Carleton
UNIVERSITY

Information Agents in a Peer Data Exchange System

Leopoldo Bertossi

Carleton University
Ottawa, Canada

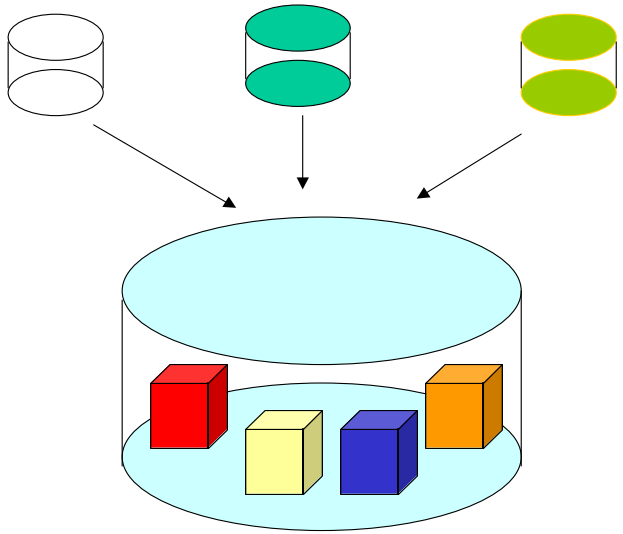
Join work with: **Loreto Bravo, Universidad de Concepcion**

Some Forms of Data Integration

There are different approaches and paradigms for data integration

- Materialized: physical repository is created
- Mediated: data stay at the sources, a virtual integration system is created
- Federated and cooperative: DBMSs are coordinated to collaborate
- Exchange: Data is exported from one system to another
- Peer-to-Peer: Many peers exchange data without a central control mechanism
- ...

Materialized Approaches



A new physical database is created importing data from other data sources

Usually in one direction only

Data sources are independent and autonomous

Data in the new database may be structured differently

E.g. relational tables at the sources, but "cubes" in the new DB

Cubes suggesting different dimensions of data, that give context to (usually) numerical data

Possible a collection of materialized views defined on the combination of the original data sources

Most common incarnation of the materialized approach: Data Warehouses (DWHs)

Issues (among others):

- What views to define and materialize considering future queries?

- How to move data from the sources to the DWH?

  How to populate the DWH?

  ETL tools: Extract, Transport, Load ...

  At this stage: data cleaning and data reconciliation

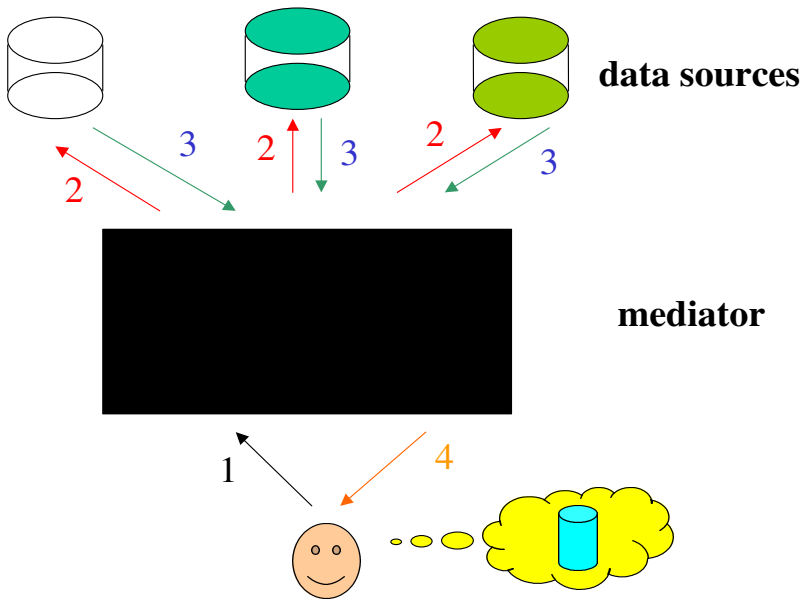
- How and when to update the views?

  Incremental better ...

DWH can be done with relational technology

The multidimensional view of data can be captured using suitable relational schemas

Mediator-Based Data Integration



Collection of data sources that are independent and autonomous

A virtual database is created which is accessed via a *mediator*

A software system that produces the illusion of being interacting with a real database

Queries are posed and answered via the mediator

Mediator: At query (or run) time, produces a query plan

- Detects relevant data at the sources
- Collects data from sources
- Combines the received answers from the sources to build up the final answer

For all this, the mediator must contain mappings between local schemas and its global schema

Mediator is not used or conceived to update data at the sources

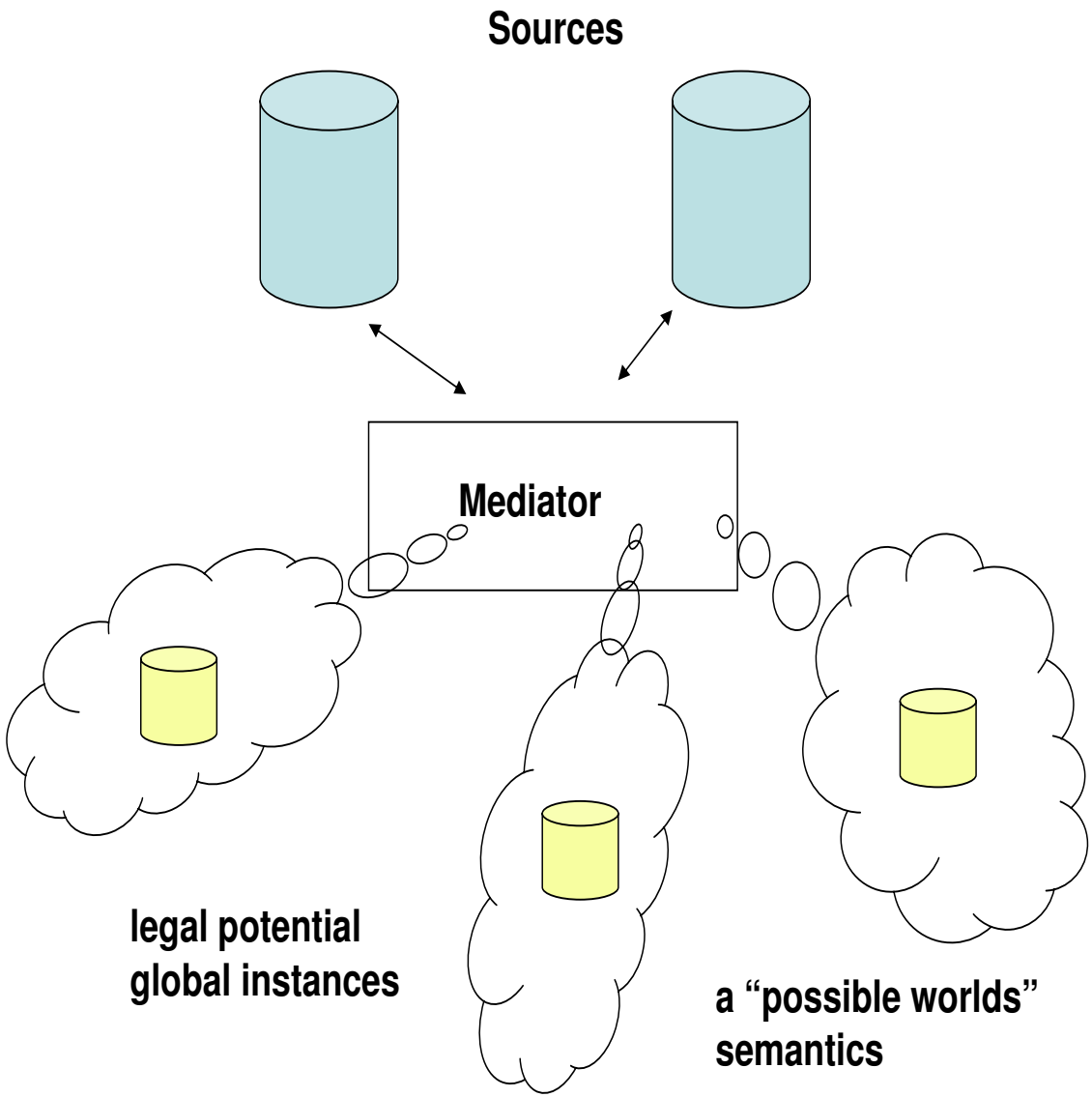
Important Question: What is the semantics of such a system?

More specifically, if we get answers to queries from the integration system, what are the correct, intended answers?

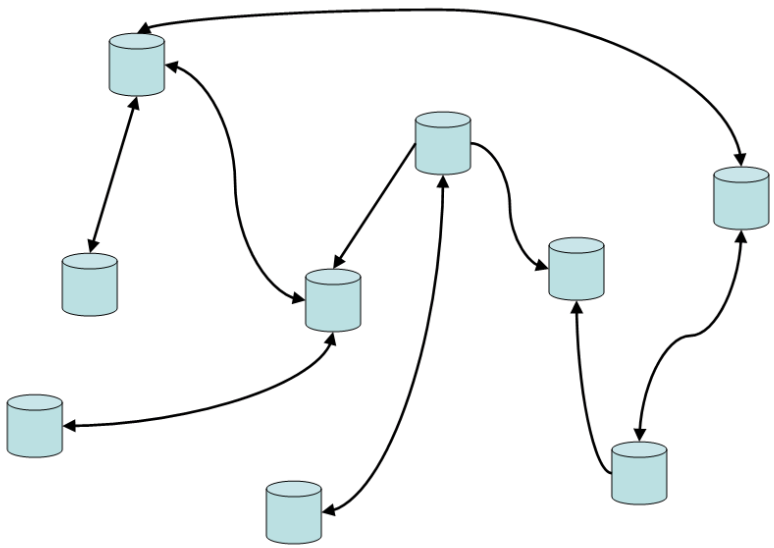
The semantics is given in terms of a collection of legal and intended database instances over the global schema

What is true of the system is what is true of all those instances

This is a possible world semantics


Sources
Mediator
legal potential
global instances
a “possible worlds”
semantics

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

Peers have disjoint schemas

A peer does not update its physical instance according to its DECs and other peers' instances

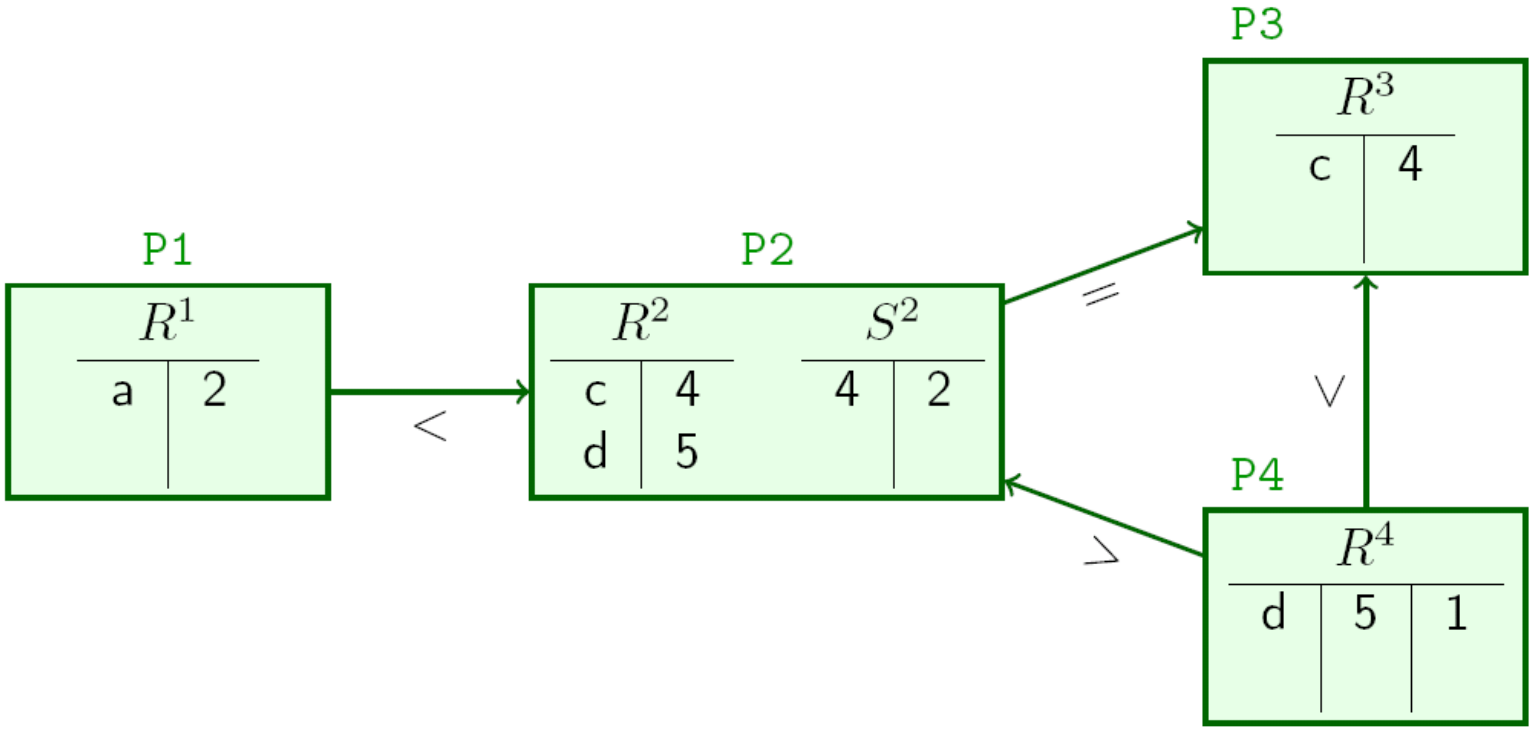
However, if a peer P is answering a (local) query $Q_{\mathtt{P}}$, it may, at query time:

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

All this depending upon its own DECs 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:

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

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

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

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

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 $\mathcal{R}(\mathrm{P})$ and local integrity constraints (ICs)
  - ▷ its set $\Sigma(\mathrm{P}) = \bigcup_{\mathrm{P'} \in \mathcal{P}} \Sigma(\mathrm{P}, \mathrm{P'})$ 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 DECs 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* (UDEC) between peers P1, P2:

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

  $R_i, Q_j \in \mathcal{R}(\mathtt{P1}) \cup \mathcal{R}(\mathtt{P2})$, 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:

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

  $R, Q \in \mathcal{R}(\mathtt{P1}) \cup \mathcal{R}(\mathtt{P2})$, 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

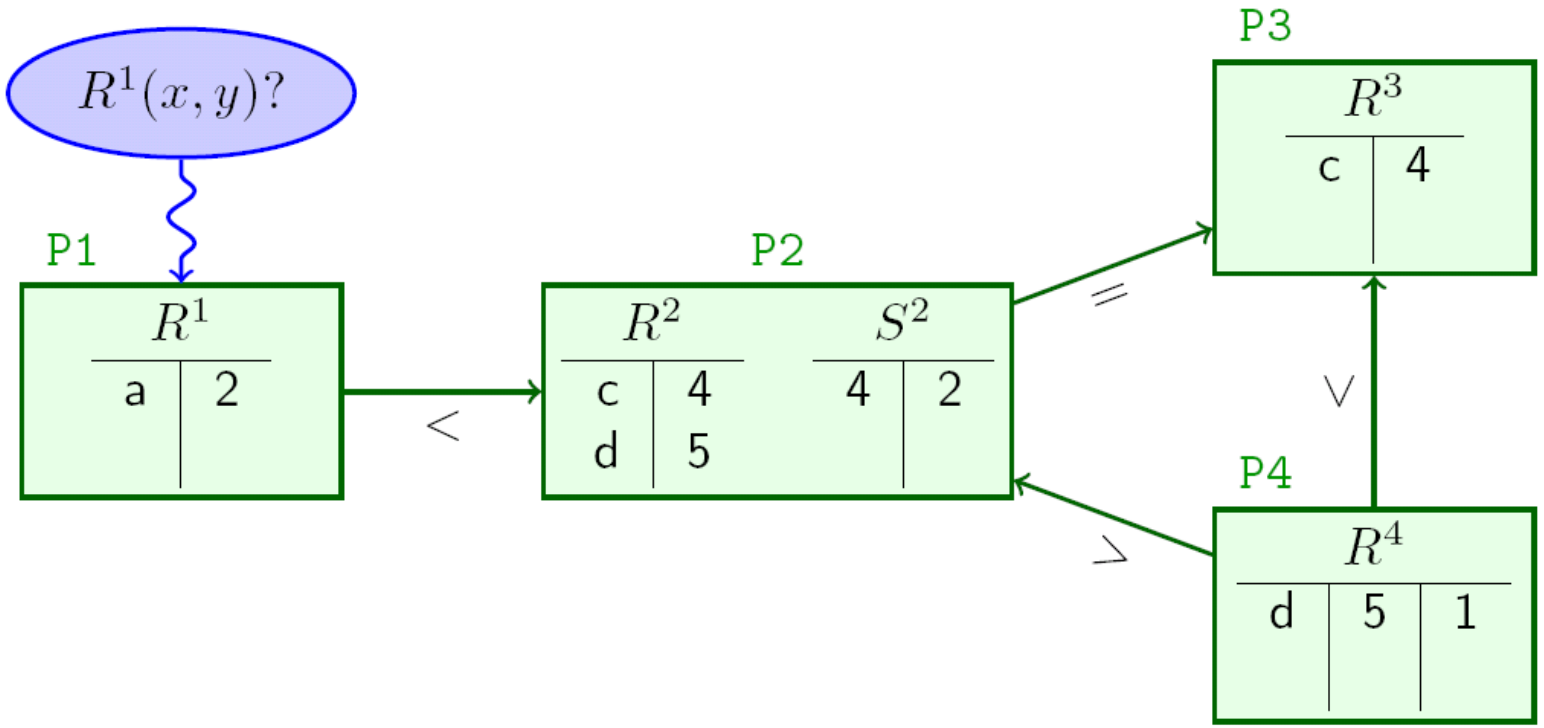
- What did P receive from a neighbor P'?

  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

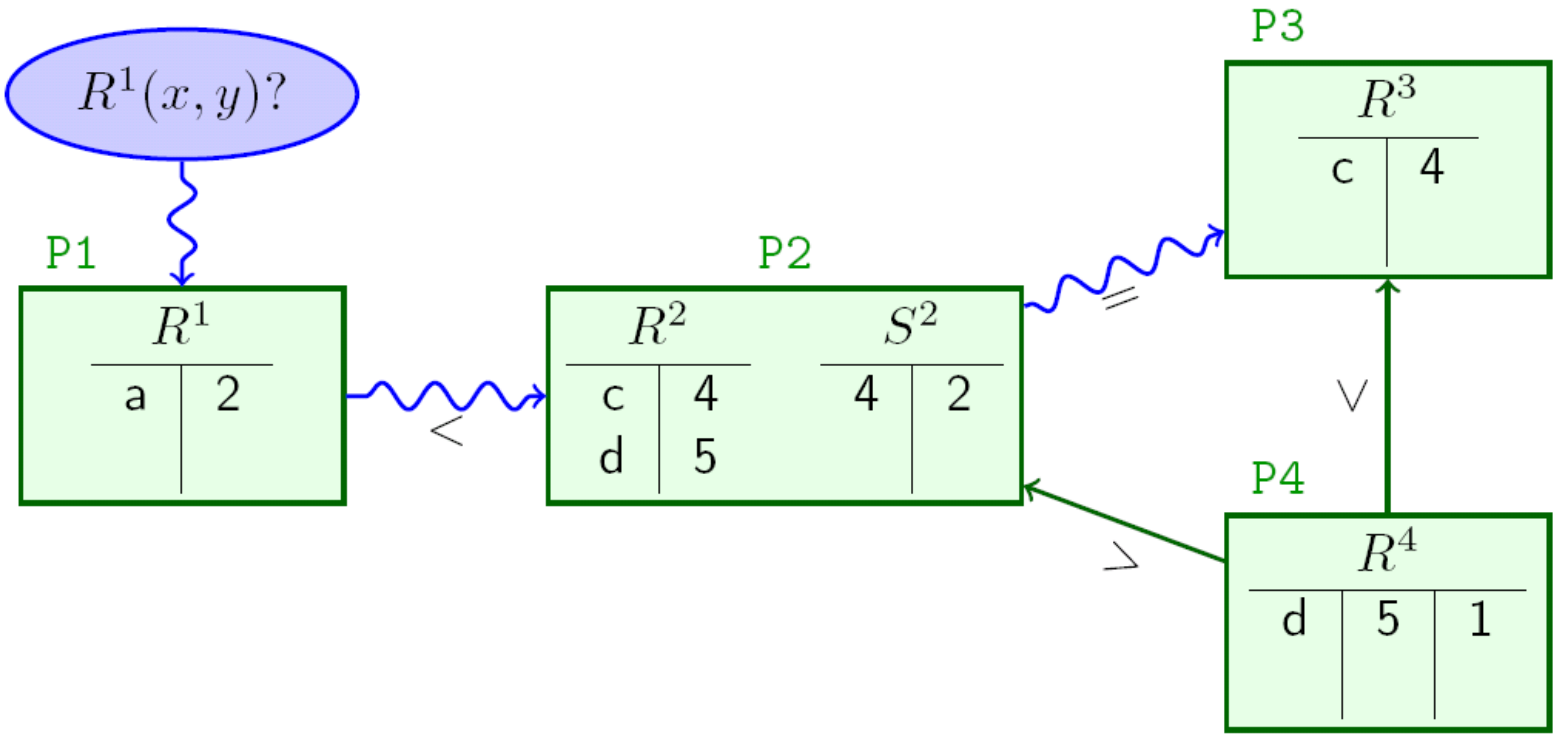
Idea, Intuitions, "Operational" Semantics



DECs:

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

$$\left.\begin{array}{ll} \Sigma(\mathsf{P4},\mathsf{P3}) = & \{\forall x \forall y (R^3(x,y) \rightarrow \exists z R^4(x,y,z))\} \end{array}\right\} \text{RDEC}$$

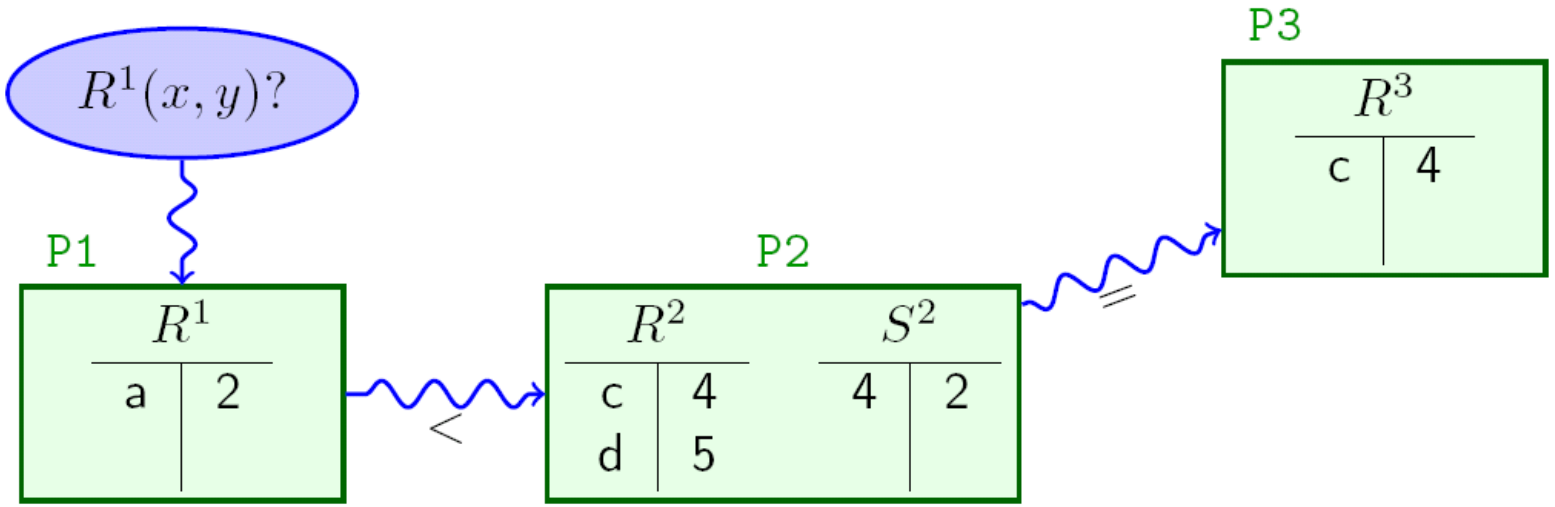


DECs:

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

$$\left.\begin{array}{ll}\Sigma(\mathtt{P4},\mathtt{P3}) = & \{\forall x\forall y(R^3(x,y) \rightarrow \exists z R^4(x,y,z))\}\end{array}\right\} \text{RDEC}$$

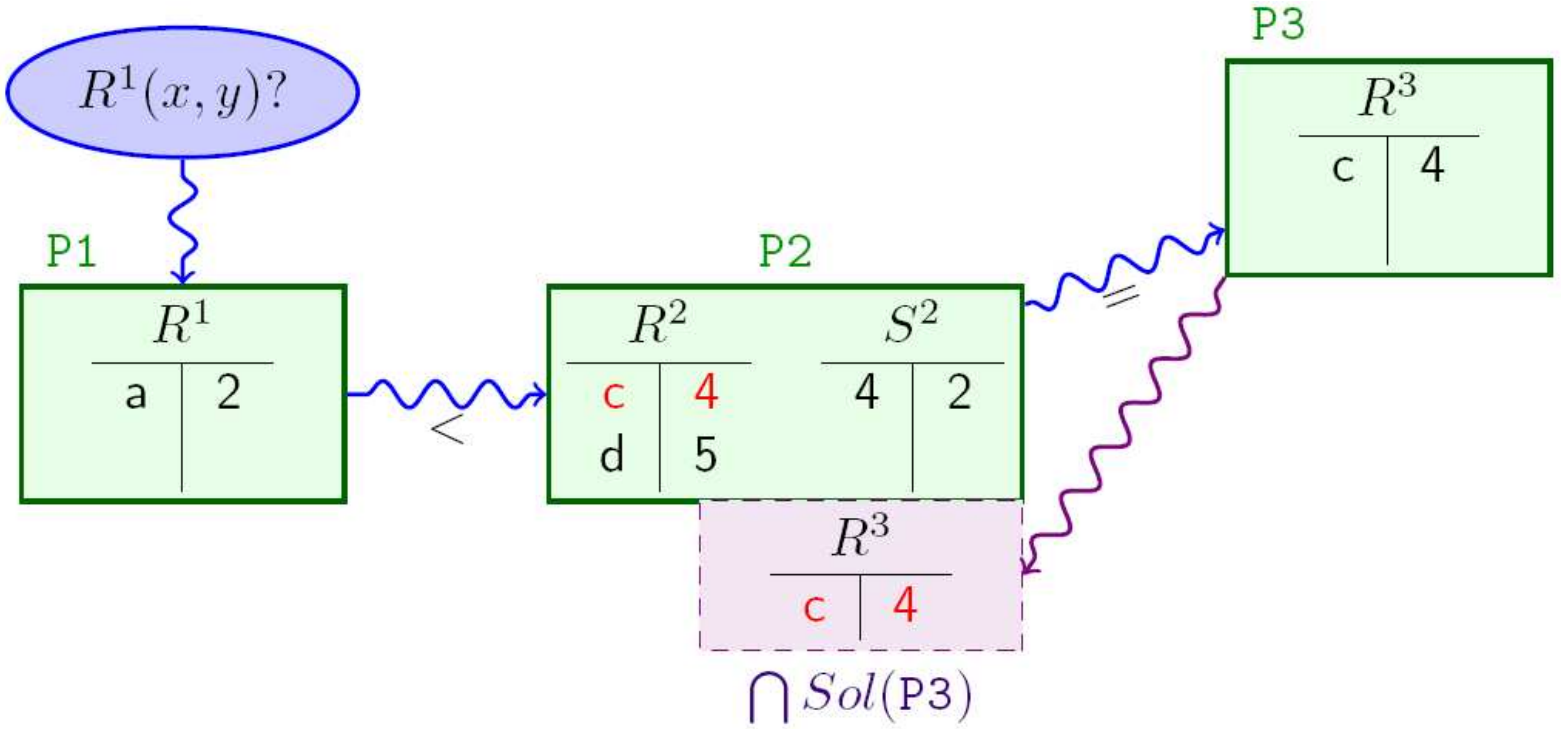
Peer P4 will have no effect on the query!



DECs:

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

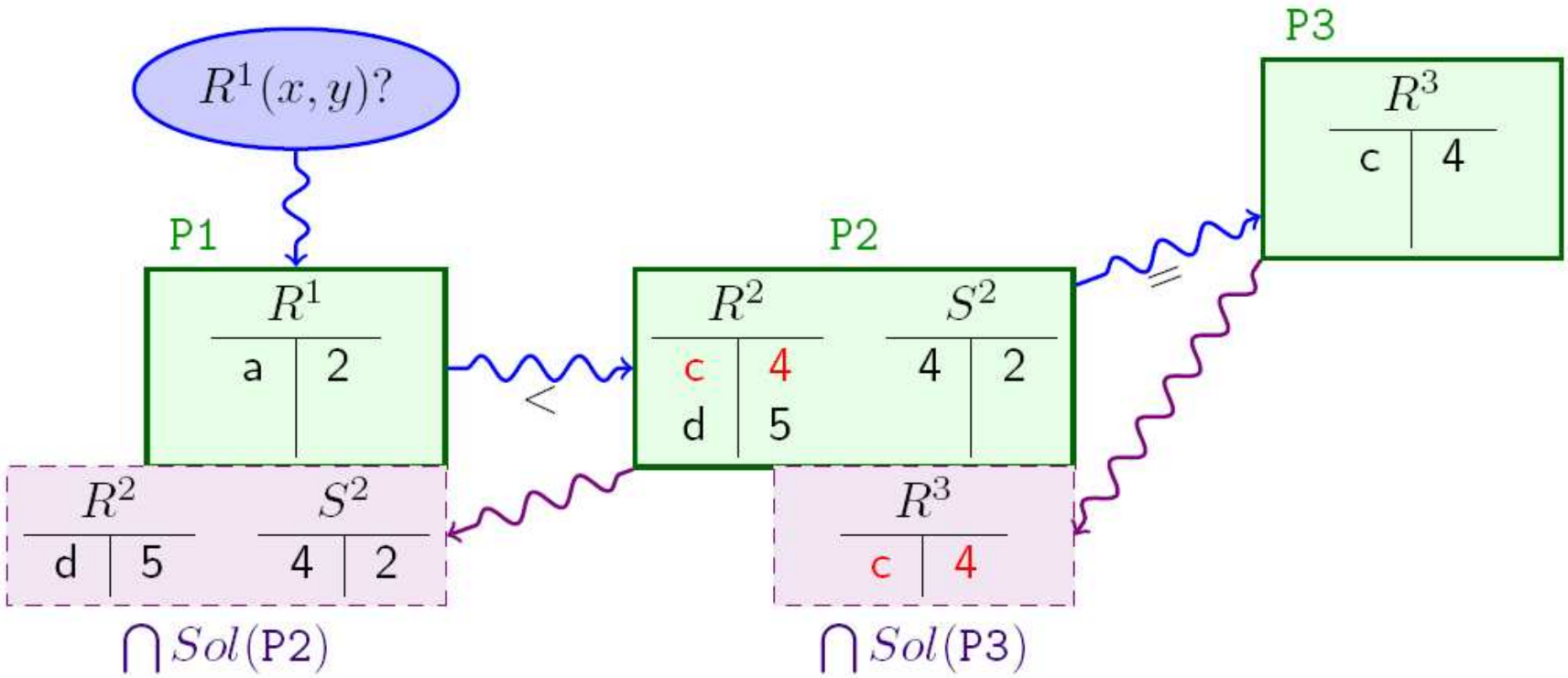
$\Sigma(\mathsf{P2},\mathsf{P3}) = \{\forall x \forall y (R^2(x,y) \wedge R^3(x,y) \rightarrow \mathbf{false})\}$ (a denial DEC)



DECs:

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

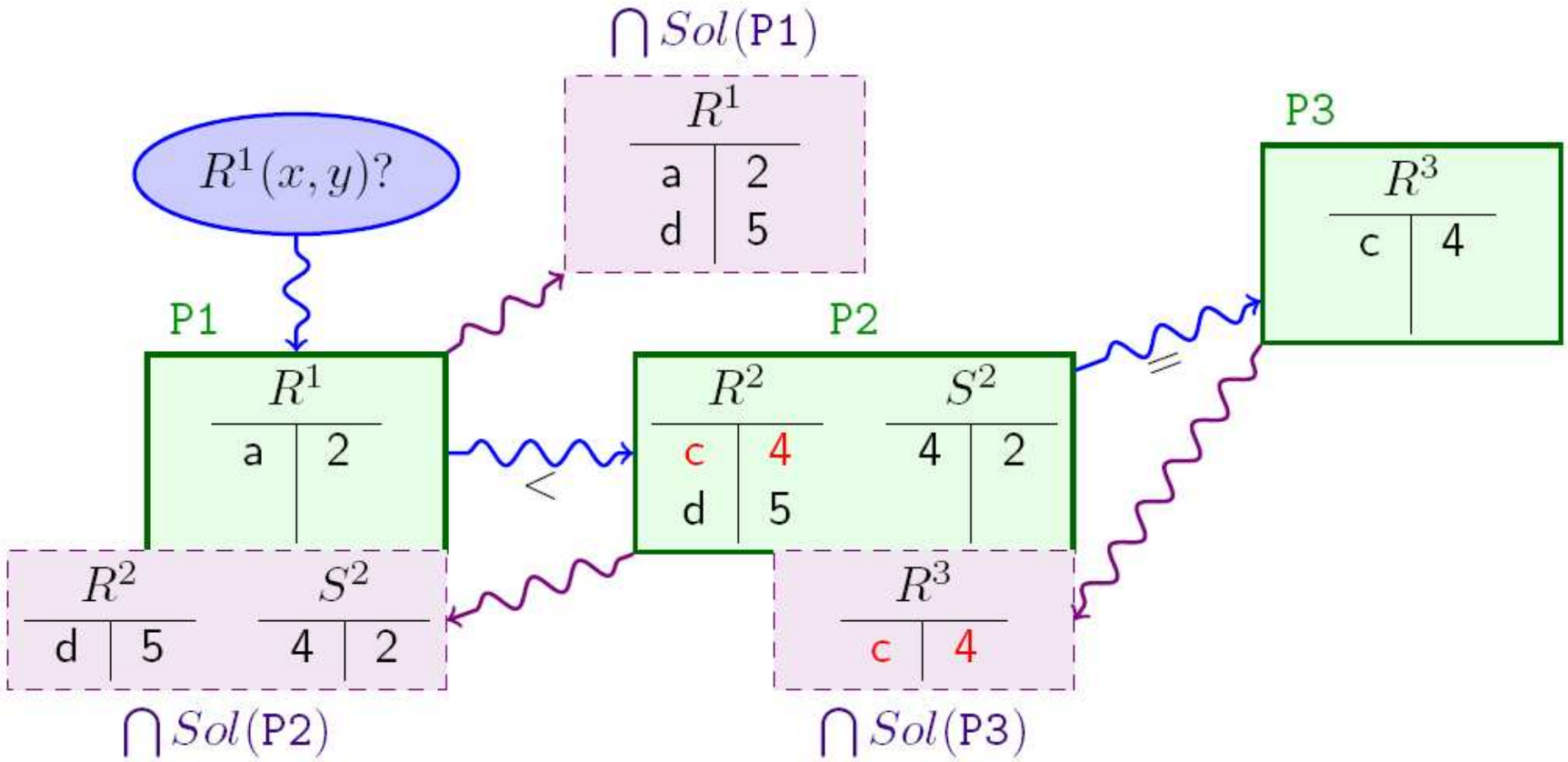
$\Sigma(\mathtt{P2},\mathtt{P3}) = \{\forall x \forall y (R^2(x,y) \wedge R^3(x,y) \rightarrow \mathbf{false})\}$



DECs:

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

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



DECs:

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

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

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

In this example, a peer passes a complete certain instance back to a neighbor, e.g. P2 to P1

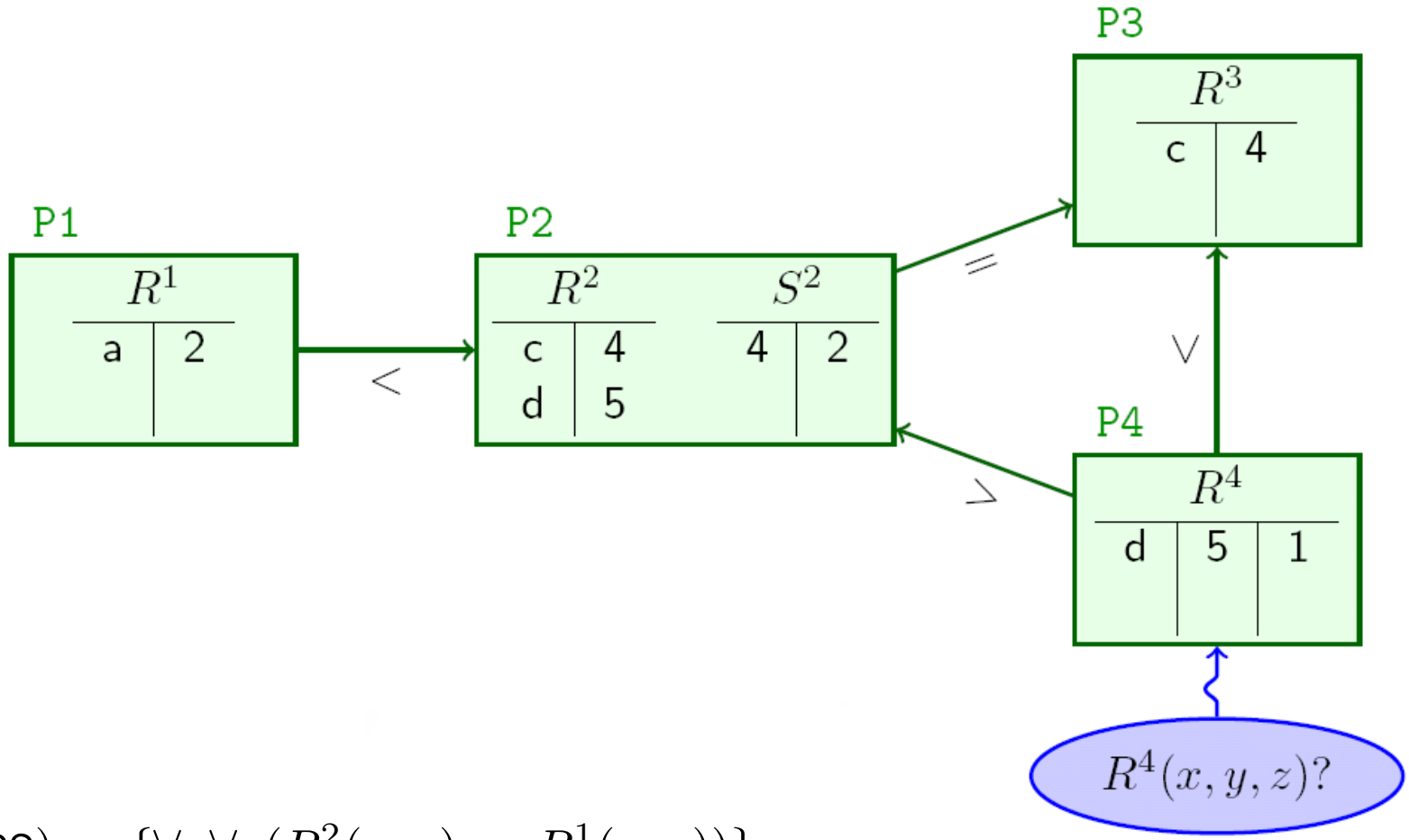
This may be more than what P1 needs to answer the original query, e.g. P1 does not need S_2 from P2

We have done things this way to show the issues behind the semantics of the system, and of each peer in particular

Each peer will have a set of local solution instances, and this set is not determined by a particular query

In terms of data movement, many things can be optimized in comparison with the example above

Example:



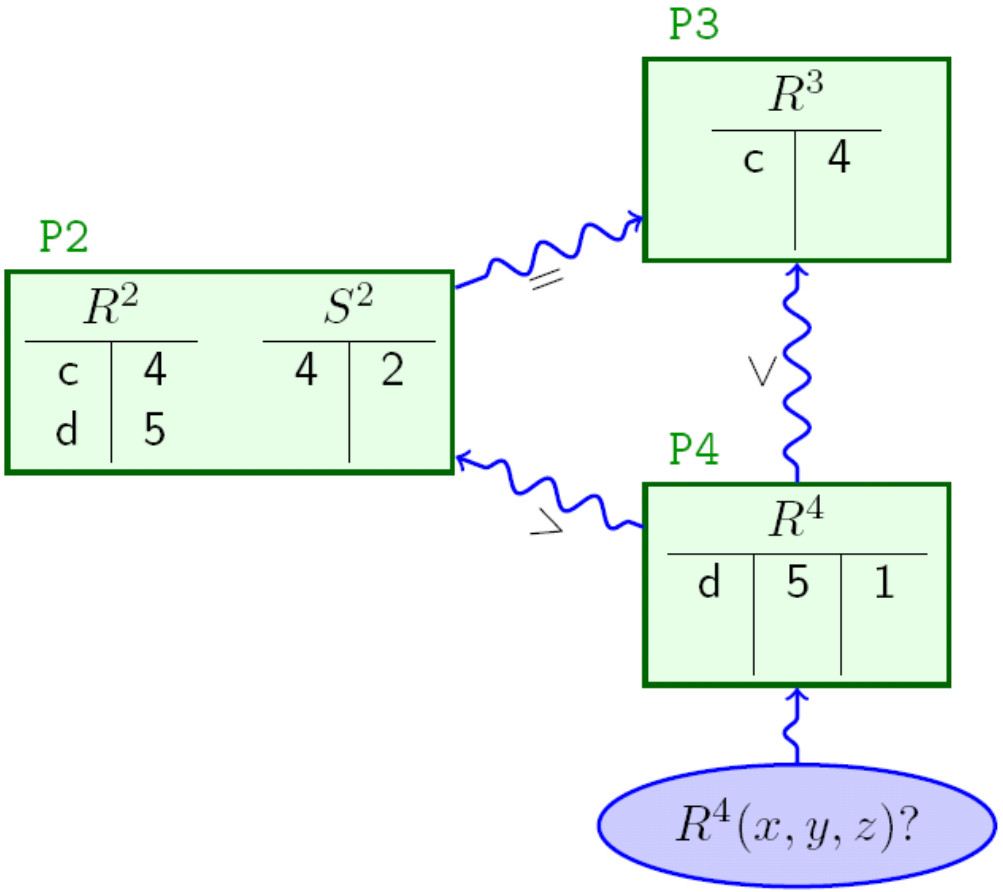
DECs:

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

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

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

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

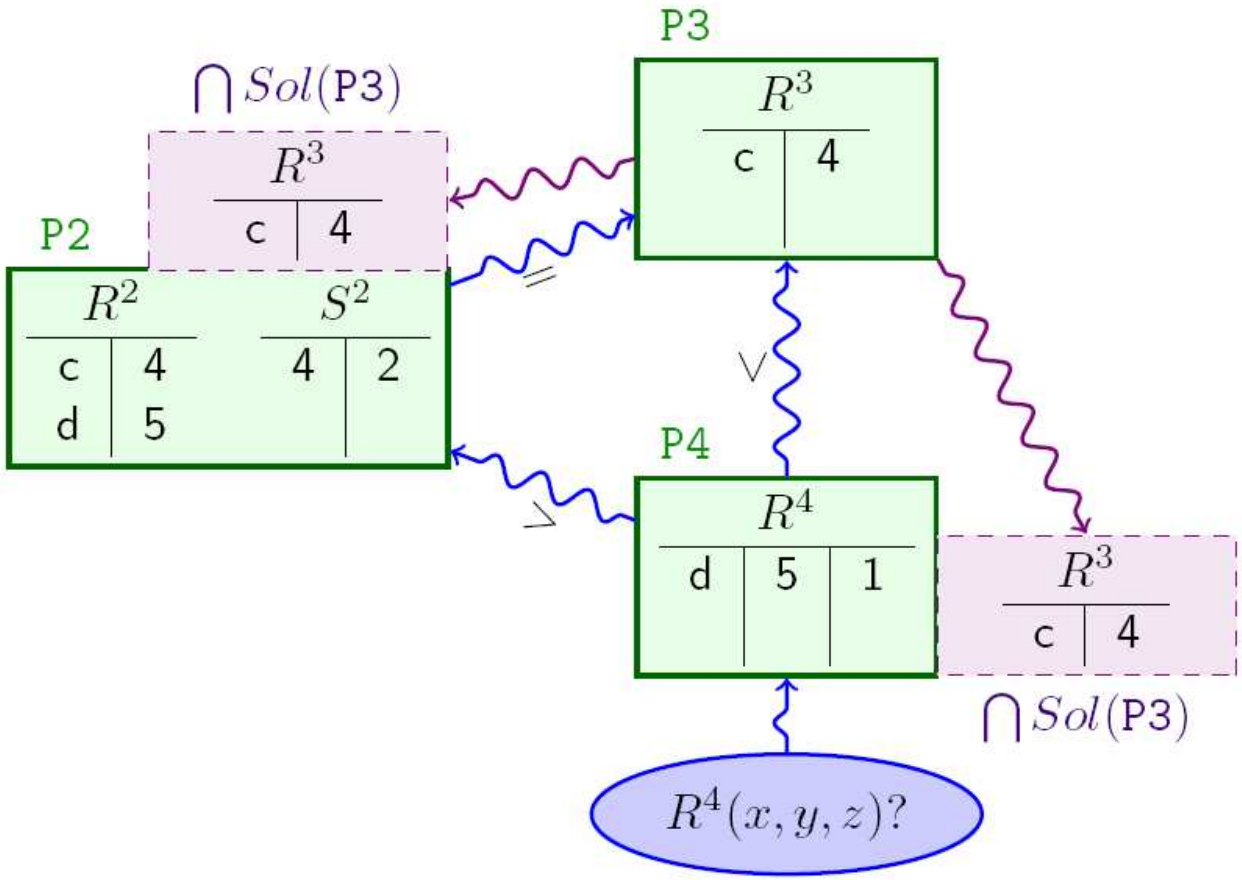


DECs:

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

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

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

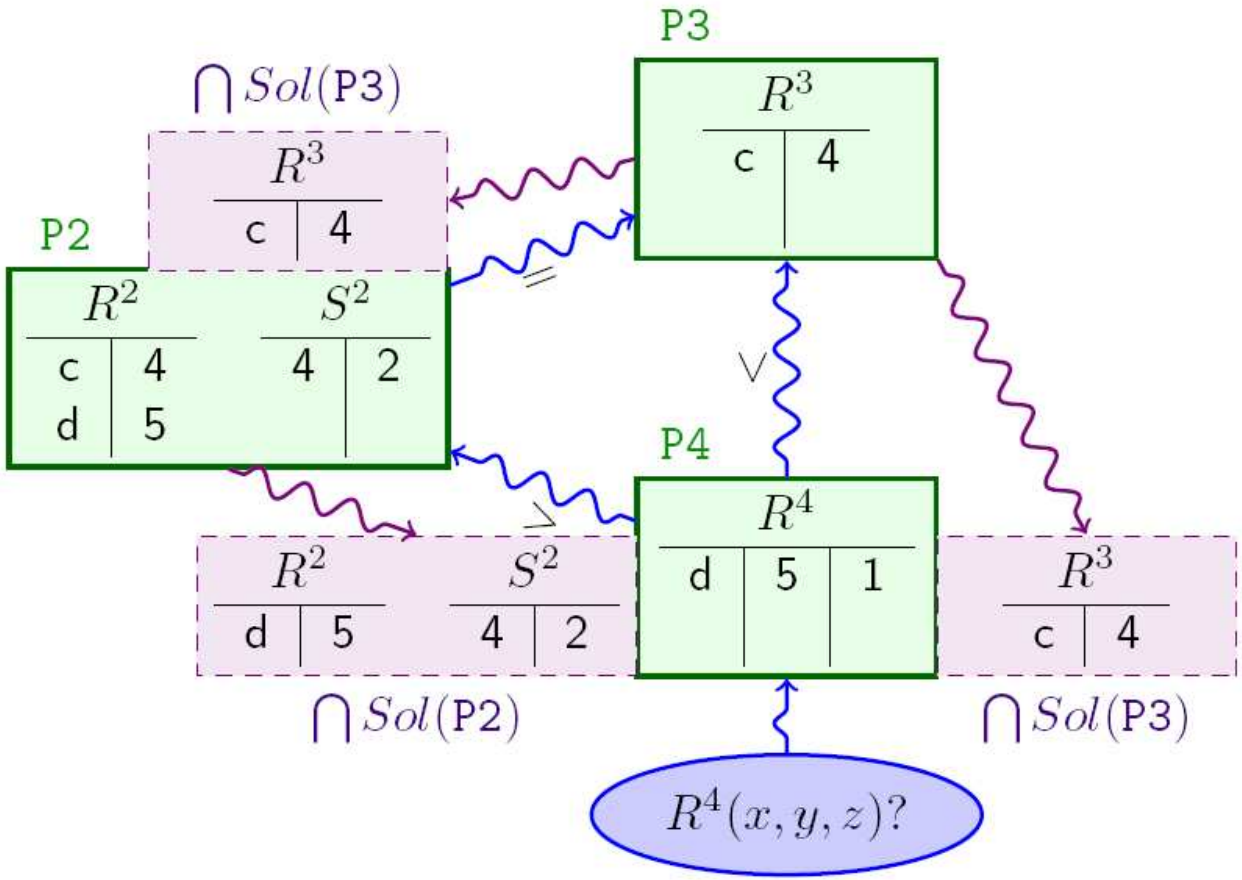


DECs:

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

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

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

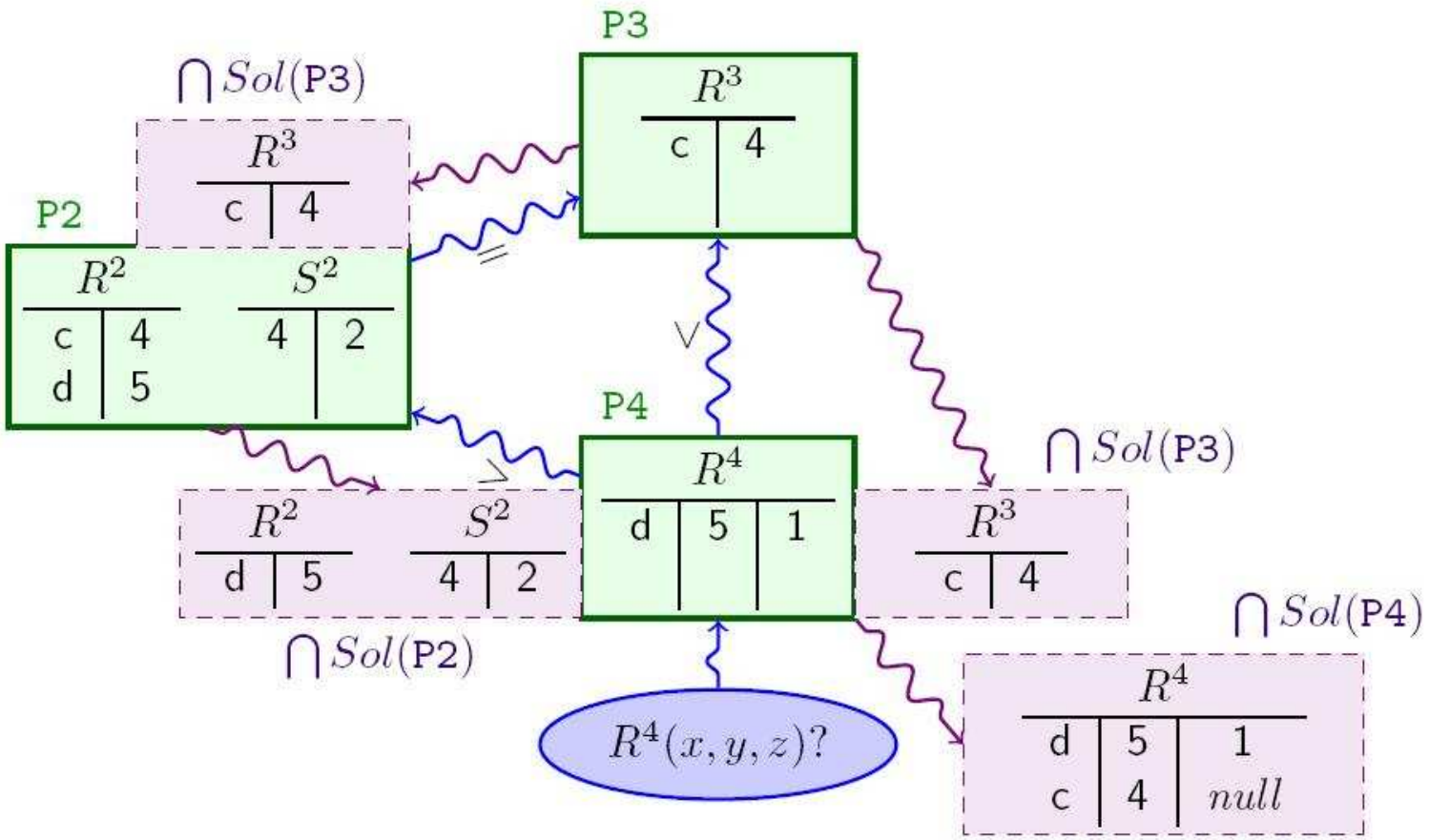


DECs:

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

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

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



DECs:

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

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

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

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

- Incomplete information is represented by means of null values, which 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 DECs via joins (existential quantification over a join variable); cf. page 16

  We can extend RDECs with more complex antecedents of the implication

- Other semantics for incomplete databases (and null values) could be easily adopted in our framework, being able to deal with more complex consequents in DECs

- It is also possible to impose local ICs on peers' instances

  They can be uniformly handled as before by means of DECs of the form $\Sigma(\mathtt{P}, \mathtt{P})$ and an $=$-trust relationship

What follows?

1. A logic-based (model-based) semantics

2. Capturing peers' solutions as models of logic programs

Formal Semantics

We define now the solution instances of a peer; in two steps

1. First locally for a peer and its neighbors

2. Using 1., we recursively consider transitive relations to other peers

We start from a peer P who has a local instance $D(P)$, which was somehow extended to an instance D over the union of its schema and those of its neighbors

D may not satisfy the DECs from P to its neighbors, and inconsistencies have to be solved, minimally ...

Let D' be an instance for the union schema of P and its neighbors

When is D' a solution for D?

D' is a *neighborhood solution for* $\mathtt{P}$ *and* D:

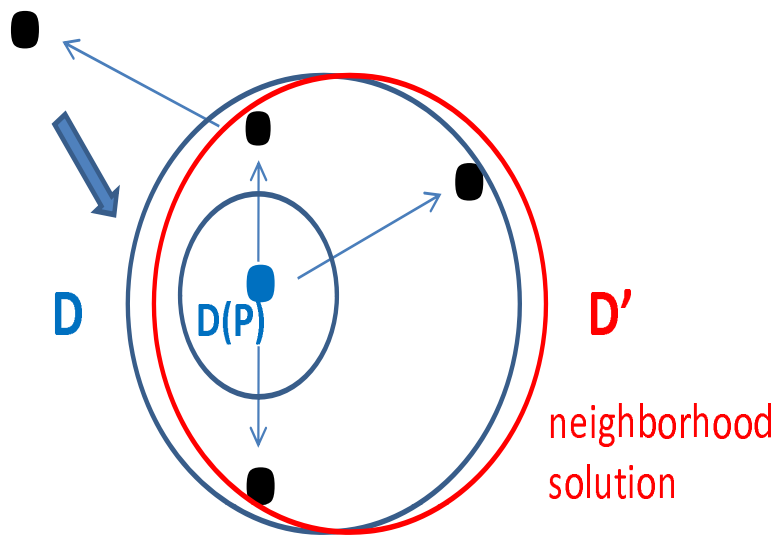
1. [satisfaction] D' satisfies the DECs and local constraints of $\mathtt{P}$, i.e. $\Sigma(\mathtt{P},\mathtt{P}) \cup \bigcup_{\mathtt{Pi} \in N(\mathtt{P})} \Sigma(\mathtt{P},\mathtt{Pi})$

2. [trust] The data in D' associated to the peers that $\mathtt{P}$ 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)

There may be more than one neighborhood solution D'



Now we define the solution instances (solutions) for P

Now consider transitive relationships too

Transitive peers will contribute to the creation of the D above; which will lead to the D's

Transitive peers will contribute to D with the intersection of their own solutions, recursively

Formally ...

Let $D(\mathtt{P})$ be the database instance for peer $\mathtt{P}$

A solution for $\mathtt{P}$ can be recursively defined as:

- If $\mathtt{P}$ has no DECs, then the solution is $D(\mathtt{P})$
- Otherwise:

  1. Let D be a database instance containing the union of

     ▷ $D(\mathtt{P})$
     ▷ for each neighboring peer, the intersection of its solutions

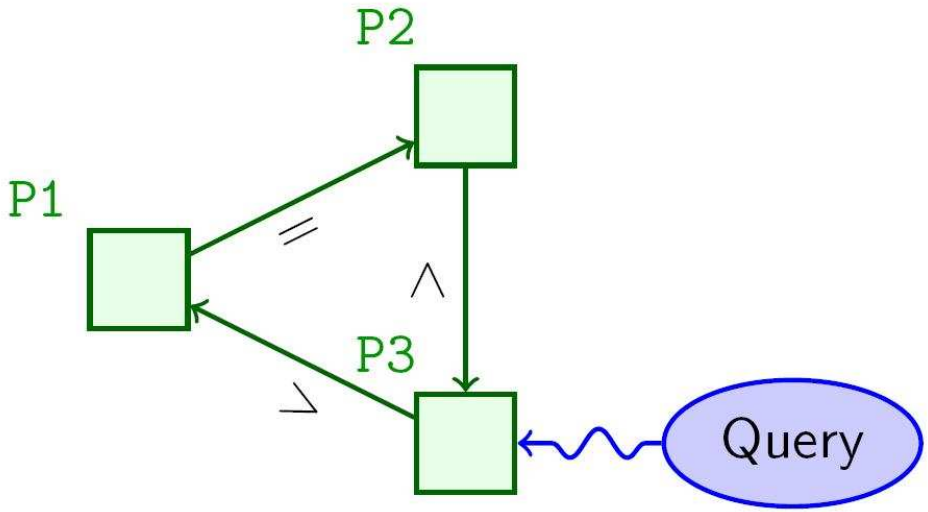
     (an instance over the union schema of the neighborhood around $\mathtt{P}$)

  2. Let D' be a neighborhood solution for $\mathtt{P}$ and D

  Then, D' restricted to the schema of $\mathtt{P}$ is a solution for $\mathtt{P}$

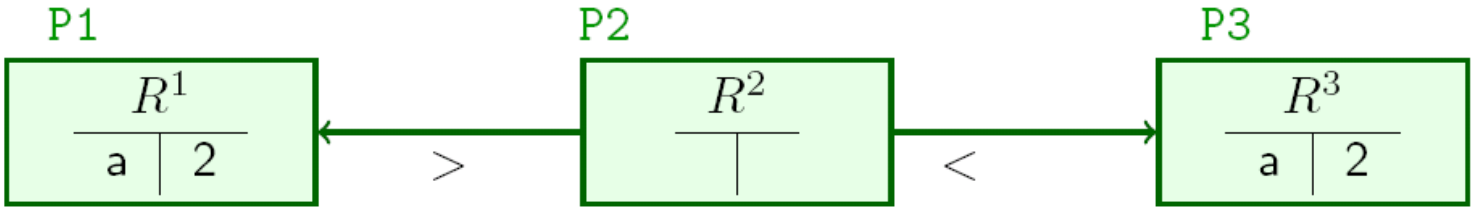
For this definition to work, we restrict ourselves to acyclic peer data exchange systems, i.e. graphs of neighbors is acyclic (not necessarily the DECs)

We also want 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 $\Rightarrow \cdots$
- 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:

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

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

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

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

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

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

Our null-based repair semnatics allow for decidability

Even with acyclic neighbors' graph acyclic there could be undecidability depending on interaction between DECs 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, DECs 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

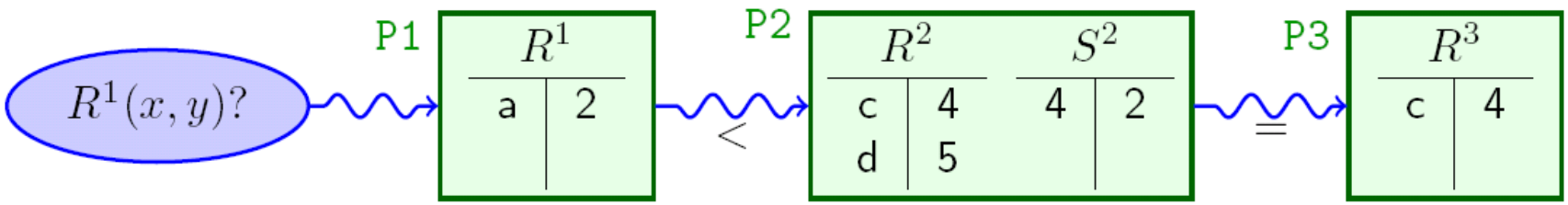
- The program uses annotation constants to indicate the atoms that may virtually inserted or deleted in order to restore consistency:

Annotation	Atom	The tuple $P(\bar{a})$ is ...
$\mathbf{t}$	$P\_(\bar{a}, \mathbf{t})$	made true (inserted)
$\mathbf{f}$	$P\_(\bar{a}, \mathbf{f})$	made false (deleted)
$\mathbf{t}^\star$	$P\_(\bar{a}, \mathbf{t}^\star)$	true or becomes true
$\mathbf{f}^\star$	$P\_(\bar{a}, \mathbf{f}^\star)$	false or becomes false
$\mathbf{t}^{\star\star}$	$P\_(\bar{a}, \mathbf{t}^{\star\star})$	true in the solution

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

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

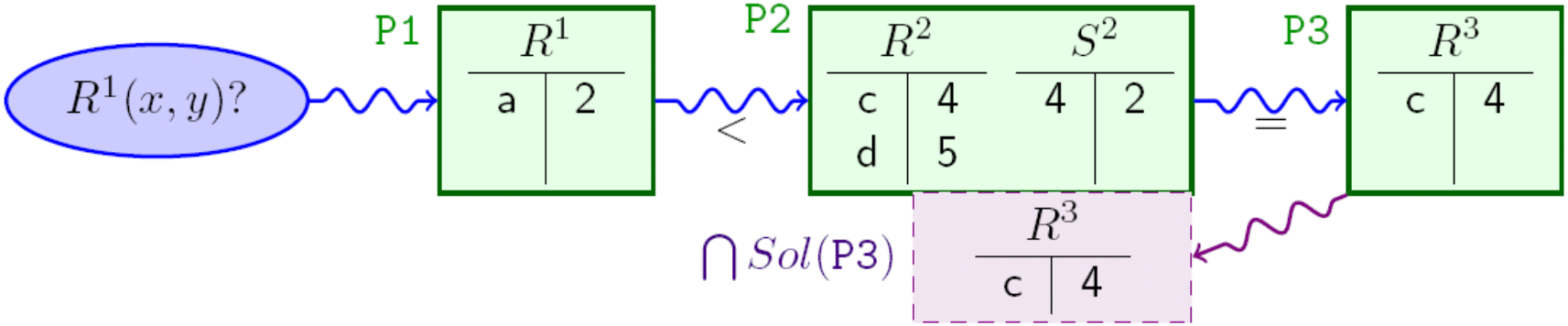
$\Sigma(\mathtt{P2}, \mathtt{P3}) = \{\forall x \forall y (R^2(x,y) \wedge R^3(x,y) \rightarrow \mathbf{false})\}$



- Peer P3 has no DECs, therefore its only solution is $D(\mathtt{P3})$

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

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



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

$dom(a).\quad dom(c).\quad \ldots$

$R^3(c,4)\quad R^2(c,4)\quad R^2(d,5)\quad S^2(4,2)$

$R^3_{\_}(x,y,\mathbf{f}) \vee R^2_{\_}(x,y,\mathbf{f}) \leftarrow R^2_{\_}(x,y,\mathbf{t}^\star),\ R^3_{\_}(x,y,\mathbf{t}^\star), x \neq null, y \neq null.$

$R^3_{\_}(x,y,\mathbf{t}^\star) \leftarrow R^3_{\_}(x,y,\mathbf{t}).$

$R^3_{\_}(x,y,\mathbf{t}^\star) \leftarrow R^3(x,y).$

$R^3_{\_}(x,y,\mathbf{f}^\star) \leftarrow R^3_{\_}(x,y,\mathbf{f}).$

$R^3_{\_}(x,y,\mathbf{f}^\star) \leftarrow dom(x), dom(y),\ not\ R^3(x,y).$

$R^2_{\_}(x,y,\mathbf{t}^{\star\star}) \leftarrow R^2_{\_}(x,y,\mathbf{t}^\star),\ not\ R^2_{\_}(x,y,\mathbf{f}).$

$\leftarrow R^3_{\_}(x,y,\mathbf{t}), R^3_{\_}(x,y,\mathbf{f}).$

} (Similarly for R^2)

$dom(a).\quad dom(c).\quad \ldots$

$R^3(c,4)\quad R^2(c,4)\quad R^2(d,5)\quad S^2(4,2)$

$R^3_{\_}(x,y,\mathbf{f}) \vee R^2_{\_}(x,y,\mathbf{f}) \leftarrow R^2_{\_}(x,y,\mathbf{t}^\star),\ R^3_{\_}(x,y,\mathbf{t}^\star), x \neq null, y \neq null.$

$R^3_{\_}(x,y,\mathbf{t}^\star) \leftarrow R^3_{\_}(x,y,\mathbf{t}).$
$R^3_{\_}(x,y,\mathbf{t}^\star) \leftarrow R^3(x,y).$
$R^3_{\_}(x,y,\mathbf{f}^\star) \leftarrow R^3_{\_}(x,y,\mathbf{f}).$
$R^3_{\_}(x,y,\mathbf{f}^\star) \leftarrow dom(x), dom(y),\ not\ R^3(x,y).$
$R^2_{\_}(x,y,\mathbf{t}^{\star\star}) \leftarrow R^2_{\_}(x,y,\mathbf{t}^\star),\ not\ R^2_{\_}(x,y,\mathbf{f}).$
$\leftarrow R^3_{\_}(x,y,\mathbf{t}), R^3_{\_}(x,y,\mathbf{f}).$

} (Similarly for R^2)

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

D_1 :

R^2	
d	5

S^2	
4	2

R^3	
c	4

D_2 :

R^2	
c	4
d	5

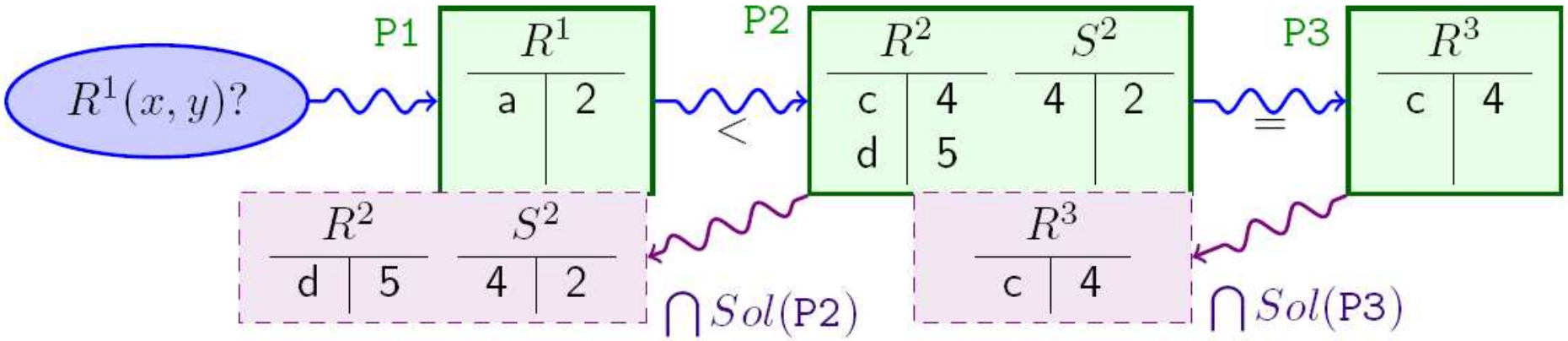
S^2	
4	2

R^3

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

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

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



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

$dom(a).\quad dom(c).\quad \ldots$

$R^1(a,2)\quad R^2(d,5)\quad S^2(4,2)$

$R^1\_(x,y,\mathbf{t}) \leftarrow R^2\_(x,y,\mathbf{t}^\star),\ R^1\_(x,y,\mathbf{f}^\star), x \neq null, y \neq null.$

$R^1\_(x,y,\mathbf{t}^\star) \leftarrow R^1\_(x,y,\mathbf{t}).$

$R^1\_(x,y,\mathbf{t}^\star) \leftarrow R^1(x,y).$

$R^1\_(x,y,\mathbf{f}^\star) \leftarrow R^1\_(x,y,\mathbf{f}).$

$R^1\_(x,y,\mathbf{f}^\star) \leftarrow dom(x), dom(y),\ not\ R^1(x,y).$

$R^1\_(x,y,\mathbf{t}^{\star\star}) \leftarrow R^1\_(x,y,\mathbf{t}^\star),\ not\ R^1\_(x,y,\mathbf{f}).$

$\leftarrow R^1\_(x,y,\mathbf{t}), R^1\_(x,y,\mathbf{f}).$

} (Similarly for R^2)

$dom(a).\quad dom(c).\quad \ldots$
$R^1(a,2)\quad R^2(d,5)\quad S^2(4,2)$
$R^1_{\_}(x,y,\mathbf{t}) \leftarrow R^2_{\_}(x,y,\mathbf{t}^\star),\ R^1_{\_}(x,y,\mathbf{f}^\star), x \neq null, y \neq null.$
$R^1_{\_}(x,y,\mathbf{t}^\star) \leftarrow R^1_{\_}(x,y,\mathbf{t}).$
$R^1_{\_}(x,y,\mathbf{t}^\star) \leftarrow R^1(x,y).$
$R^1_{\_}(x,y,\mathbf{f}^\star) \leftarrow R^1_{\_}(x,y,\mathbf{f}).$
$R^1_{\_}(x,y,\mathbf{f}^\star) \leftarrow dom(x), dom(y),\ not\ R^1(x,y).$
$R^1_{\_}(x,y,\mathbf{t}^{\star\star}) \leftarrow R^1_{\_}(x,y,\mathbf{t}^\star),\ not\ R^1_{\_}(x,y,\mathbf{f}).$
$\leftarrow R^1_{\_}(x,y,\mathbf{t}), R^1_{\_}(x,y,\mathbf{f}).$

(Similarly for R^2)

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

D_1 :

R^1	
a	2
d	5

R^2	
d	5

S^2	
4	2

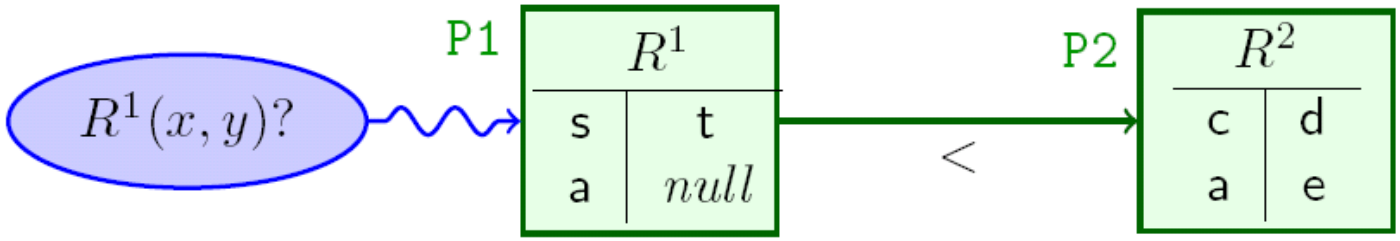
- The query $R^1(x,y)?$ can also be added to the answer set program:

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

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

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

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



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

The solution obtained from the answer set is:

R^1	
s	t
a	*null*
c	*null*

- A set of DECs and ICs is Ref-acyclic if there is no cycles through referential DECs or ICs

- For example:
  - ▷ Ref-acyclic

$$\begin{aligned}\Sigma(\mathtt{P1},\mathtt{P2}) = \{&\forall xy\ (R^1(x,y) \rightarrow R^2(x,y)),\\ &\forall xy\ (R^2(x,y) \rightarrow R^1(x,y))\}\\ \Sigma(\mathtt{P2},\mathtt{P1}) = \{&\forall x(S^2(x) \rightarrow \exists y S^1(x,y))\}\end{aligned}$$

  - ▷ Not ref-acyclic

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

Theorem: for a ref-acyclic set of DECs and ICs, there is a one-to-one correspondence between answer sets and solutions of a peer

Special Case

- Unrestricted Import Case:
  - ▷ The DECs 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

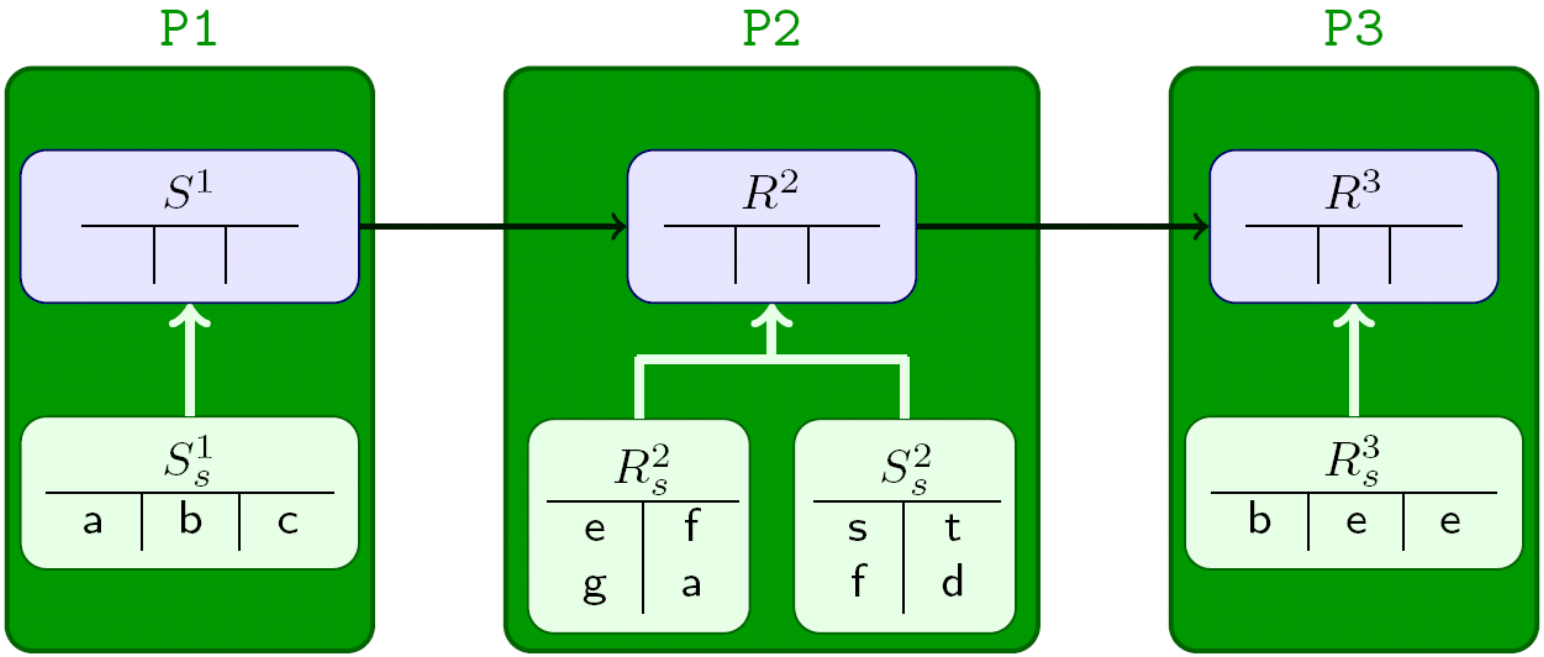
  For example when:
  - ▷ The cycles in the graph are not relevant to the query
  - ▷ Even if the DECs and ICs are not ref-acyclic, depending on the interaction with the trust relationships, the solution-program can provide exactly the set of solutions

- Instead of requesting all the data of the neighboring sources:
  - ▷ restrict to the data that is relevant to check the DECs 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
- DECs are of the form: $cq_i \rightarrow cq_j$
  where cq_i and cq_j are conjunctive queries over `Pi` and `Pj`'s schemas, resp.
- No trust relationships
  - ▷ Implicitly: peers trust themselves less than other peers
- Local ICs violations are avoided
  - ▷ A peer that is inconsistent wrt its local ICs 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:
  $\forall xyz(S^1_s(x,y,z) \rightarrow S^1(x,y,z))$
  $\forall xyz(R^2_s(x,y) \wedge R^2_s(y,z) \rightarrow R^2(x,y,z))$
  $\forall xyz(R^3_s(x,y,z) \rightarrow R^3(x,y,z))$
- DECs:
  $\forall xy(R^2(x,y,z) \rightarrow \exists w R^1(x,y,w))$
  $\forall xy(R^3(x,y,y) \rightarrow \exists uv R^3(u,x,v))$

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

$$\left.\begin{array}{l}\mathbf{K_1}(\forall xyz(S_s^1(x,y,z) \rightarrow S^1(x,y,z))) \\ \forall xy(\mathbf{K_2}(R^2(x,y,z)) \rightarrow \mathbf{K_1}(\exists w R^1(x,y,w)))\end{array}\right\} \text{Specification of P1}$$

$$\left.\begin{array}{l}\mathbf{K_2}(\forall xyz(R_s^2(x,y) \wedge R_s^2(y,z) \rightarrow R^2(x,y,z))) \\ \forall xy(\mathbf{K_3}(R^3(x,y,y)) \rightarrow \mathbf{K_2}(\exists uv R^3(u,x,v)))\}\end{array}\right\} \text{Specification of P2}$$

$$\left.\begin{array}{l}\mathbf{K_3}(\forall xyz(R_s^3(x,y,z) \rightarrow R^3(x,y,z)))\end{array}\right\} \text{Specification of P3}$$

- $\mathbf{K_i}\phi$ can be interpreted as ϕ is known by peer Pi
- A tuple $\bar{t}$ is a peer consistent answer to a query Q posed to peer Pi if $\mathbf{K_i}Q(\bar{t})$ 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 DECs 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 DECs and ICs considering the same semantics of satisfaction of constraints as commercial DBMSs
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