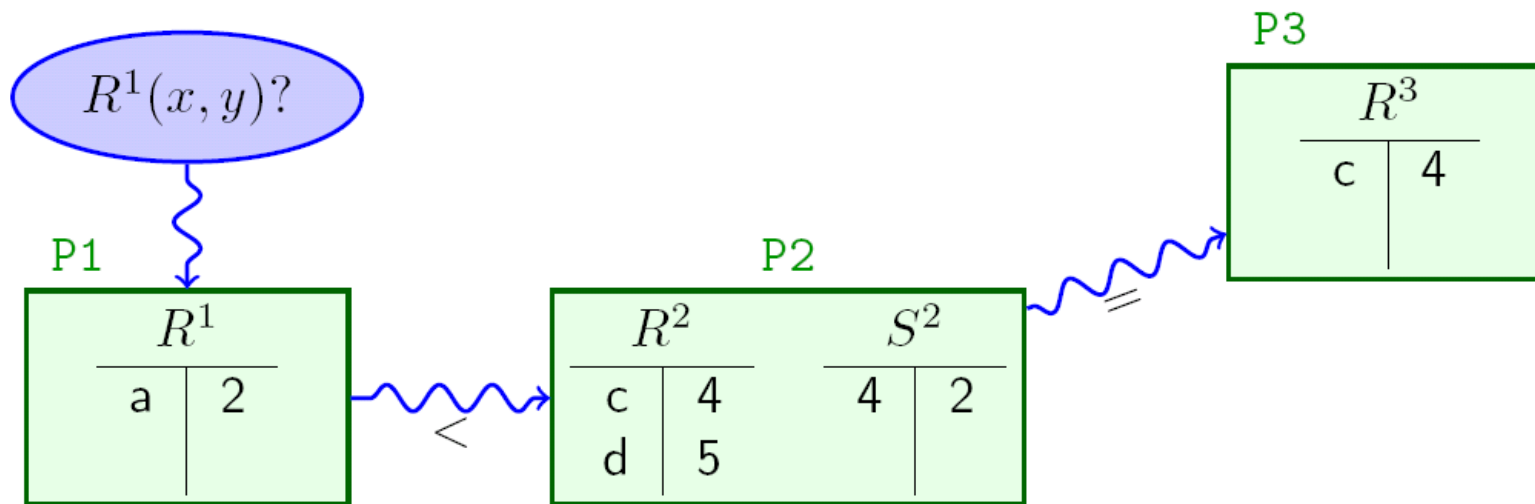
$$\left. \begin{aligned}
 \Sigma(\text{P1}, \text{P2}) &= \{ \forall x \forall y (R^2(x, y) \rightarrow R^1(x, y)) \} \\
 \Sigma(\text{P2}, \text{P3}) &= \{ \forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false}) \} \\
 \Sigma(\text{P4}, \text{P2}) &= \{ \forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z)) \} \\
 \Sigma(\text{P4}, \text{P3}) &= \{ \forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z)) \}
 \end{aligned} \right\} \begin{array}{l} \text{UDECs} \\ \text{RDEC} \end{array}$$



DECs:

$$\begin{aligned}
 \Sigma(P1, P2) &= \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\} \\
 \Sigma(P2, P3) &= \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\} \\
 \Sigma(P4, P2) &= \{\forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z))\} \\
 \Sigma(P4, P3) &= \{\forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z))\}
 \end{aligned}
 \left. \begin{array}{l} \\ \\ \\ \end{array} \right\} \begin{array}{l} \text{UDECs} \\ \text{RDEC} \end{array}$$

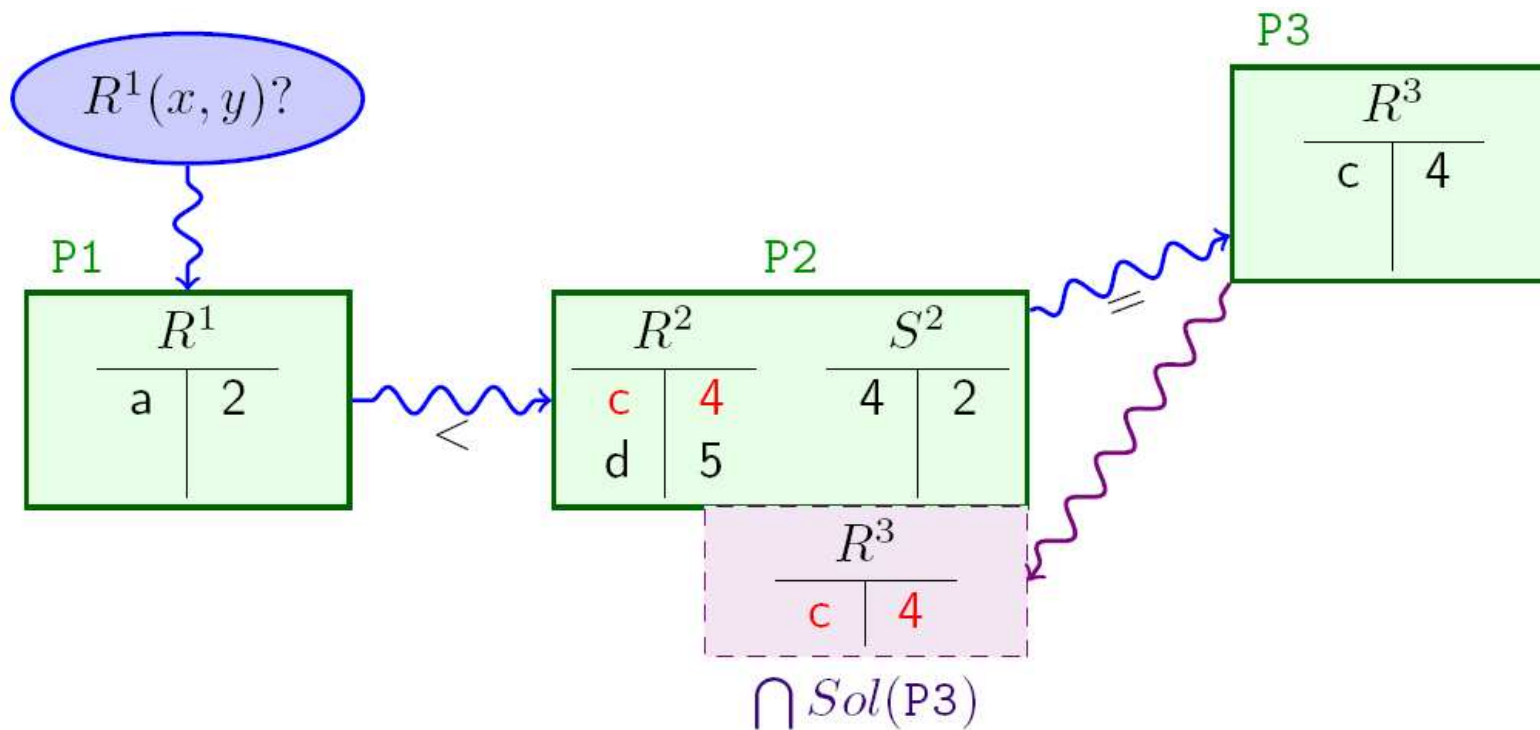
Peer P4 will have no effect on the query!



DECs:

$$\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$$

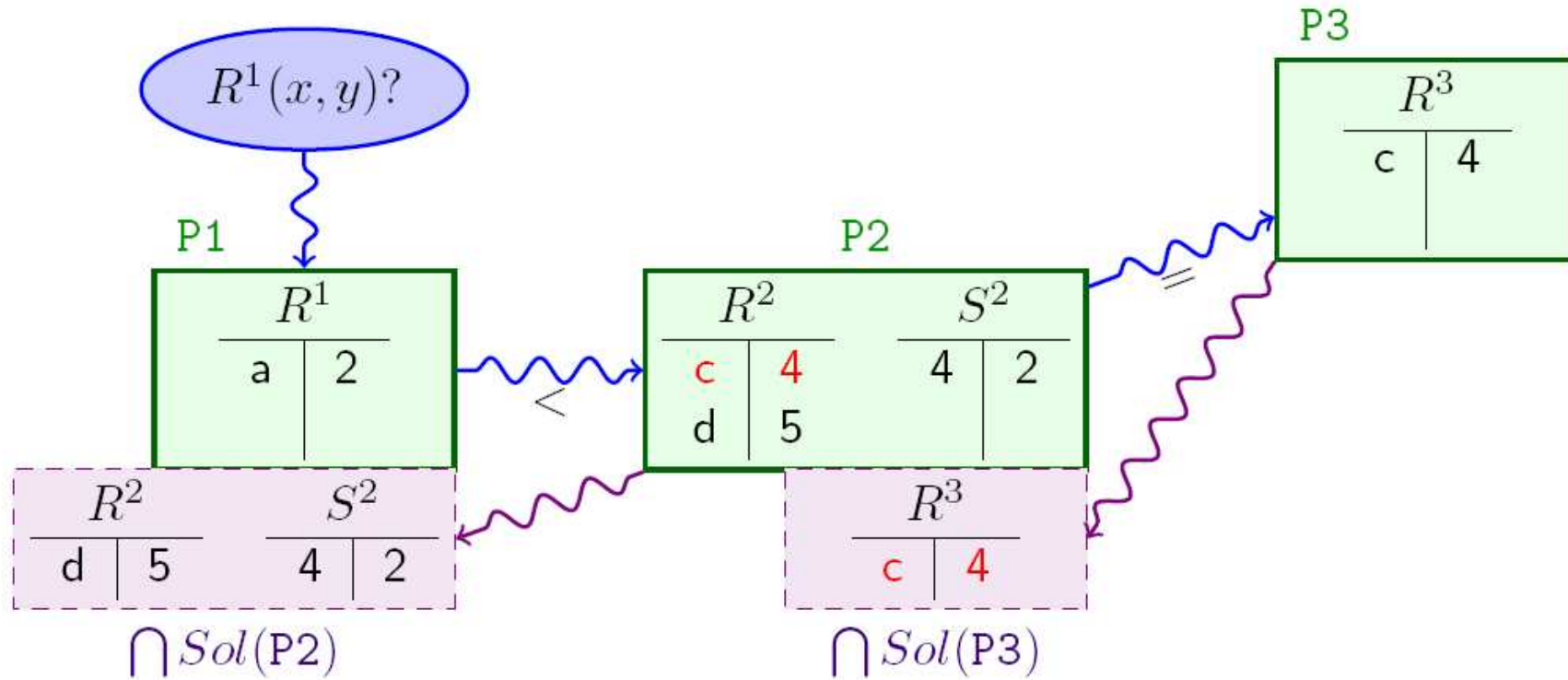
$$\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\} \quad (\text{a denial DEC})$$



DECs:

$$\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$$

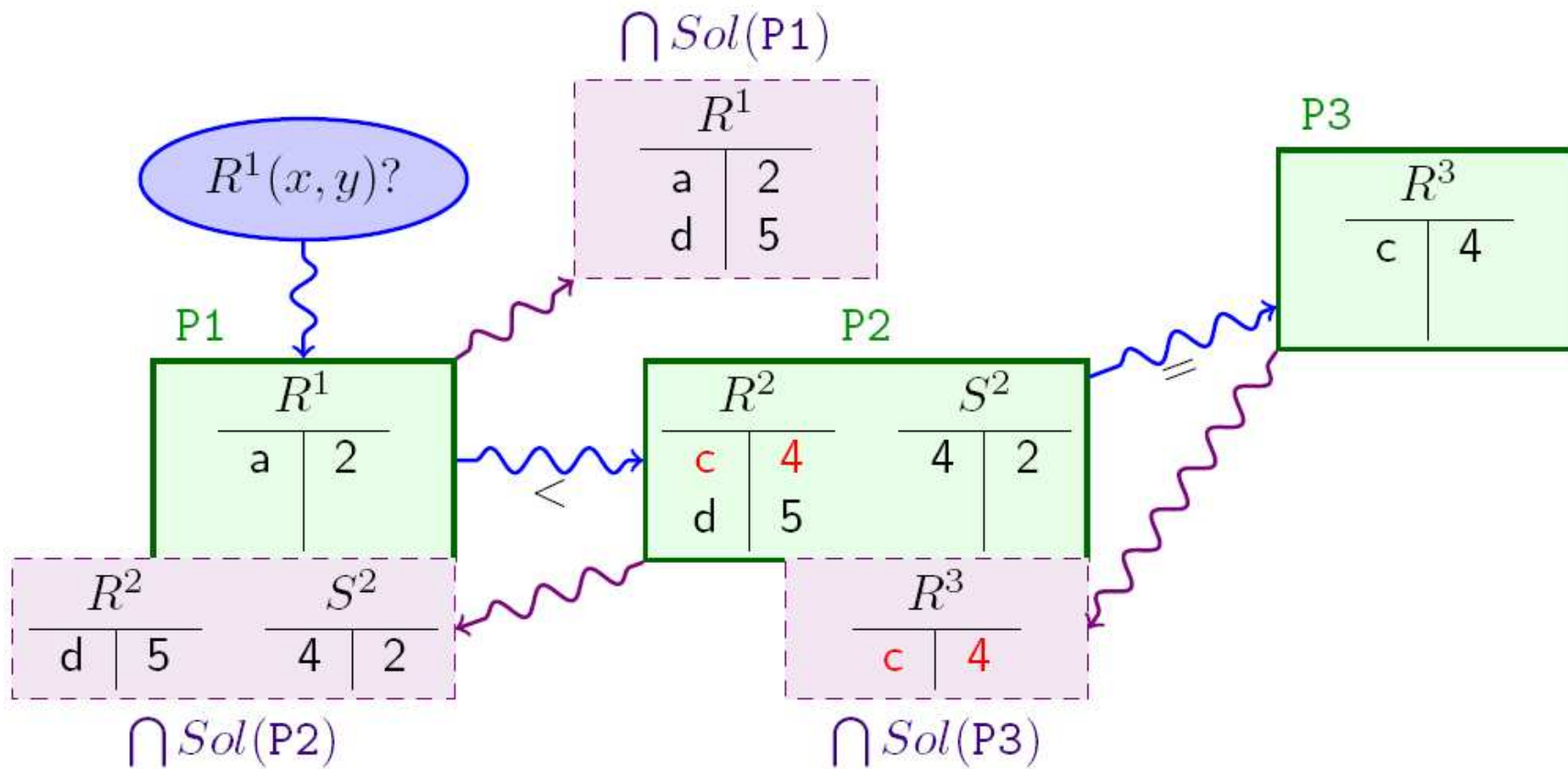
$$\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$



DECs:

$$\Sigma(P1, P2) = \{ \forall x \forall y (R^2(x, y) \rightarrow R^1(x, y)) \}$$

$$\Sigma(P2, P3) = \{ \forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false}) \}$$



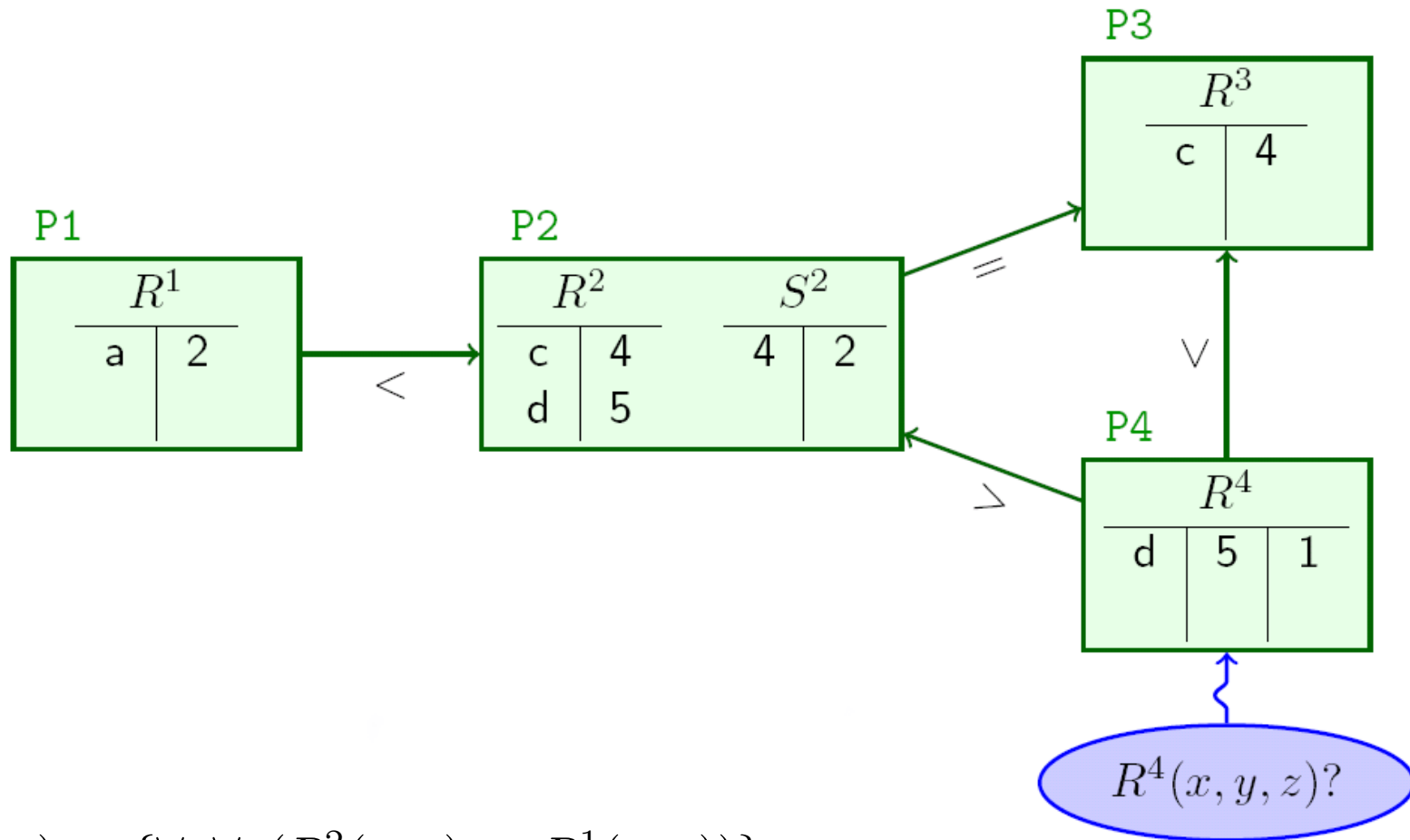
DECs:

$$\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$$

$$\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

Peer consistent answers from P1: $(a, 2)$ and $(d, 5)$

Example:



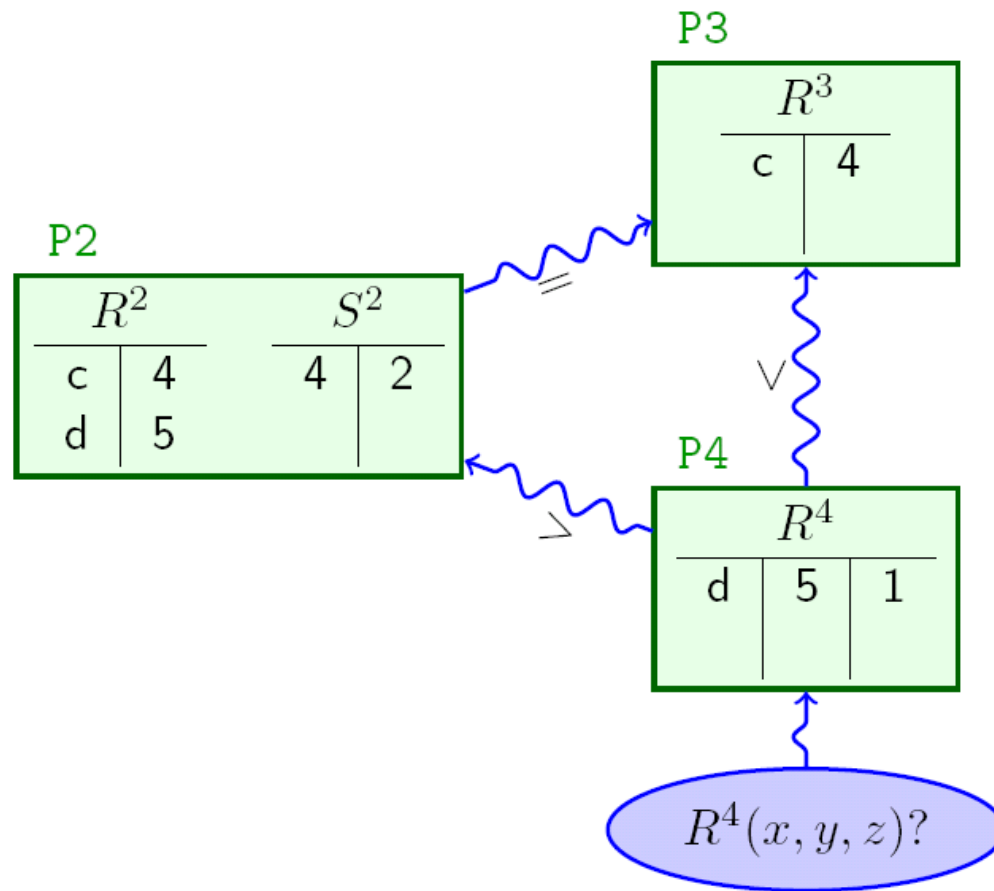
DECs:

$$\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$$

$$\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

$$\Sigma(P4, P2) = \{\forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z))\}$$

$$\Sigma(P4, P3) = \{\forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z))\}$$

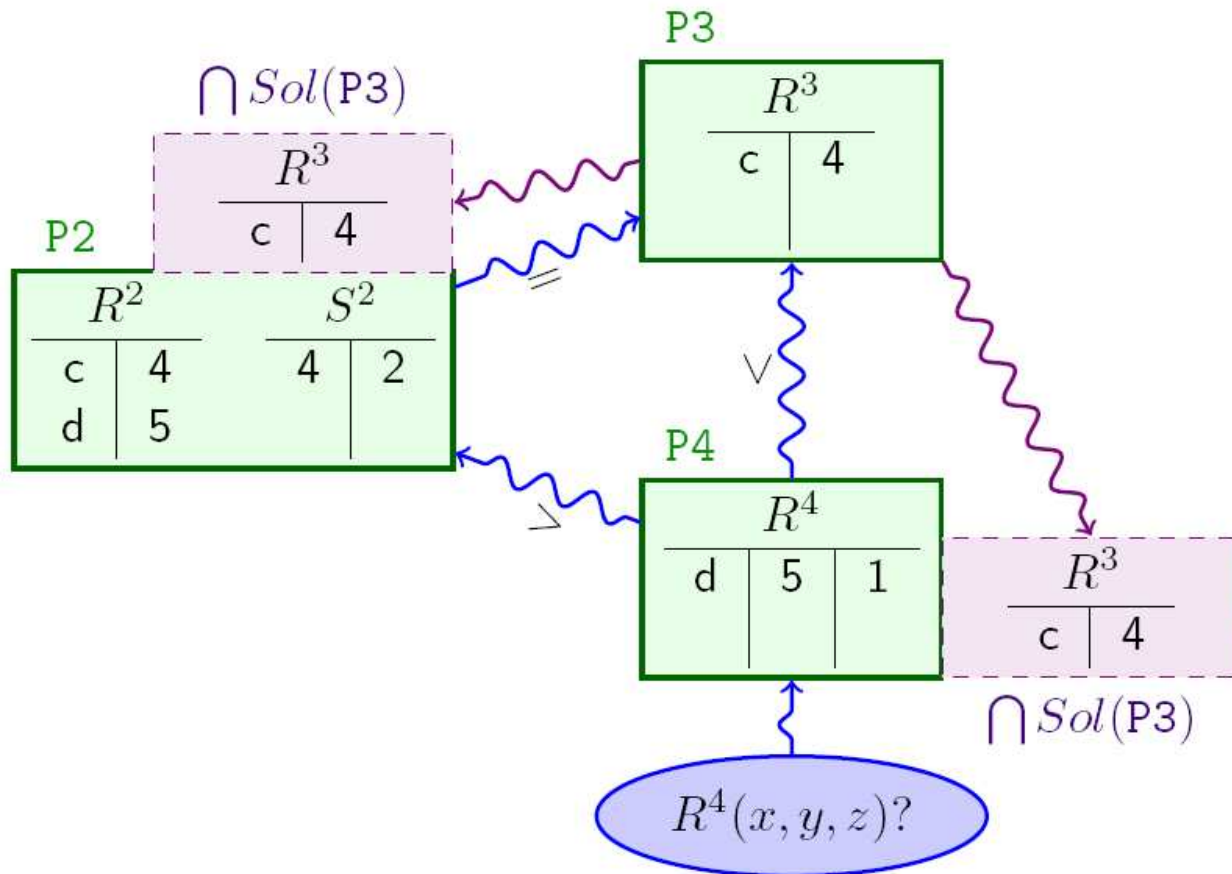


DECs:

$$\Sigma(\text{P2}, \text{P3}) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

$$\Sigma(\text{P4}, \text{P2}) = \{\forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z))\}$$

$$\Sigma(\text{P4}, \text{P3}) = \{\forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z))\}$$

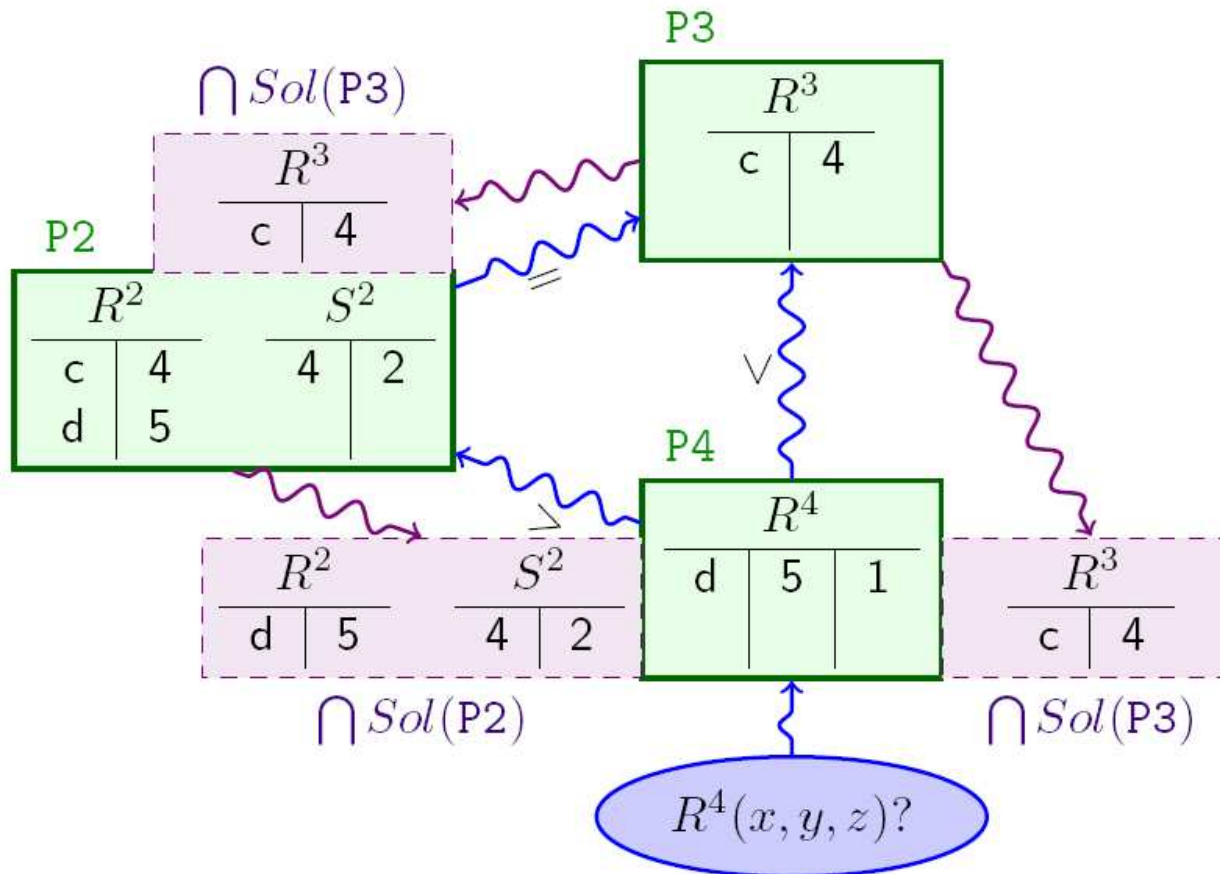


DECs:

$$\Sigma(\text{P2}, \text{P3}) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

$$\Sigma(\text{P4}, \text{P2}) = \{\forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z))\}$$

$$\Sigma(\text{P4}, \text{P3}) = \{\forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z))\}$$

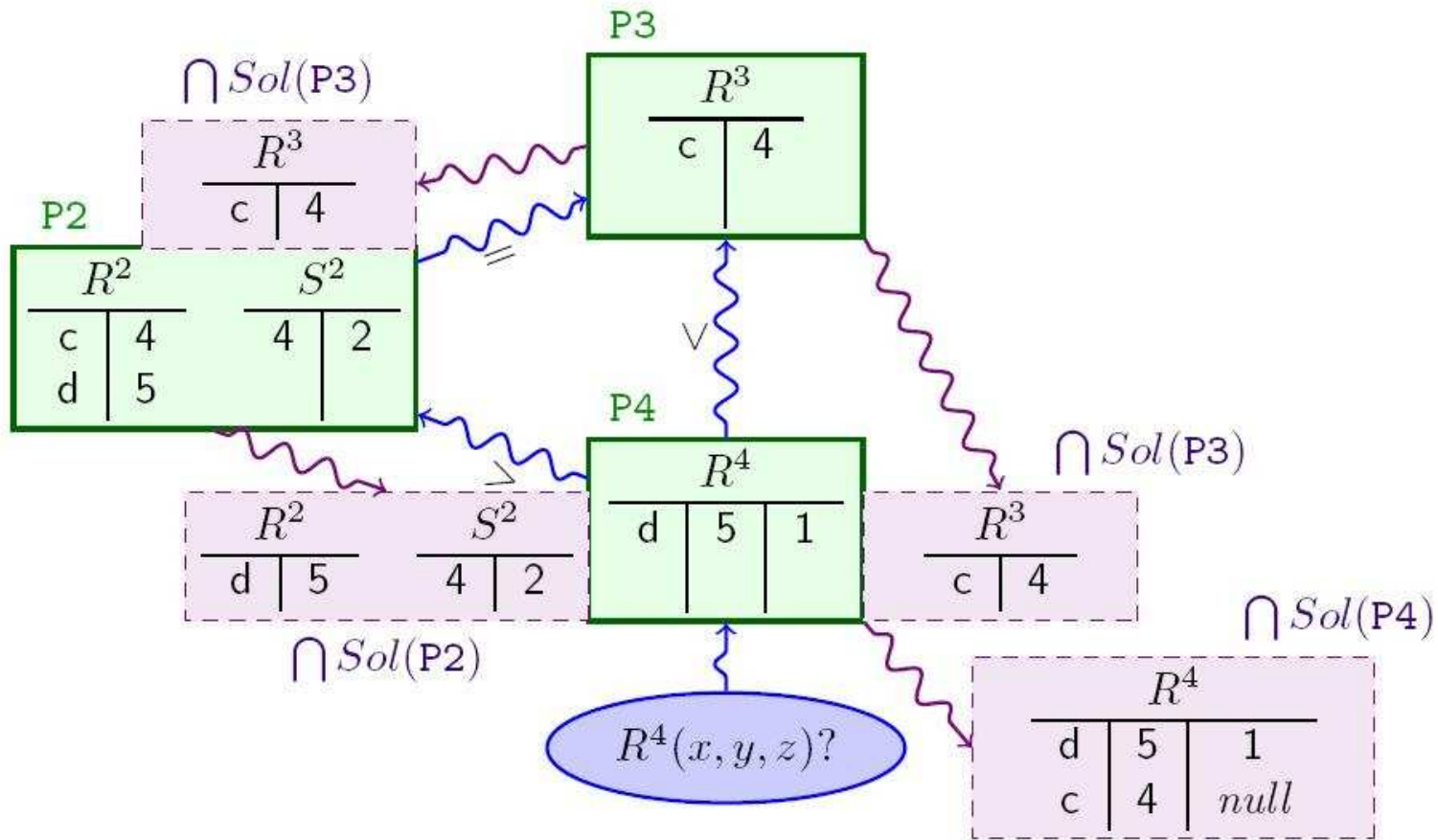


DECs:

$$\Sigma(\text{P2}, \text{P3}) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

$$\Sigma(\text{P4}, \text{P2}) = \{\forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z))\}$$

$$\Sigma(\text{P4}, \text{P3}) = \{\forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z))\}$$



DECs:

$$\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

$$\Sigma(P4, P2) = \{\forall x \forall y \forall z (R^2(x, y) \wedge S^2(y, z) \rightarrow R^4(x, y, z))\}$$

$$\Sigma(P4, P3) = \{\forall x \forall y (R^3(x, y) \rightarrow \exists z R^4(x, y, z))\}$$

PCAs from P4: $(d, 5, 1)$ and $(c, 4, null)$

- ▶ **Incomplete information** is represented by means of null values

This is important

Actually, they follow a FO semantics that is a “logical reconstruction” of IC satisfaction with **nulls in the SQL Standard** (Bravo, Bertossi; IIDB 2006)

- ▶ This is why we do not accept tuple-generating DEC's via joins (existential quantification over a join variable)

However, we could extend RDEC's with more complex antecedents of the implication

- ▶ Other semantics for incomplete databases could be adopted in this framework
- ▶ It is also possible to impose **local IC's** on peers' instances

They can be uniformly handled as before by means of DEC's of the form $\Sigma(P, P)$ and an $=$ -trust relationship

What follows?

1. A logic-based (model-based) semantics
2. Capturing peers' solutions as models of logic programs

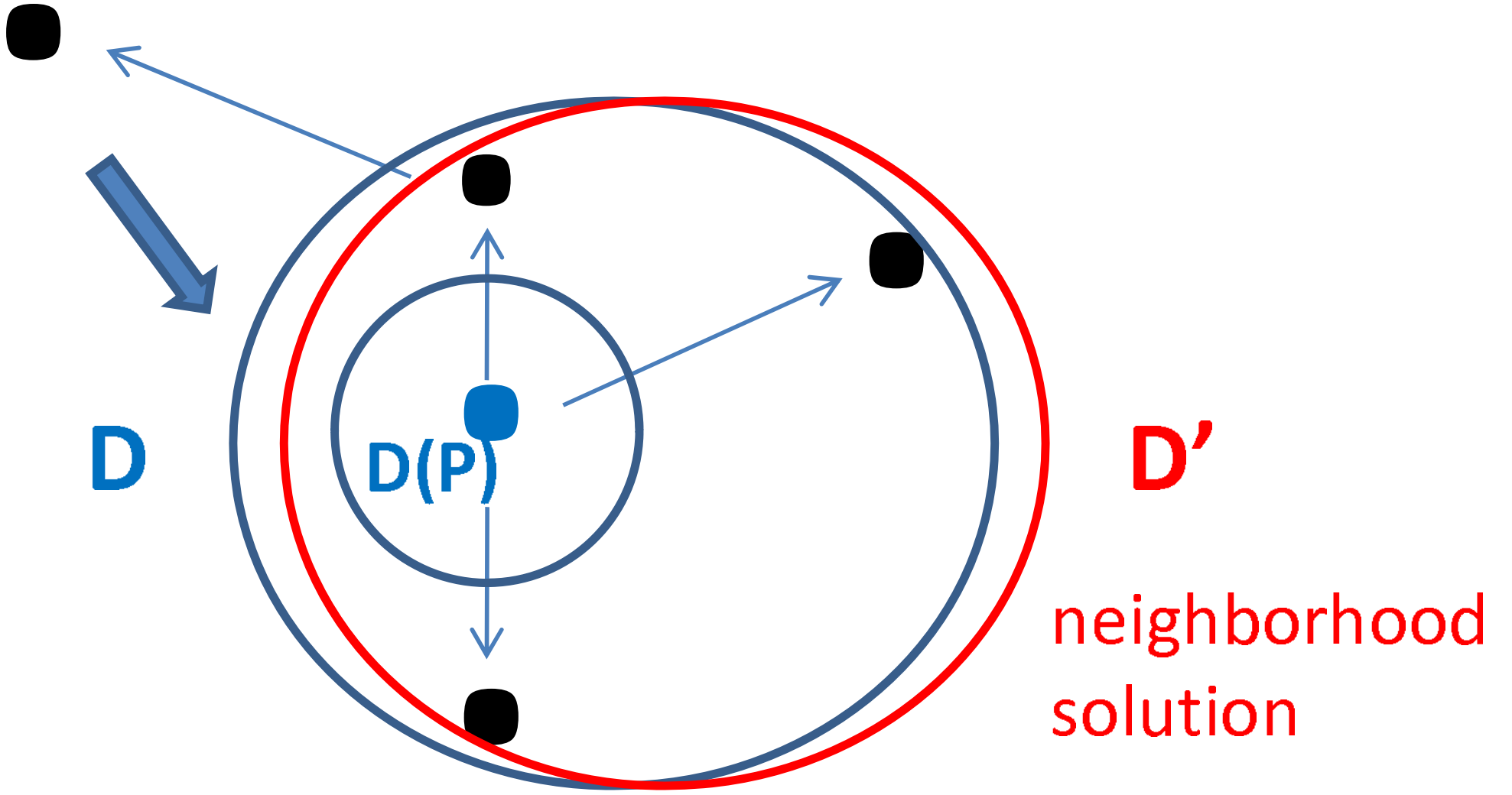
Formal Semantics

Peer P and instances D, D' for the union schema of P and its neighbors

D' is a *neighborhood solution for P and D* :

1. [satisfaction] D' satisfies the DECs and local constraints of P
2. [trust] The data in D' associated to the peers that P trusts more than itself is the same as the one in D
3. [minimality] There is no instance D'' that satisfies 1. and 2. that D'' is “closer” to D than D'

Closeness is defined in such a way that: changes are minimal under set inclusion of sets of tuples (Arenas, Bertossi, Chomicki, PODS 1999), and referential DECs are repaired by insertions of *null*, as in (Bravo, Bertossi; IIDB 2006)



Now we have to consider **transitive relationships**

Let $D(P)$ be the database instance for peer P

A **solution** for P can be recursively defined as:

- ▶ If P has no DEC's, then the solution is $D(P)$
- ▶ Otherwise:
 1. Let D be a database instance containing the union of
 - ▷ $D(P)$
 - ▷ for each neighboring peer, the intersection of its solutions

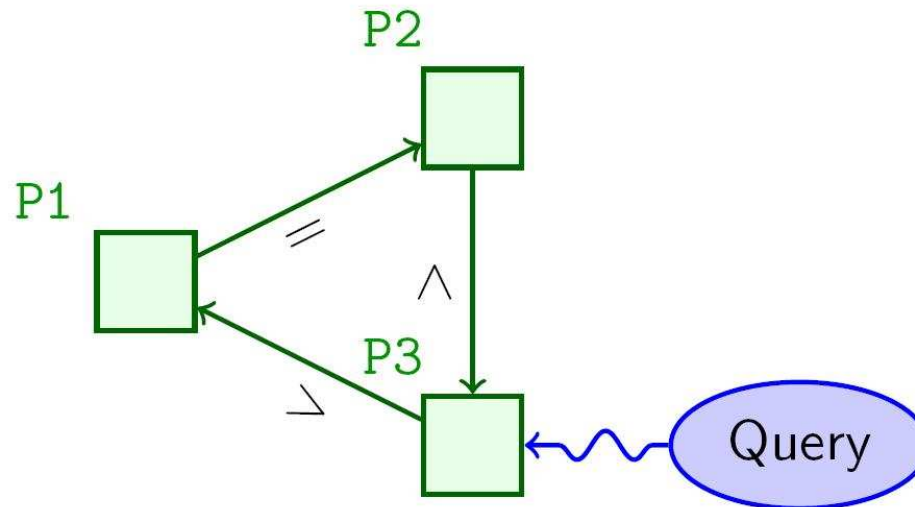
This an instance over the union schema of the neighborhood around P

2. Let D' be a **neighborhood solution** for P and D

Then, D' restricted to the schema of P is a **solution** for P

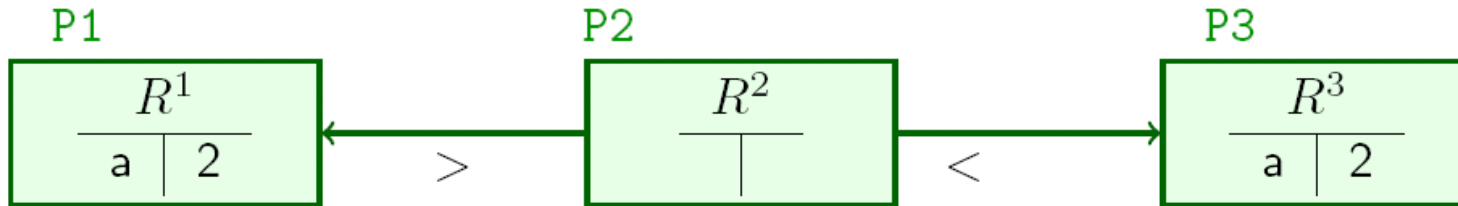
We restrict to **acyclic** peer data exchange systems, i.e. graphs of neighbors is acyclic (not necessarily the DECAs)

To avoid the following:



- ▶ P3 needs the intersection of P1's solutions \Rightarrow P1 needs the intersection of P2's solutions \Rightarrow P2 needs the intersection of P3's solutions $\Rightarrow \dots$
- ▶ Cycles can be detected by using a query identifier that is propagated as an annotation and detected

It might be the case that a peer has no solution:



DECs:

$$\Sigma(P2, P1) = \{\forall x \forall y (R^1(x, y) \rightarrow R^2(x, y))\}$$

$$\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$$

Peer P2 trusts P1 and P3 more than itself, but both provide contradictory information

First-order query $Q(\bar{x}) \in L(\mathbb{P})$ posed to peer \mathbb{P} :

The ground tuple \bar{t} is a *peer consistent answer* to Q from \mathbb{P} iff $D' \models Q(\bar{t})$ for every solution instance D' of \mathbb{P}

Theorem: Deciding if a tuple is a peer consistent answer to a query is Π_2^P –complete

Our null-based repair semantics allow for decidability

Even with acyclic neighbors' graph acyclic there could be undecidability depending on interaction between DEC's if arbitrary values from domain were used

Logic Programs for Peers' Solutions

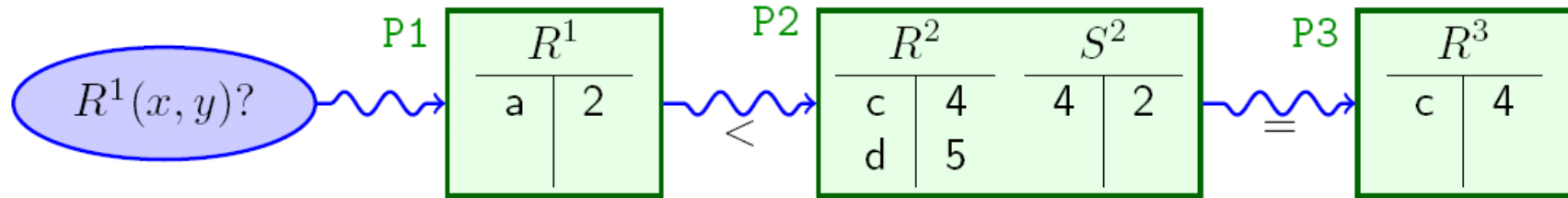
- ▶ Answer set programs have been used to specify and compute repairs of databases that are inconsistent wrt ICs (Barcelo, Bertossi; PADL'03), (Barcelo, Bertossi, Bravo; LNCS 2582)
- ▶ Those programs can be adapted to our framework, and there is a one-to-one correspondence between their answer sets (stable models) and the solutions of peer
 - ▷ The trust relationships, DEC's and local ICs have to be taken into consideration
- ▶ In this we will have a compact representation of the class of solutions and the possibility of reasoning about that class

- The program uses annotation constants to indicate the atoms that may virtually inserted or deleted in order to restore consistency:

Annotation	Atom	The tuple $P(\bar{a})$ is ...
\mathbf{t}_a	$P_-(\bar{a}, \mathbf{t}_a)$	advised to be made true
\mathbf{f}_a	$P_-(\bar{a}, \mathbf{f}_a)$	advised to be made false
\mathbf{t}^*	$P_-(\bar{a}, \mathbf{t}^*)$	true or becomes true
\mathbf{f}^*	$P_-(\bar{a}, \mathbf{f}^*)$	false or becomes false
\mathbf{t}^{**}	$P_-(\bar{a}, \mathbf{t}^{**})$	true in the solution

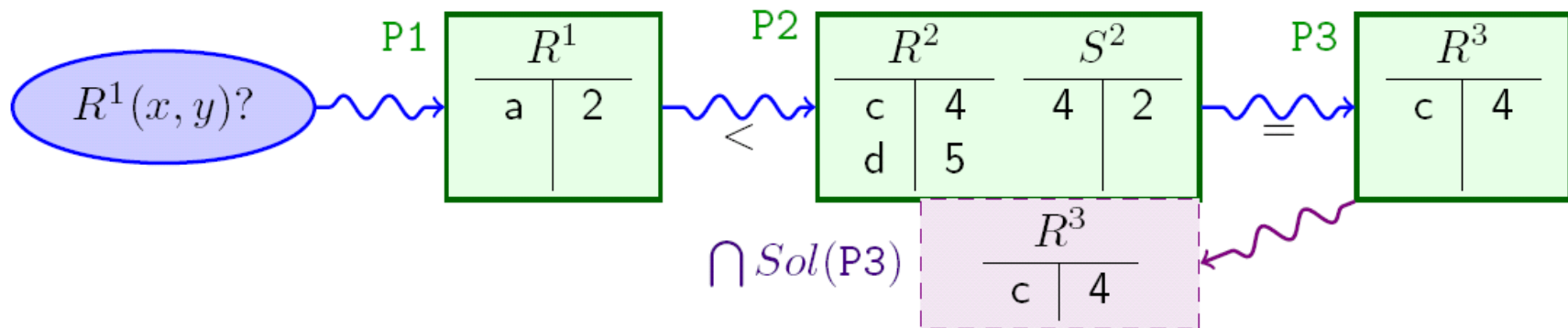
All of them needed if there are interacting DEC's for a peer; and several repair steps become necessary

Example: $\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$
 $\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$



- Peer P3 has no DEC's, therefore its only solution is $D(P3)$

DECs: $\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$
 $\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$



- To find the solution of peer P2 we can use its solution program!

$dom(a). \quad dom(c). \quad \dots$ $R^3(c, 4) \quad R^2(c, 4) \quad R^2(d, 5) \quad S^2(4, 2)$ $R^3_-(x, y, \mathbf{f}_a) \vee R^2_-(x, y, \mathbf{f}_a) \leftarrow R^2_-(x, y, \mathbf{t}^*), R^3_-(x, y, \mathbf{t}^*), x \neq null, y \neq null.$ $R^3_-(x, y, \mathbf{t}^*) \leftarrow R^3_-(x, y, \mathbf{t}_a).$ $R^3_-(x, y, \mathbf{t}^*) \leftarrow R^3_-(x, y).$ $R^3_-(x, y, \mathbf{f}^*) \leftarrow R^3_-(x, y, \mathbf{f}_a).$ $R^3_-(x, y, \mathbf{f}^*) \leftarrow dom(x), dom(y), not R^3(x, y).$ $R^2_-(x, y, \mathbf{t}^{**}) \leftarrow R^2_-(x, y, \mathbf{t}^*), not R^2_-(x, y, \mathbf{f}_a).$ $\leftarrow R^3_-(x, y, \mathbf{t}_a), R^3_-(x, y, \mathbf{f}_a).$	}	(Similarly for R^2)
--	---	------------------------

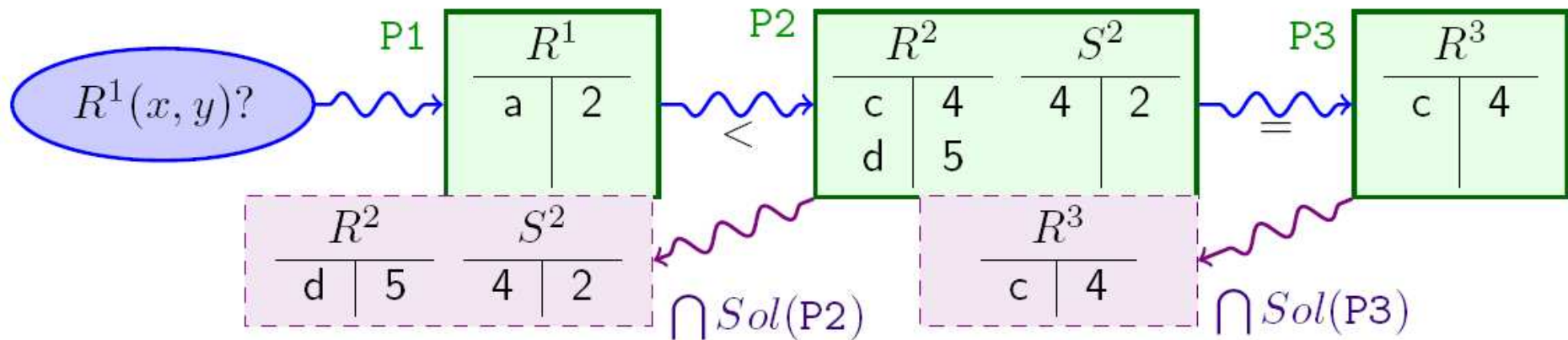
$dom(a).$	$dom(c).$	\dots		
$R^3(c, 4)$	$R^2(c, 4)$	$R^2(d, 5)$	$S^2(4, 2)$	
$R^3_-(x, y, \mathbf{f}_a) \vee R^2_-(x, y, \mathbf{f}_a) \leftarrow R^2_-(x, y, \mathbf{t}^*), R^3_-(x, y, \mathbf{t}^*), x \neq null, y \neq null.$				
$R^3_-(x, y, \mathbf{t}^*) \leftarrow R^3_-(x, y, \mathbf{t}_a).$	} (Similarly for R^2)			
$R^3_-(x, y, \mathbf{t}^*) \leftarrow R^3(x, y).$				
$R^3_-(x, y, \mathbf{f}^*) \leftarrow R^3_-(x, y, \mathbf{f}_a).$				
$R^3_-(x, y, \mathbf{f}^*) \leftarrow dom(x), dom(y), not R^3(x, y).$				
$R^2_-(x, y, \mathbf{t}^{**}) \leftarrow R^2_-(x, y, \mathbf{t}^*), not R^2_-(x, y, \mathbf{f}_a).$				
$\leftarrow R^3_-(x, y, \mathbf{t}_a), R^3_-(x, y, \mathbf{f}_a).$				

- ▶ This program has two answer sets
- ▶ By collecting the atoms with annotation constant \mathbf{t}^{**} we get (neighborhood) solutions:

$D_1 :$	$\begin{array}{c c} R^2 & \\ \hline d & 5 \\ \hline \end{array}$	$\begin{array}{c c} S^2 & \\ \hline 4 & 2 \\ \hline \end{array}$	$\begin{array}{c c} R^3 & \\ \hline c & 4 \\ \hline \end{array}$
$D_2 :$	$\begin{array}{c c} R^2 & \\ \hline c & 4 \\ d & 5 \\ \hline \end{array}$	$\begin{array}{c c} S^2 & \\ \hline 4 & 2 \\ \hline \end{array}$	$\begin{array}{c c} R^3 & \\ \hline & \\ \hline \end{array}$

$\cap?$: Just skeptical query answering, no materialization: $Ans(x, y) \leftarrow R^2(x, y, \mathbf{t}^{**})$

DECs: $\Sigma(P1, P2) = \{\forall x \forall y (R^2(x, y) \rightarrow R^1(x, y))\}$
 $\Sigma(P2, P3) = \{\forall x \forall y (R^2(x, y) \wedge R^3(x, y) \rightarrow \mathbf{false})\}$



- To find the solution of peer P1 we can use its solution program!

$dom(a).$ $dom(c).$... $R^1(a, 2)$ $R^2(d, 5)$ $S^2(4, 2)$ $R^1_-(x, y, \mathbf{t}_a) \leftarrow R^2_-(x, y, \mathbf{t}^*), R^1_-(x, y, \mathbf{f}^*), x \neq null, y \neq null.$ $R^1_-(x, y, \mathbf{t}^*) \leftarrow R^1_-(x, y, \mathbf{t}_a).$ $R^1_-(x, y, \mathbf{t}^*) \leftarrow R^1(x, y).$ $R^1_-(x, y, \mathbf{f}^*) \leftarrow R^1_-(x, y, \mathbf{f}_a).$ $R^1_-(x, y, \mathbf{f}^*) \leftarrow dom(x), dom(y), not R^1(x, y).$ $R^1_-(x, y, \mathbf{t}^{**}) \leftarrow R^1_-(x, y, \mathbf{t}^*), not R^1_-(x, y, \mathbf{f}_a).$ $\leftarrow R^1_-(x, y, \mathbf{t}_a), R^1_-(x, y, \mathbf{f}_a).$	}	(Similarly for R^2)
---	---	------------------------

$dom(a).$	$dom(c).$	\dots	
$R^1(a, 2)$	$R^2(d, 5)$	$S^2(4, 2)$	
$R^1_-(x, y, \mathbf{t}_a) \leftarrow R^2_-(x, y, \mathbf{t}^*), R^1_-(x, y, \mathbf{f}^*), x \neq null, y \neq null.$			
$R^1_-(x, y, \mathbf{t}^*) \leftarrow R^1_-(x, y, \mathbf{t}_a).$			}
$R^1_-(x, y, \mathbf{t}^*) \leftarrow R^1(x, y).$			
$R^1_-(x, y, \mathbf{f}^*) \leftarrow R^1_-(x, y, \mathbf{f}_a).$			
$R^1_-(x, y, \mathbf{f}^*) \leftarrow dom(x), dom(y), not R^1(x, y).$			
$R^1_-(x, y, \mathbf{t}^{**}) \leftarrow R^1_-(x, y, \mathbf{t}^*), not R^1_-(x, y, \mathbf{f}_a).$			
$\leftarrow R^1_-(x, y, \mathbf{t}_a), R^1_-(x, y, \mathbf{f}_a).$			(Similarly for R^2)

- ▶ This program has one answer set; then one neighbor solution
- ▶ By collecting from it the atoms with annotation constant \mathbf{t}^{**} we get this (neighborhood) solution:

$$D_1 : \begin{array}{c|c|c|c} & \overline{R^1} & \overline{R^2} & \overline{S^2} \\ \hline a & 2 & d & 5 \\ \hline d & 5 & & 4 \\ & & & 2 \end{array}$$

- ▶ The query $R^1(x, y)?$ can also be added to the answer set program:

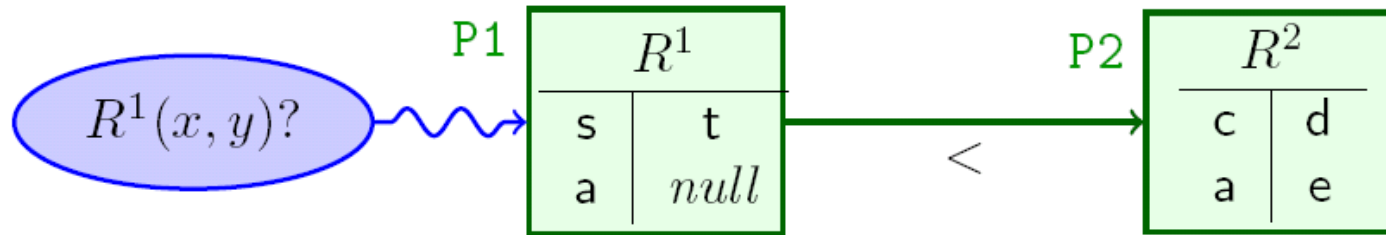
$$Ans(x, y) \leftarrow R^1(x, y, \mathbf{t}^{**})$$

- ▶ The query answer is then obtained from the result $\{Ans(a, 2), Ans(d, 5)\}$

Example:

$$\Sigma(\text{P1}, \text{P2}) = \{\forall xy (R^2(x, y) \rightarrow \exists z R^1(x, z))\}$$

$$IC(\text{P1}) = \{\forall xyz (R^1(x, y) \wedge R^1(x, z) \rightarrow y = z)\}$$



$dom(a). dom(b). \dots R^1(a, null). R^1(s, t). R^2(c, d). R^2(a, e).$
 $R^1_-(x, null, \mathbf{t}_a) \leftarrow R^2_-(x, \mathbf{t}^*), \text{ not } aux(x), x \neq null.$
 $aux(x) \leftarrow R^1(x, null), \text{ not } R^1_-(x, null, \mathbf{f}_a).$
 $aux(x) \leftarrow R^1(x, y, \mathbf{t}^*), \text{ not } R^1_-(x, y, \mathbf{f}_a), x \neq null, y \neq null.$
 $R^1(x, y, \mathbf{f}_a) \vee R^1(x, z, \mathbf{f}_a) \leftarrow R^1(x, y, \mathbf{t}^*), R^1(x, z, \mathbf{t}^*), x \neq null, y \neq z.$

The solution obtained from the answer set is:

R^1	
s	t
a	null
c	null

► A set of DEC's and IC's is **Ref-acyclic** if there is no cycles through referential DEC's or IC's

► For example:

▷ Ref-acyclic

$$\Sigma(P1, P2) = \{\forall xy (R^1(x, y) \rightarrow R^2(x, y)), \\ \forall xy (R^2(x, y) \rightarrow R^1(x, y))\}$$

$$\Sigma(P2, P1) = \{\forall x(S^2(x) \rightarrow \exists yS^1(x, y))\}$$

▷ Not ref-acyclic

$$\Sigma(P1, P2) = \{\forall xy (R^1(x, y) \rightarrow \exists z R^2(x, z)), \\ \forall xy (R^2(x, y) \rightarrow R^1(x, y))\}$$

Theorem: for a ref-acyclic set of DEC's and IC's, there is a **one-to-one correspondence** between answer sets and solutions of a peer

Special Case

- ▶ **Unrestricted Import Case:**

- ▷ The DEC's are such that data is only **imported** to the peer (nothing is deleted)
- ▷ All peers trust other peers more than themselves

- ▶ Nice properties:

- ▷ A **solution always exist**
- ▷ The solution program can be replaced by a non-disjunctive program:

The problem of determining if a tuple is a peer consistent answer to a query is in **coNP** (could be coNP-complete depending on the query)

Optimizations and Relaxing Conditions

- ▶ It is possible to relax the conditions of ref-acyclicity and acyclicity of the neighbors' graph and:
 - ▷ A sensible semantics can be provided
 - ▷ Correct and complete solution-programs can be given

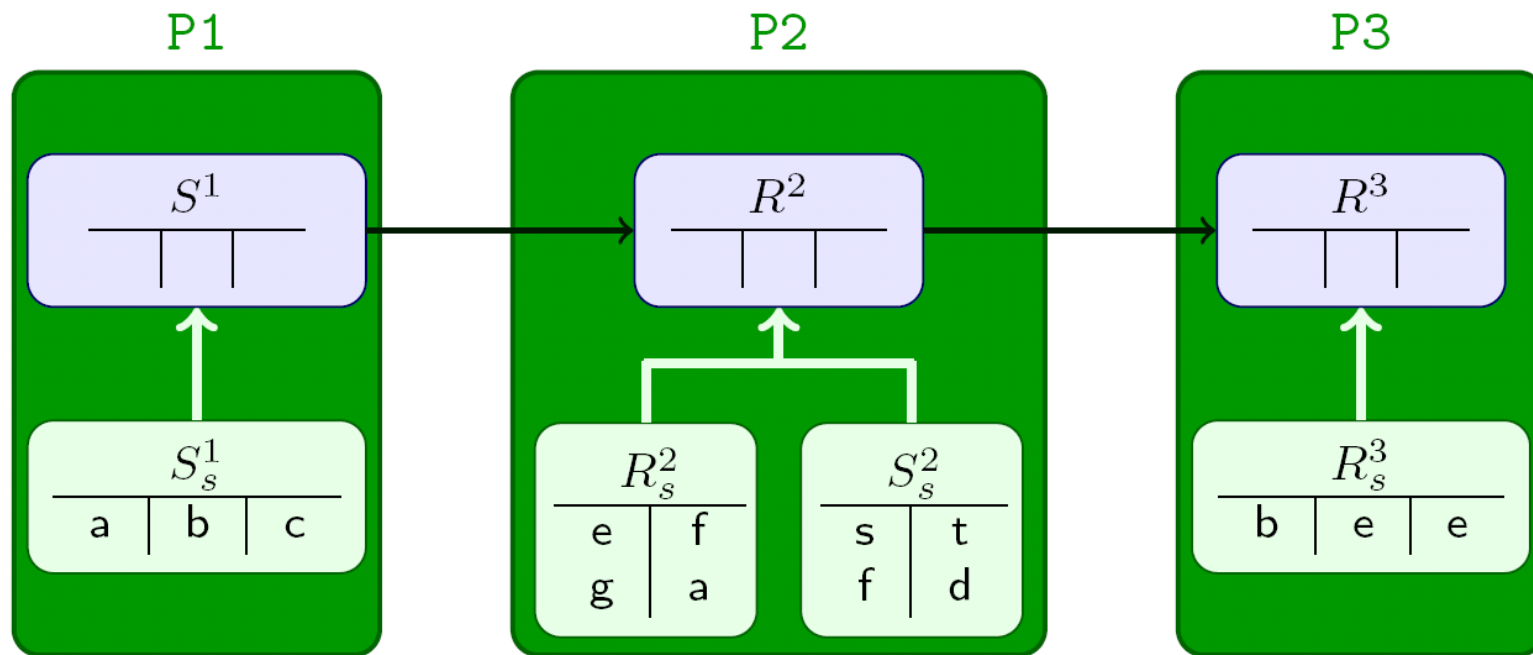
For example when:

- ▷ The cycles in the graph are not relevant to the query
 - ▷ Even if the DEC's and IC's are not ref-acyclic, depending on the interaction with the trust relationships, the solution-program can provide exactly the set of solutions
- ▶ Instead of requesting all the data of the neighboring sources:
 - ▷ restrict to the data that is relevant to check the DEC's that have an impact on a query

Related Work

(Calvanese et al.; DBISP2P 2003, DBPL 2005, PODS 2004),
(Franconi et al.; P2P&DB 2004)

- ▶ Based on **epistemic logic**
- ▶ DEC's are of the form: $cq_i \rightarrow cq_j$
where cq_i and cq_j are conjunctive queries over P_i and P_j 's schemas, resp.
- ▶ **No trust relationships**
 - ▷ Implicitly: peers trust themselves less than other peers
- ▶ **Local IC's violations are avoided**
 - ▷ A peer that is inconsistent wrt its local IC's is ignored
 - ▷ New atoms are added into a peer by interaction with other peers only if this does not produce a local IC violation



► GAV Local Mappings:

$$\forall xyz(S_s^1(x, y, z) \rightarrow S^1(x, y, z))$$

$$\forall xyz(R_s^2(x, y) \wedge R_s^2(y, z) \rightarrow R^2(x, y, z))$$

$$\forall xyz(R_s^3(x, y, z) \rightarrow R^3(x, y, z))$$

► DECs:

$$\forall xy(R^2(x, y, z) \rightarrow \exists wR^1(x, y, w))$$

$$\forall xy(R^3(x, y, y) \rightarrow \exists uvR^3(u, x, v))$$

If peer P1 receives a query, it will need the following theory in epistemic logic:

$$\begin{array}{l}
 \mathbf{K}_1(\forall xyz(S_s^1(x, y, z) \rightarrow S^1(x, y, z))) \\
 \forall xy(\mathbf{K}_2(R^2(x, y, z)) \rightarrow \mathbf{K}_1(\exists wR^1(x, y, w))) \\
 \mathbf{K}_2(\forall xyz(R_s^2(x, y) \wedge R_s^2(y, z) \rightarrow R^2(x, y, z))) \\
 \forall xy(\mathbf{K}_3(R^3(x, y, y)) \rightarrow \mathbf{K}_2(\exists uvR^3(u, x, v))) \\
 \mathbf{K}_3(\forall xyz(R_s^3(x, y, z) \rightarrow R^3(x, y, z)))
 \end{array}
 \left. \begin{array}{l}
 \} \\
 \} \\
 \} \\
 \}
 \end{array}
 \begin{array}{l}
 \text{Specification of P1} \\
 \text{Specification of P2} \\
 \text{Specification of P3}
 \end{array}$$

- ▶ $\mathbf{K}_i\phi$ can be interpreted as ϕ is known by peer P_i
- ▶ A tuple \bar{t} is a peer consistent answer to a query Q posed to peer P_i if $\mathbf{K}_iQ(\bar{t})$ is a logical consequence of the epistemic theory
- ▶ **Pros**: semantics can be applied in the presence of cycles
- ▶ **Cons**: requires (possible massive and complex) reasoning by peer P₁
 - ▷ Requires data, mappings and DEC's not only of neighbors, but of all accessible peers

Our approach can be easily adapted so that each peer is a data integration system

Final remarks

We have provided:

1. A semantics for a peer data exchange system which
 - ▶ Respects the modularity and independence of the different peers
 - ▶ Takes trust relationships into consideration
 - ▶ Uses *null* to repair referential DEC's and IC's considering the same semantics of satisfaction of constraints as commercial DBMS's
2. A solution program with answer set programming which can be used to obtain the peer consistent answers
3. Each peer has a single and fixed facts-free logic program, for all queries

Only facts depend on query and other peers' data