Securing Materialized Views: a Rewriting-Based Approach

Sarah Nait Bahloul, Emmanuel Coquery and Mohand-Saïd Hacid

Université de Lyon, France

First Franco-American Workshop Security

Outline

- Context
- Problem statement
- Related work
- Authorization views
- Rewriting-based approach
- Approach properties
  - Security
  - Termination
  - Maximality
- Conclusion

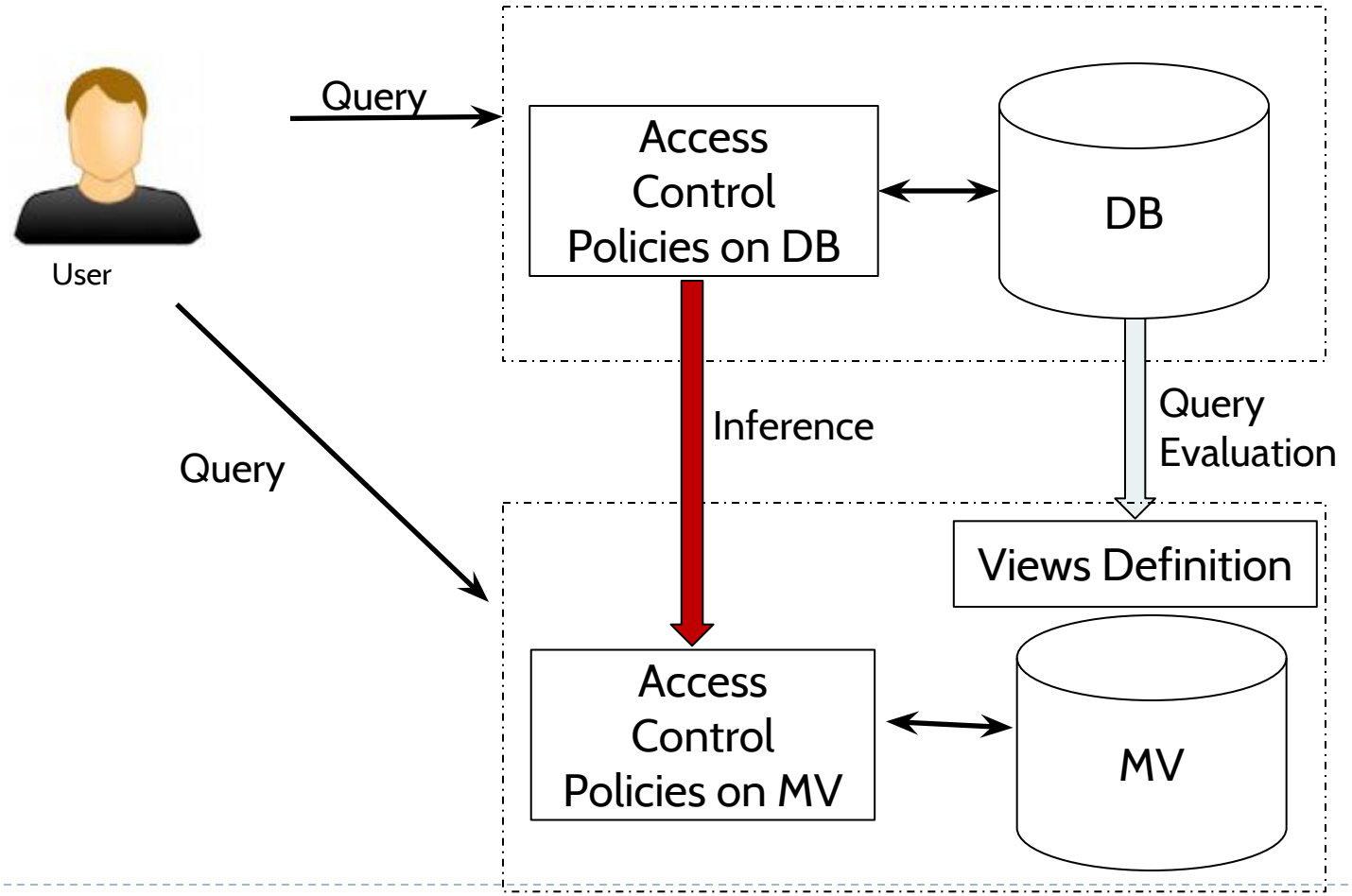
Context

- Data security
  - Confidentiality, Integrity, Availability,...
- Materialized views
  - Used in decision and distributed systems: Data warehouses, Mediators, ...
  - Store the results returned by a query
    - They can be used as any other table.

➔ Ensuring confidentiality of materialized view data is also important.

Problem Statement

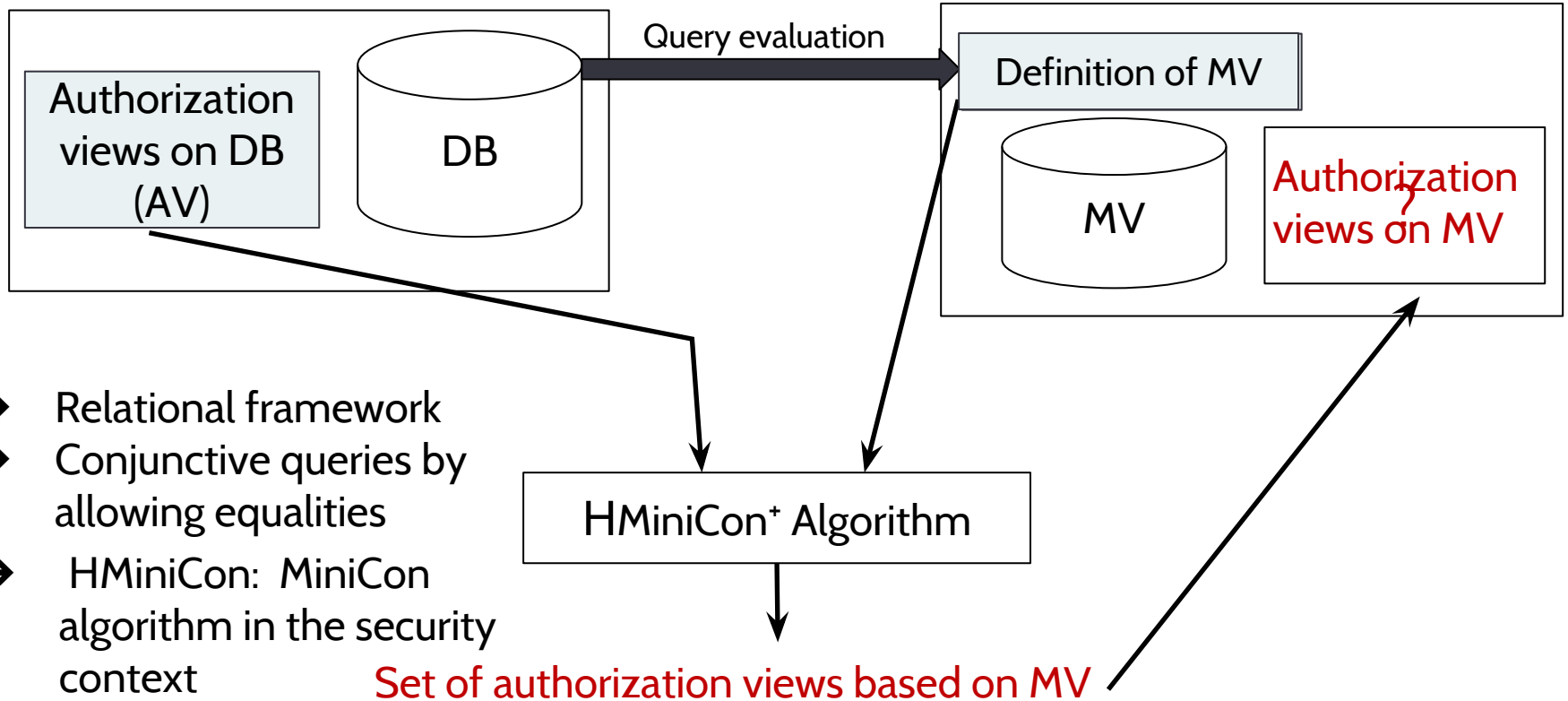
- How to ensure Security at the materialized view level?



Related Work

	Granularity	Derived access control policies
[Ros&Sci IFIP'01]	Coarse	Defined on base relations
[Cuz&al. IDEAS'10]	Fine	Defined on base relations
Our approach	Fine	Defined on MVs

Our approach



- ➔ Relational framework
- ➔ Conjunctive queries by allowing equalities
- ➔ HMiniCon: MiniCon algorithm in the security context

Desired properties: Secure and Maximum

Desired Properties

- **Security**: The generated views should not give access to information that are not allowed by the basic authorization views.

- **Maximality**: Generated views should return as much information as possible, while satisfying the secure property.

Access control policies

- Fine grained Access Control model based on "Authorization Views" [Riz&al. SIGMOD'04].
  - Authorization views are logical tables that specify exactly the accessible data, either drawn from a single table or from multiple tables.
  - An authorization view can be a traditional relational view or a parameterized view
    - Allowing fine grained authorization at the cell-level.
    - Parameterized views provide an efficient and powerful way of expressing fine grained authorization policies.

Access control policies - Example

Relations:

patients (IdP, IdD, Snum, Pname, Pfname, Disease).

Create authorization view patients_info as

SELECT Pname, Pfname

FROM patients

WHERE Snum = 1;

Datalog:

patients_info (Pname, Pfname) ←

patients (IdP, IdD, Snum, Pname, Pfname, Disease),

Snum = 1;

Access control policies

- Authorization-transparent querying
  - A Query makes reference to base relations
  - System can
    - Accept the query, if it can rewrite it using only authorization views
    - Reject the query

- Directly Querying only the authorization views
- Our proposal is independent of the way the MV(s) are accessed.
  - We assume in our approach that the user can query only the authorization views.

Information non-disclosure

- Determine which set of tuples can be accessed without disclosure information.

Authorization view:

av(x') ← patients(x',y').

Materialized view definition:

mv(x) ← patients(x,y), emergency(x,y).

Authorization view on the materialized view:

avmv(x) ← mv(x).

- There is no authorized access to mv to ensure the information non-disclosure.

HMiniCon Algorithm

- Adaptation of a query rewriting algorithm to the security context.
- MiniCon algorithm: proposed as an efficient method for answering queries using views **[Pot&Lev VLDB'00]** .
  - It takes as input a query q and a set of views V and calculates all possible rewritings of q using views in V, such that:

$$rw \subseteq q$$

  - Condition: Each rewriting must have the same head variables as the query.

Why adapt MiniCon?

Query:

$q(x,y) \leftarrow patients(x,y).$

Views:

$v(x') \leftarrow patients(x',y').$

- For the traditional MiniCon Algorithm, this view is not relevant.
  - The condition regarding the head variables is not satisfied.
- In the security context, this view is relevant
  - Conjunctive rewriting: $rw(x) \leftarrow v(x).$

➔ **First adaptation:** Relaxing the condition on the head variables.

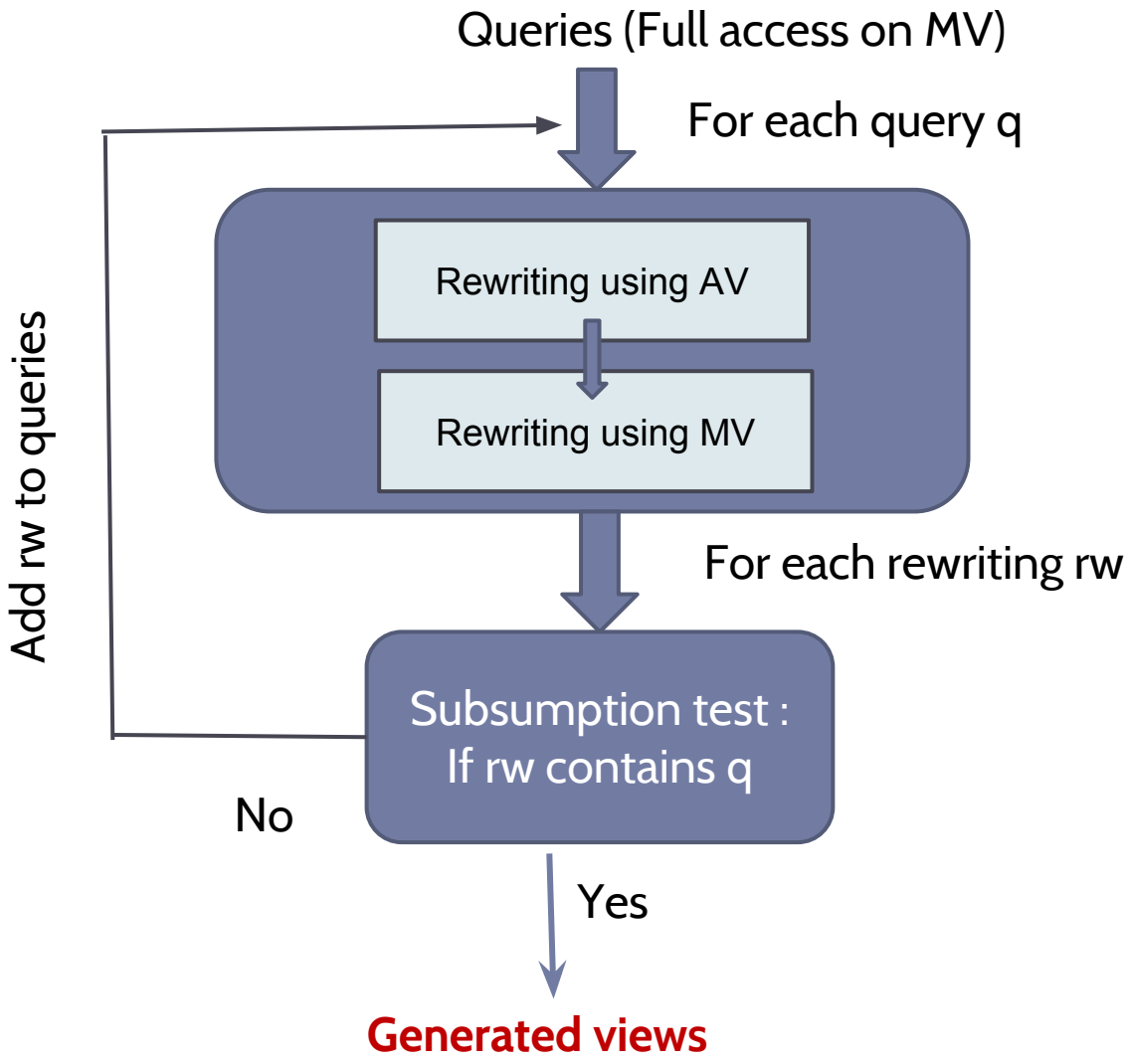
➔ **Second adaptation:** Adding variables that are newly introduced in the rewriting as head variables.

Double rewriting

- It Exploits a double query rewriting based on the HMiniCon query rewriting algorithm.

- It takes as input a set Q of queries to be rewritten and two sets of views AV and MV
  - ➔ Q: Complete queries on MV
  - ➔ AV: Authorization views
  - ➔ MV: Materialized views definitions

HMiniCon+


Queries (Full access on MV)
For each query q
Rewriting using AV
Rewriting using MV
For each rewriting rw
Subsumption test :
If rw contains q
No
Add rw to queries
Yes
Generated views

Properties of ⊢*MiniCon*$^{+}$ *Algorithm*

Security property

Property: Given the three sets *AV*, *MV and AVMV* (the set of generated views by HMiniCon$^+$ algorithm), For each query on AVMV, there exists:

$$q^{AVMV} \equiv q^{AV} \text{ et}$$

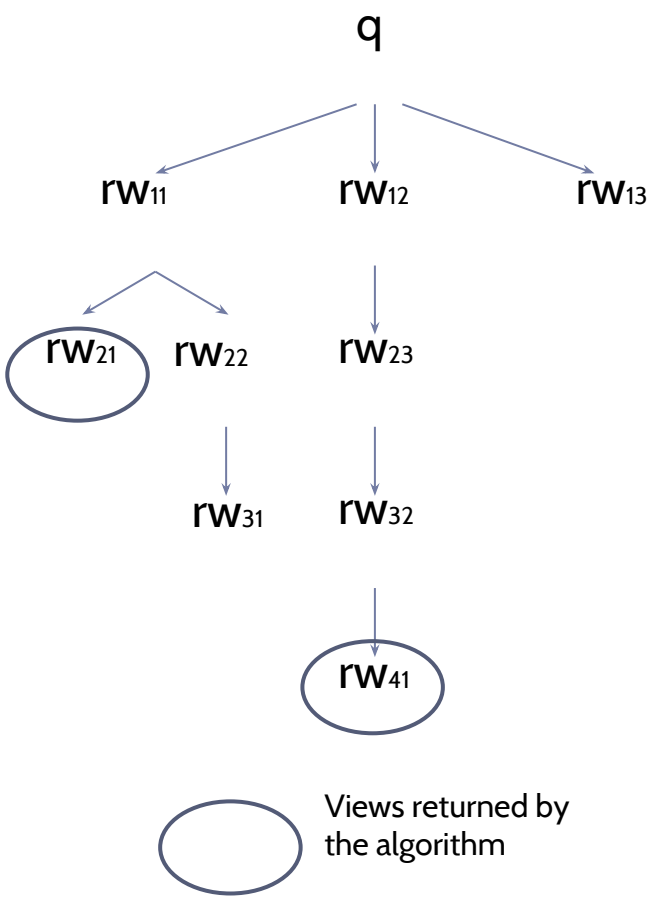
$$q^{AVMV} \equiv q^{MV}$$

Termination

- Rewriting tree
- Atom tree
- History of a node

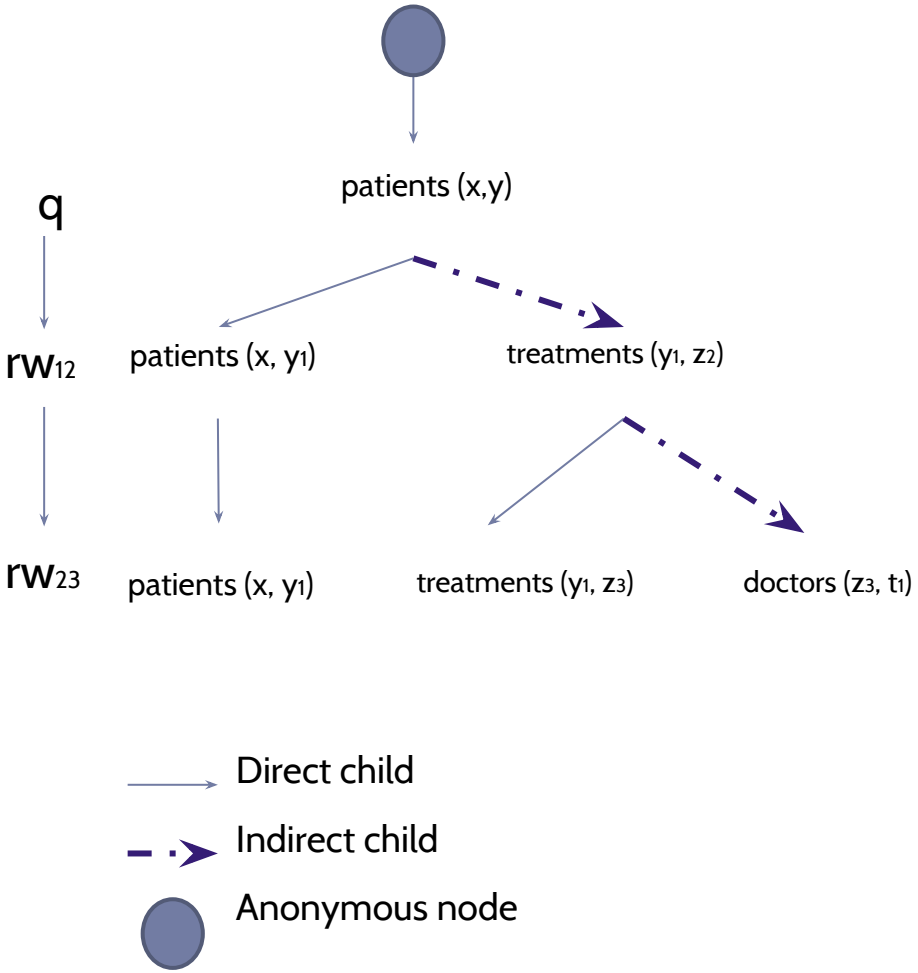
Rewriting Tree

- Let q be a query to rewrite, AV and MV are two sets of views. The rewriting tree associated with q is defined as follows:
  - The root is the query q.
  - The nodes of depth k+1 are rewritings generated by the HMiniCon algorithm by rewriting nodes of depth k using the set AV or MV.
  - A node n^{k+1} is a child of a node n^k if n^{k+1} is a rewriting of n^k.



Atom tree

- Given a branch X = B^0,B^1,... of a rewriting tree RT , the atom tree AT of RT is defined as:
  - The root is an anonymous node r.
  - Nodes at depth k+1 are occurrences of atoms of B^k, noted g^k.
  - $g^{k+1}$is a child of g^k of type:
  - ➔ **Direct:** If it is mapped to g^k at the construction of the rewriting
  - ➔ **Indirect:** If g^{k+1} belongs to the expansion of view v used to rewrite g^k and g^{k+1} has no Direct parent.



Potential infinite loop in the rewriting process Example 1

MV:

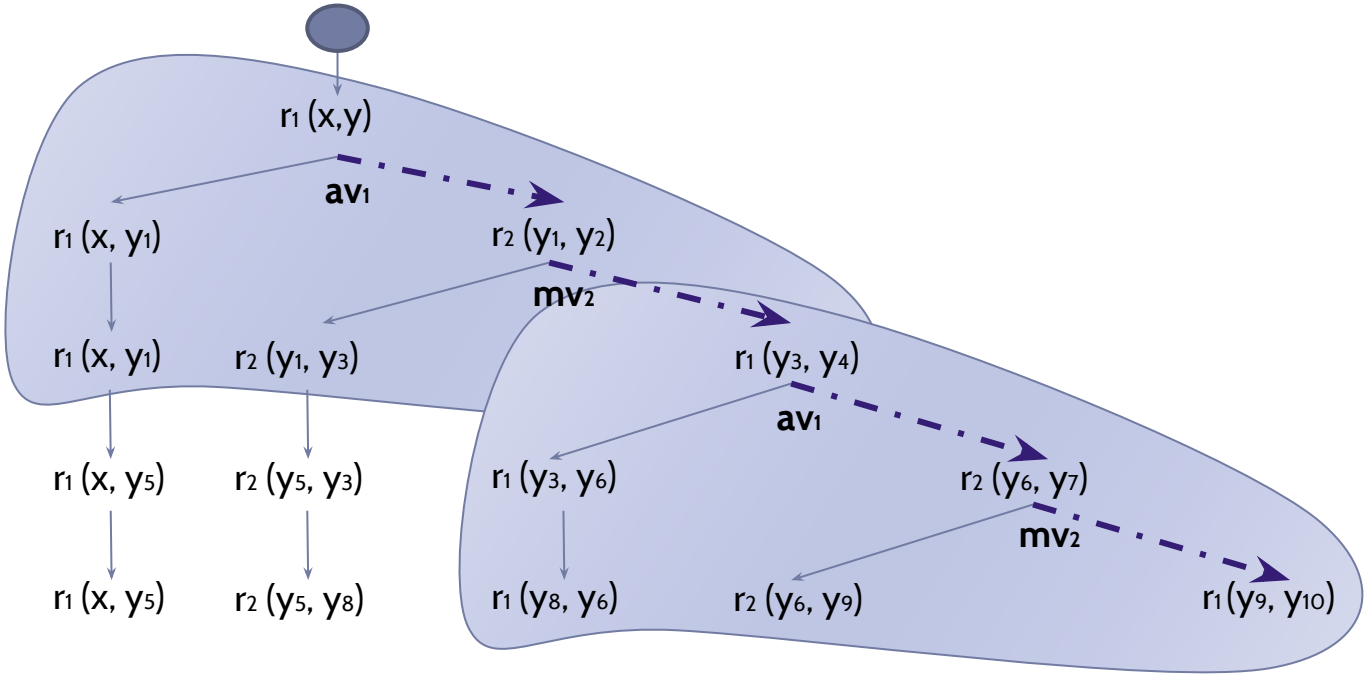
mv_1 (x,y) ← r_1 (x,y).

mv_2 (x,y) ← r2(x,y),r1(y,z).

AV:

av_1 (x,y) ← r1(x,y),r2(y,z).

av_2 (x,y) ← r2(x,y).



Potential infinite loop in the rewriting process Example 2

MV:

mv1(x,y) ← r1(x,y),r3(y,z).

mv2(x,y) ← r2(x,y).

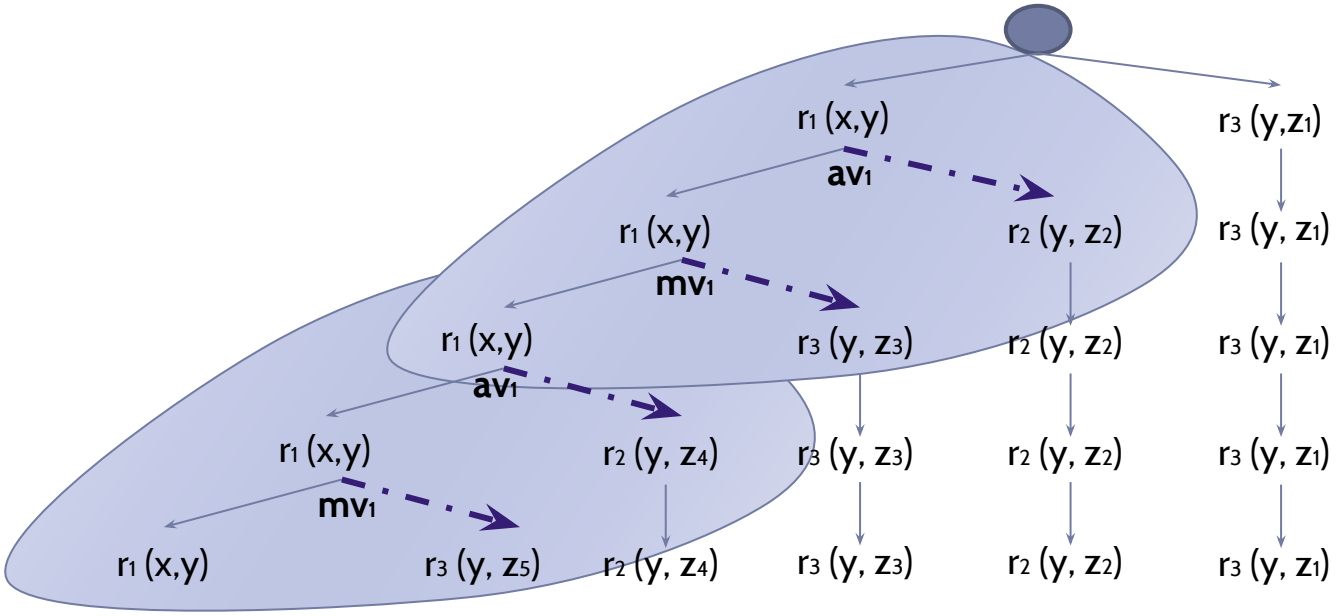
mv3(x,y) ← r3(x,y).

AV:

av1(x,y) ← r1(x,y),r2(y,z).

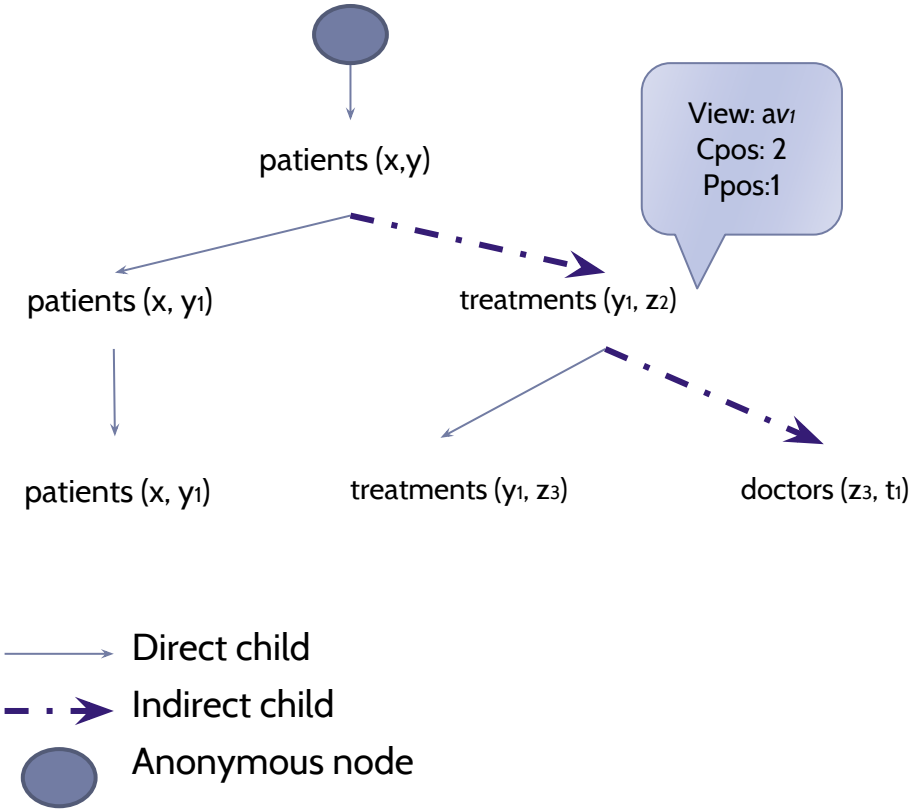
av2(x,y) ← r2(x,y).

av3(x,y) ← r3(x,y).



Node information

- For each node, we have:
  - view(g^{k+1}) = v;
  - cpos(g^{k+1}) the position of the atom matching g^{k+1}in v;
  - ppos(g^{k+1}) the position of the atom matching g^{k} in v;
  - type(g^{k+1}) = Direct or Indirect



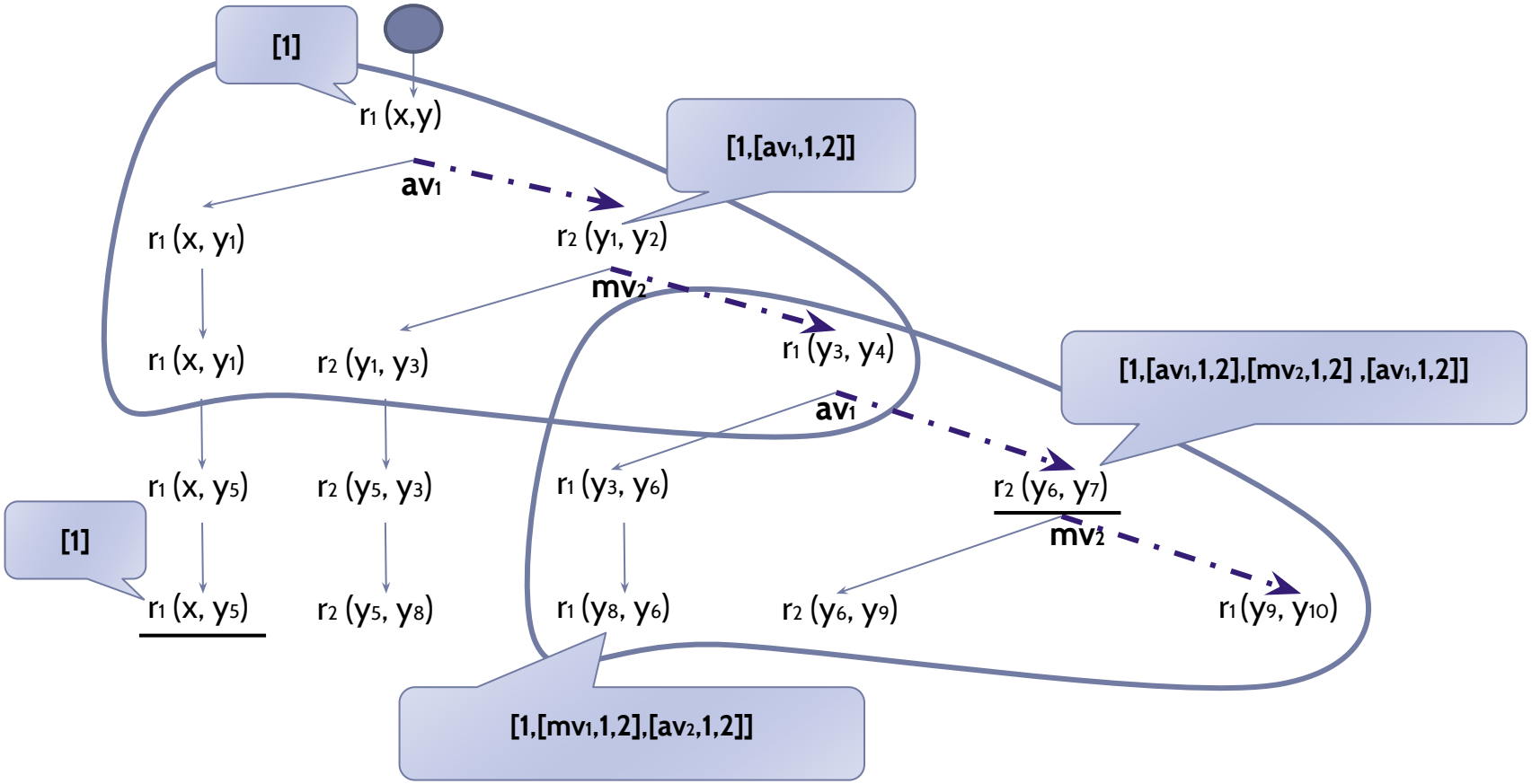
History of nodes

- For each node g in AT except for the root, History(g) is a list defined as follows:
  - if g is a child of the root, then History(g) = [pos] where pos is the position of g in the query;
  - if type(g) = Indirect then:

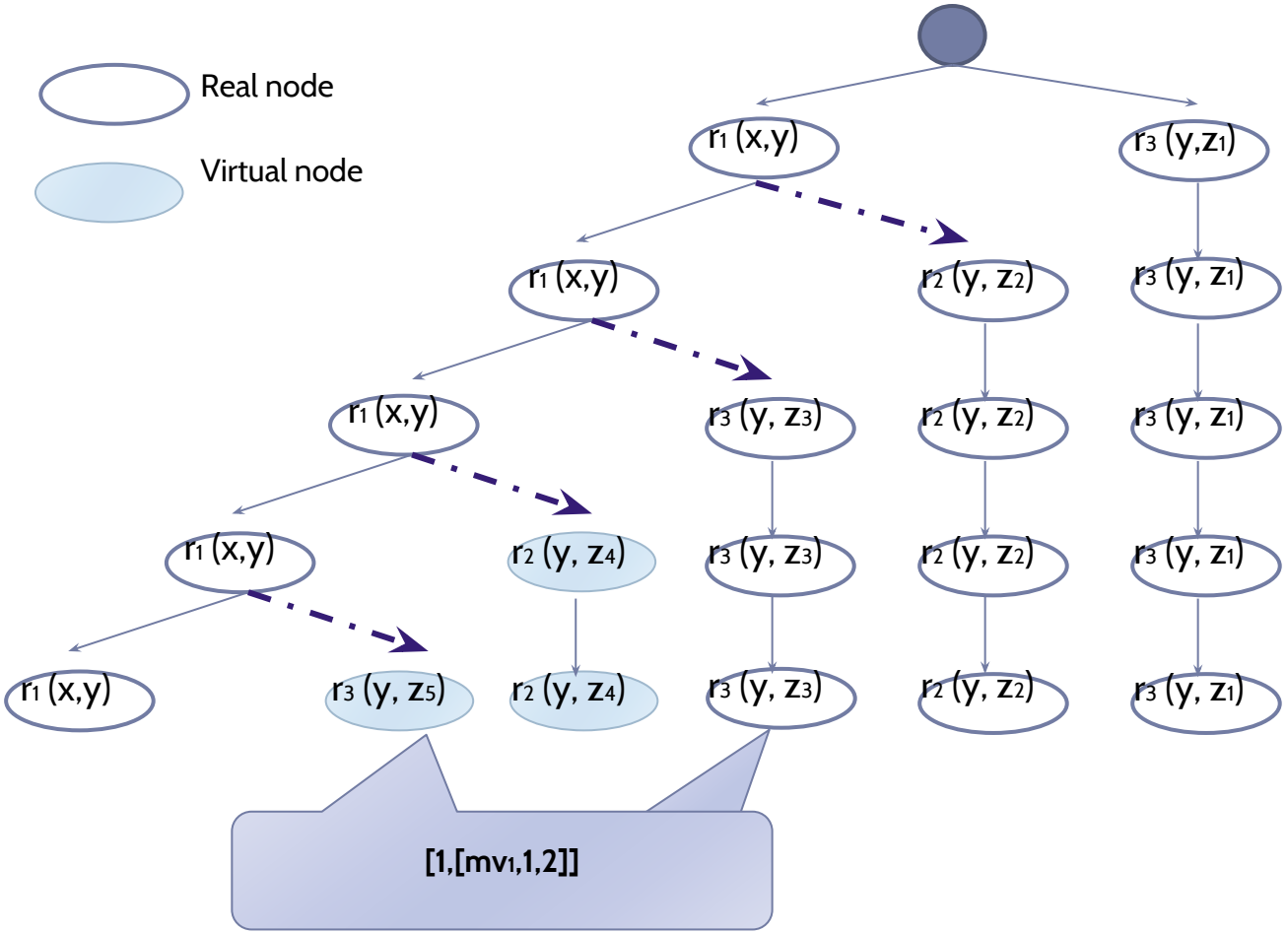
History(g) = History(parent(g)) + [(view(g),cpos(g),ppos(g))]

  - otherwise, History(g) = History(parent(g))

History of nodes - Example



Real VS Virtual nodes



Termination under constraints

Theorem 1 Let us consider a query q and two sets of views AV and MV. If for every branch X of the effective rewriting tree RT (q) generated by HMiniCon^{+}(q, AV, MV) and for every node g of the atom tree AT of X, History (g) does not contain any duplicate triple, then RT is finite.

Maximality property

Property: Given the three sets *AV*, *MV and AVMV* (the set of generated views by HMiniCon$^+$ algorithm) and for each query on AV and each query on MV, such that:

$$q^{AV} \equiv q^{MV}$$

Then, there exists a query on AVMV, such that:

$$q^{AVMV} \equiv q^{AV} \equiv q^{MV}$$

Conclusion

- An automated method to generate access control policies for materialized views.
- An adaptation of a query rewriting algorithm.
- Conjunctive queries with equalities
- A secure and maximal approach

- Study the maximality property in case of infinite rewrting trees
- Queries with aggregate functions..

Bibliography

- [Ros&Sci CAISE'00] A. Rosenthal and E. Sciore. *View security as the basis for data warehouse security*.
- [Cuz&al. IDEAS'10] A. Cuzzocrea, M.-S. Hacid, and N. Grillo. *Effectively and efficiently selecting access control rules on materialized views over relational databases.*
- [Pot&Lev VLDB'00] R. Pottinger and A. Y. Levy. *A scalable algorithm for answering queries using views.*
- [Riz&al. SIGMOD'04] S. Rizvi, A. O. Mendelzon, S. Sudarshan, and P. Roy. *Extending query rewriting techniques for fine-grained access control.*

THANK YOU FOR YOUR ATTENTION