Integrating Pattern Mining in Relational Databases

Toon Calders
Technische Universiteit Eindhoven

Bart Goethals and Adriana Prado
Universiteit Antwerpen

How?

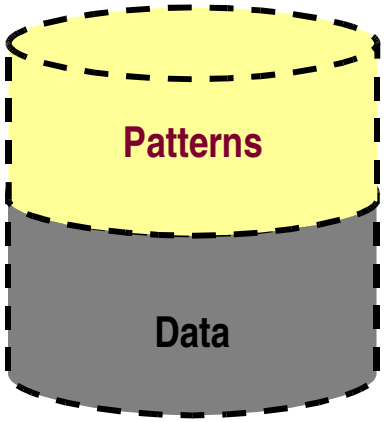
- Extend SQL
  - DMQL (Han et. al, 1996)
  - MINE RULE (Meo et. al, 1998)
  - MSQL (Imielinski and Virmani, 1999)
  - …
- Extend DBMS

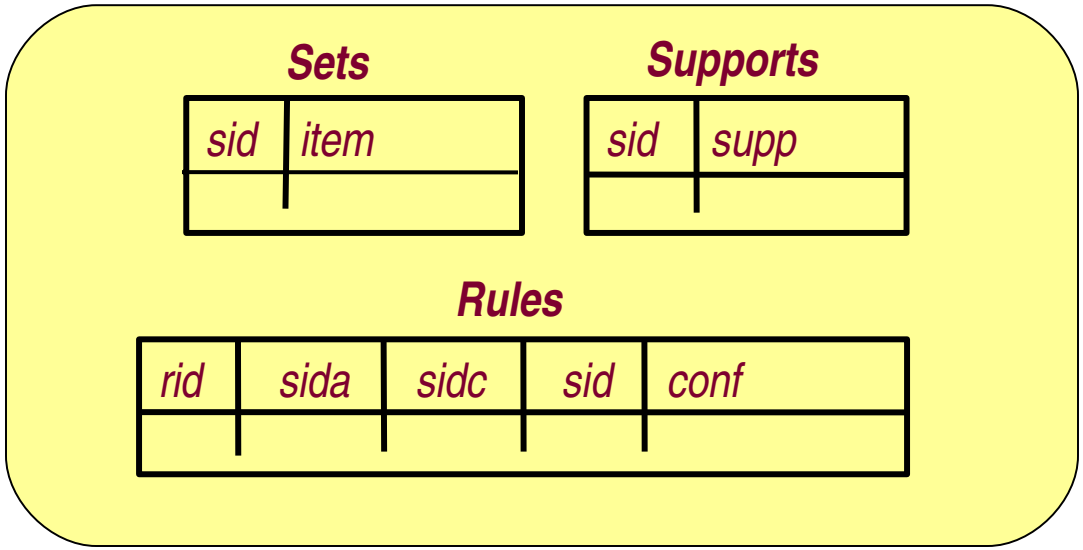
Idea

- The user can query the collection of all possible patterns as if they were stored in relational databases (pattern tables), using e.g. SQL.

- The user does not need to deal with the data mining algorithms themselves.

Pattern Tables for Association Rule Mining





Challenge

- How can this be implemented effectively for association rule mining?
  - There are an exponential number of itemsets and association rules!
  - Storing all of them is impossible in practice!

Solution

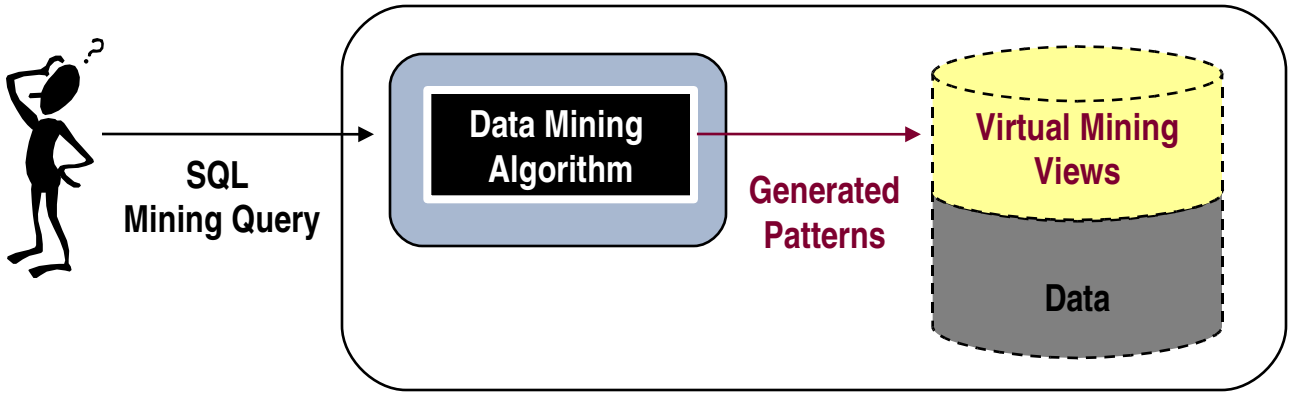
- Keep these pattern tables virtual:

"*virtual mining views".*

  - For the user: All possible patterns are stored.
  - Reality: No such complete tables exist...

How?

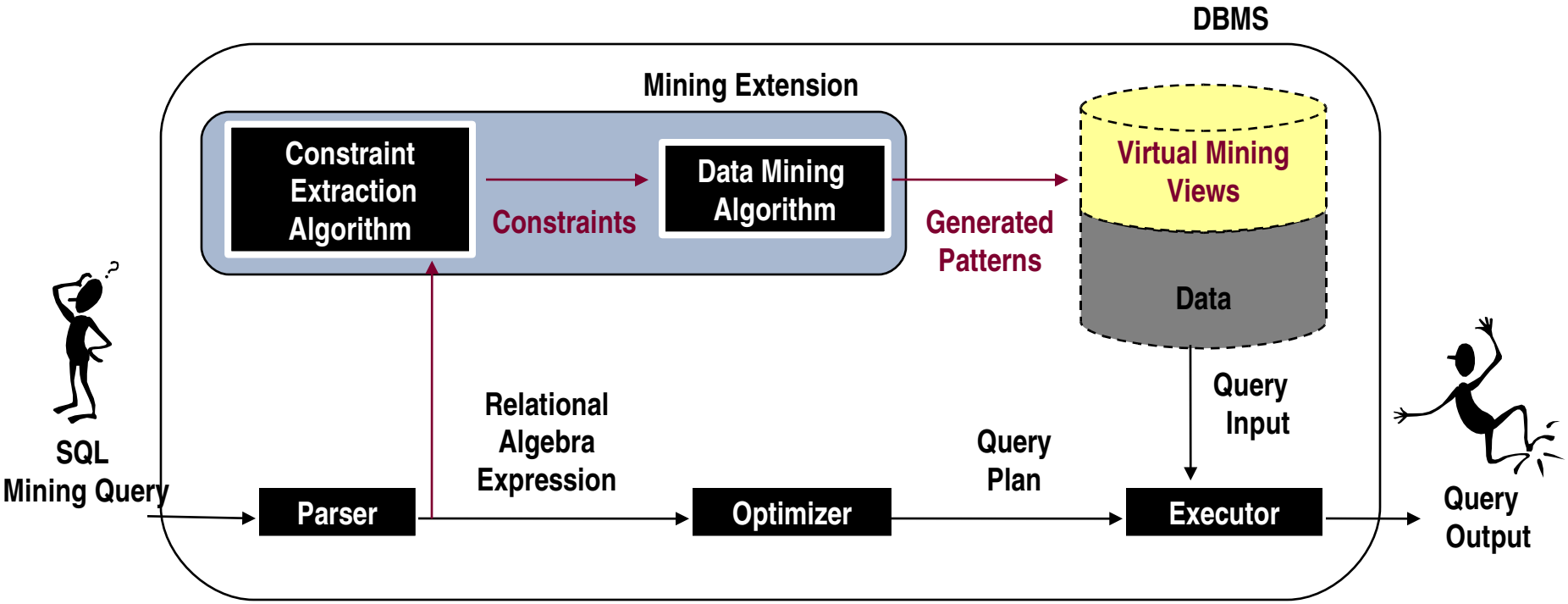
- For each query, the system finds and materializes all tuples necessary to answer the query correctly.



Goal

- Typically, the user poses certain constraints in his query.
- Detecting these constraints allows us to limit the amount of tuples that needs to be materialized.
- Constraint Extraction Algorithm.

Idea


DBMS
Mining Extension
Constraint Extraction Algorithm
Constraints
Data Mining Algorithm
Generated Patterns
Virtual Mining Views
Data
SQL Mining Query
Relational Algebra Expression
Query Plan
Query Input
Query Output
Parser
Optimizer
Executor

Example

Select all rules that
contain ‘apple’,
having support >= 40
and confidence >= 80%.

```
select R.rid, R.conf
from Sets S1,
     Supports S2,
     Rules R
where S2.sid    = R.sid
  and S1.sid    = S2.sid
  and S1.item   = 'apple'
  and S2.supp   >= 40
  and R.conf    >= 80%
```

Example

Example Query:

```
select R.rid, R.conf
from Sets S1, Supports S2,
     Rules R
where S2.sid    = R.sid
  and S1.sid    = S2.sid
  and S1.item   = 'apple'
  and S2.supp   >= 40
  and R.conf    >= 80%
```

Constraints exploited by the data mining algorithm:

Item Constraint = 'apple'
Minimum Support = 40
Minimum Confidence = 80%

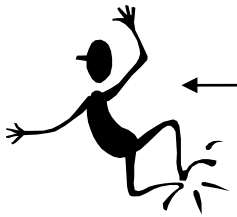
Materialization of the Virtual Mining Views:

Sets table: itemsets with 'apple' and support of at least 40.

Supports table: itemsets with 'apple' and support of at least 40.

Rules table: rules with 'apple', support of at least 40, and confidence of at least 80%.

Execution of the Query

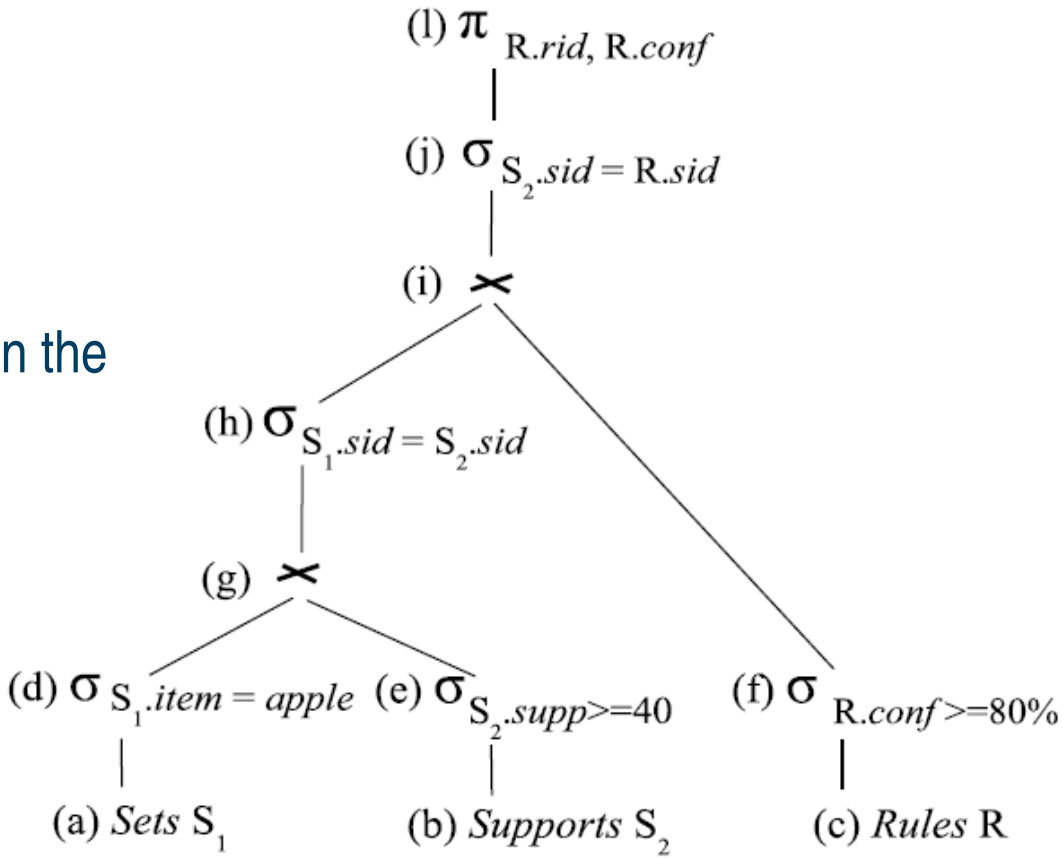


Constraint Extraction Algorithm

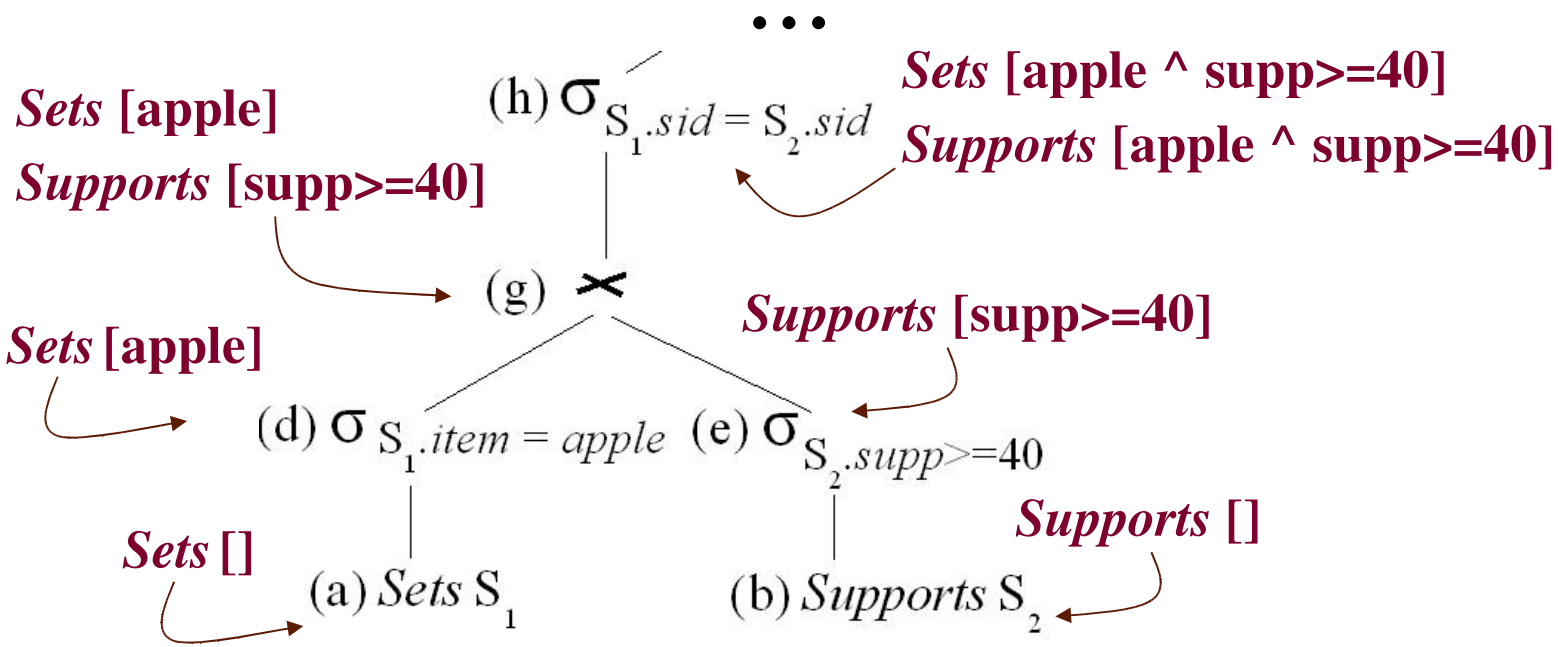
- Currently, we are able to extract Boolean combinations of the following constraints:
  - Minimum and maximal support.
  - Minimum and maximal confidence.
  - A certain item must be in the antecedent and/or in the consequent.

How?

- Botton-up, traverse a relational algebra tree.
  - Find for each subquery, which tuples should be in the virtual mining views.



Example


...
(h) $\sigma_{S_1.sid = S_2.sid}$
Sets [apple]
Supports [supp>=40]
Sets [apple ^ supp>=40]
Supports [apple ^ supp>=40]
(g) ×
Sets [apple]
Supports [supp>=40]
(d) $\sigma_{S_1.item = apple}$
(e) $\sigma_{S_2.supp>=40}$
Sets []
Supports []
(a) Sets S_1
(b) Supports S_2

Work in Progress

- Implementation into PostgreSQL
  - Already working for simple queries!
    (not yet for aggregation, correlated subqueries…).

- Adoption of the same philosophy for *Decision Trees* (with Élisa Fromont and Hendrik Blockeel).
  - Demo developed in PL-SQL (Oracle).
  - "Integrating Decision Tree Learning into Inductive Databases", Élisa Fromont and Hendrik Blockeel, KDID 2006.

Future Work

- Other types of constraints (e.g. aggregation).
- Other types of patterns.
- Incremental Data Mining.
- …

Reference

- T. Calders, B. Goethals, and A. Prado. “Integrating Pattern Mining in Relational Databases”. In Proc. PKDD Int. Conf. Principles of Data Mining and Knowledge Discovery, Berlin, Germany,2006.

Thank you!
Dank u wel!