A Novel Incremental Approach to Constraint-Based Mining

Rosa Meo

University of Torino, Italy

UNIVERSITÀ DEGLI STUDI DI TORINO
ALMA UNIVERSITAS TAURINENSIS

Outline

- Motivations
  - constraint-based mining
  - iterative and interactive mining, inductive databases
- Query evaluation exploiting materializations
- Constraints properties
  - Item dependent
  - Context dependent
- Incremental algorithms
  - performance results
- Conclusions

UNIVERSITÀ DEGLI STUDI DI TORINO
ALMA UNIVERSITAS TAURINENSIS

Motivations

- Knowledge Discovery from Databases (KDD) is usually an interactive and iterative process
- This sequence consists in constraint-based queries which are very often a refinement of previous ones
- The system is a multi-user environment
- Inductive queries are *iceberg queries* and are executed on-line

UNIVERSITÀ DEGLI STUDI DI TORINO
ALMA UNIVERSITAS TAURINENSIS

Motivations

- **Problem**

  Each new query cannot be executed from scratch: unfeasible workload for the extraction engine

- **Proposed solution**

  The problem can be solved materializing previous queries and adopting an "incremental" engine, in the sense that it can derive the result of a query Q reusing the result of previous, "correlated" queries

Incremental Mining

- The term *incremental* usually refers to updating the result set of a query when the database has been updated
- Now, we want to update the result of a query when a new ("correlated") query is submitted and the database is not changed
  - Purpose: enhancing query evaluation exploiting the materializations

UNIVERSITÀ DEGLI STUDI DI TORINO
ALMA UNIVERSITAS TAURINENSIS

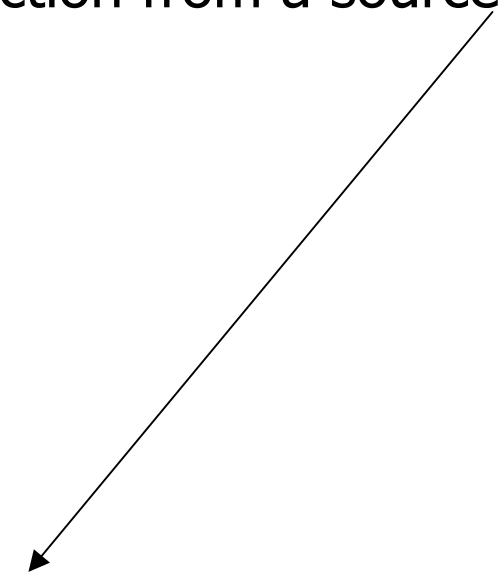
Optimizing Mining Queries

- The novel languages for data mining need for optimizers
- can exploit the available information in the database:
  - database schema (keys and functional dependencies)
  - data values distributions and constraint selectivity
  - the indices (*mining indices*)
  - results of previous, correlated queries

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table

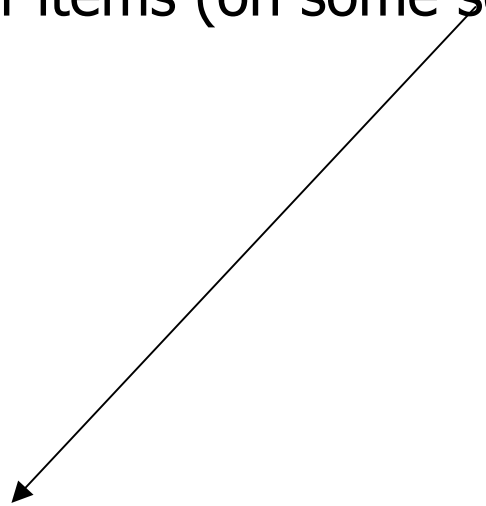


- $R = Q(T, G, I, \Gamma(M), \Xi)$

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)

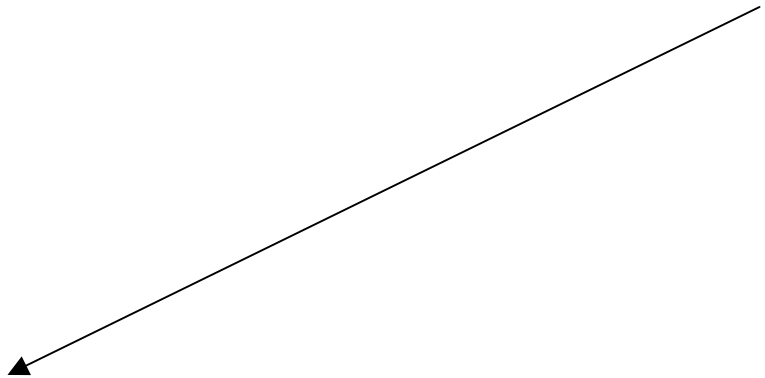


- $R=Q(T,G,I,\Gamma(M),\Xi)$

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)
- satisfying some user defined constraints (mining constraints)

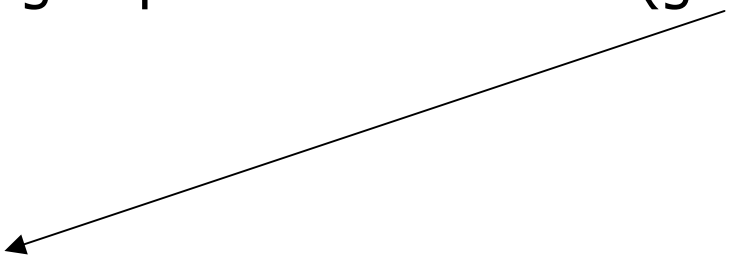


- $R=Q(T,G,I,\Gamma(M),\Xi)$

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)
- satisfying some user defined constraints (mining constraints)
- from the groups of the database (grouping constraints)



- $R=Q(T,G,I,\Gamma(M),\Xi)$

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)
- satisfying some user defined constraints (mining constraints)
- from the groups of the database (grouping constraints)
- The number of such groups must be sufficient (user defined statistical evaluation measure, such as support)

- $R=Q(T,G,I,\Gamma(M),\Xi)$

An Example

- R=Q(purchase,customer,product,price>100,support_count>=2)

purchase

transaction	customer	product	date	price	quantity
1	1001	ski_pants	12/7/98	140	1
1	1001	hiking_boots	12/7/98	180	1
3	1001	jacket	13/7/98	300	1
2	2256	col_shirt	13/7/98	25	2
2	2256	brown_boots	13/7/98	150	1
2	2256	jacket	13/7/98	300	1
4	2256	col_shirt	14/7/98	25	3
4	2256	jacket	14/7/98	300	2

Mining Query

R

itemset	support_count
{jackets}	2

Relationships Between Two Queries Q_1 and Q_2

- **Query equivalence**: $R_1 = R_2$
  no computation is needed
- **Query containment:** $R_2 \subseteq R_1$
  - We need to determine which element I in R_1 satisfies also (the more tight) constraints of Q_2
  - In the general case we need to make access to the database and verify that I satisfies constraints of Q_2
  - In same "lucky" cases we can evaluate constraints only once for each I, with no need to make any scan of the database

Context dependent constraints (CDC)

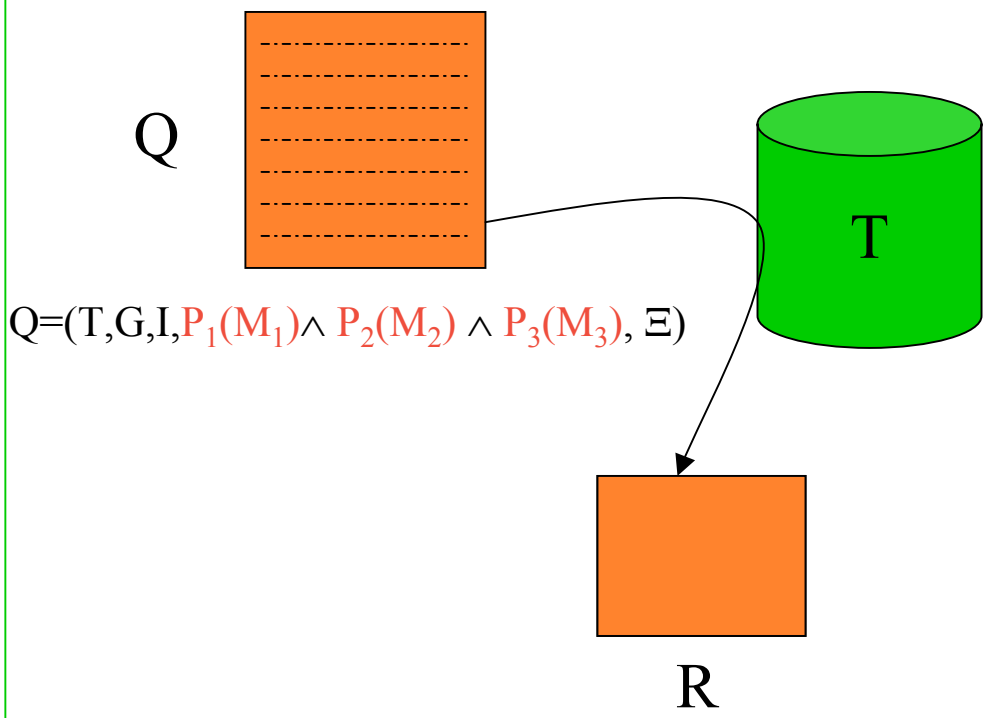
Item dependent constraints (IDC)

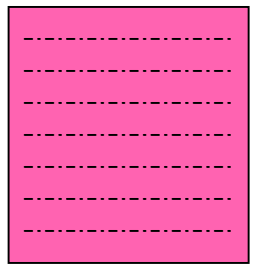
Searching for Equivalence

- In order to allow the optimizer to recognize equivalent queries, we allow *query rewriting*
- Query rewriting:
  determination of a relational expression on a set of other queries whose result is equivalent to the result of the rewritten query but is better in terms of execution costs

Query Rewriting

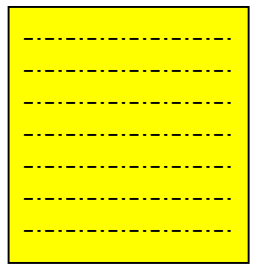
- Query rewriting of a query for itemset mining: join of other results





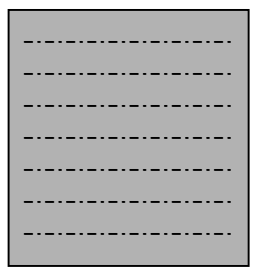
Q_1

$Q_1=(T,G,I,P_1(M_1), \Xi)$



Q_2

$Q_2=(T,G,I,P_2(M_2), \Xi)$

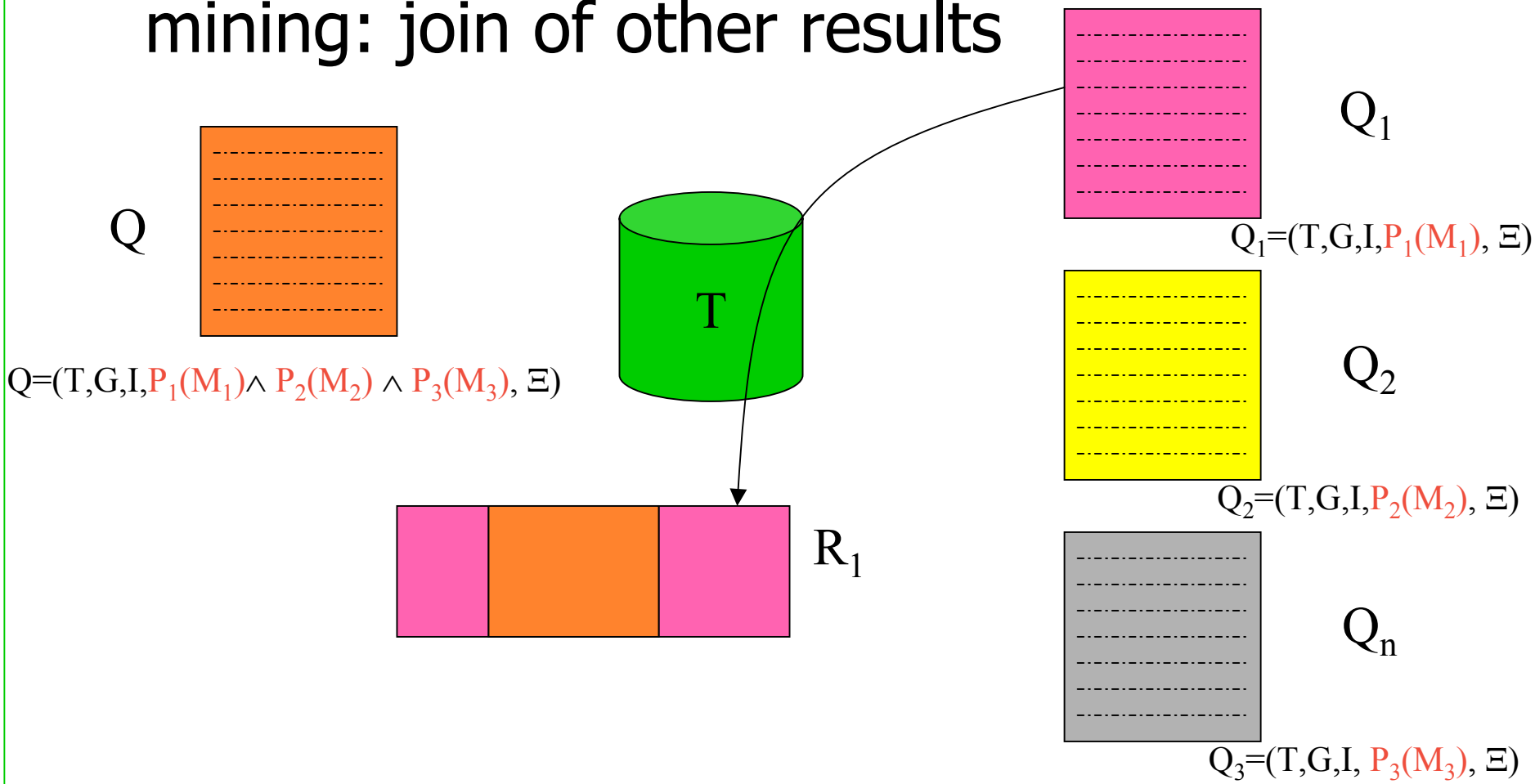


Q_n

$Q_3=(T,G,I, P_3(M_3), \Xi)$

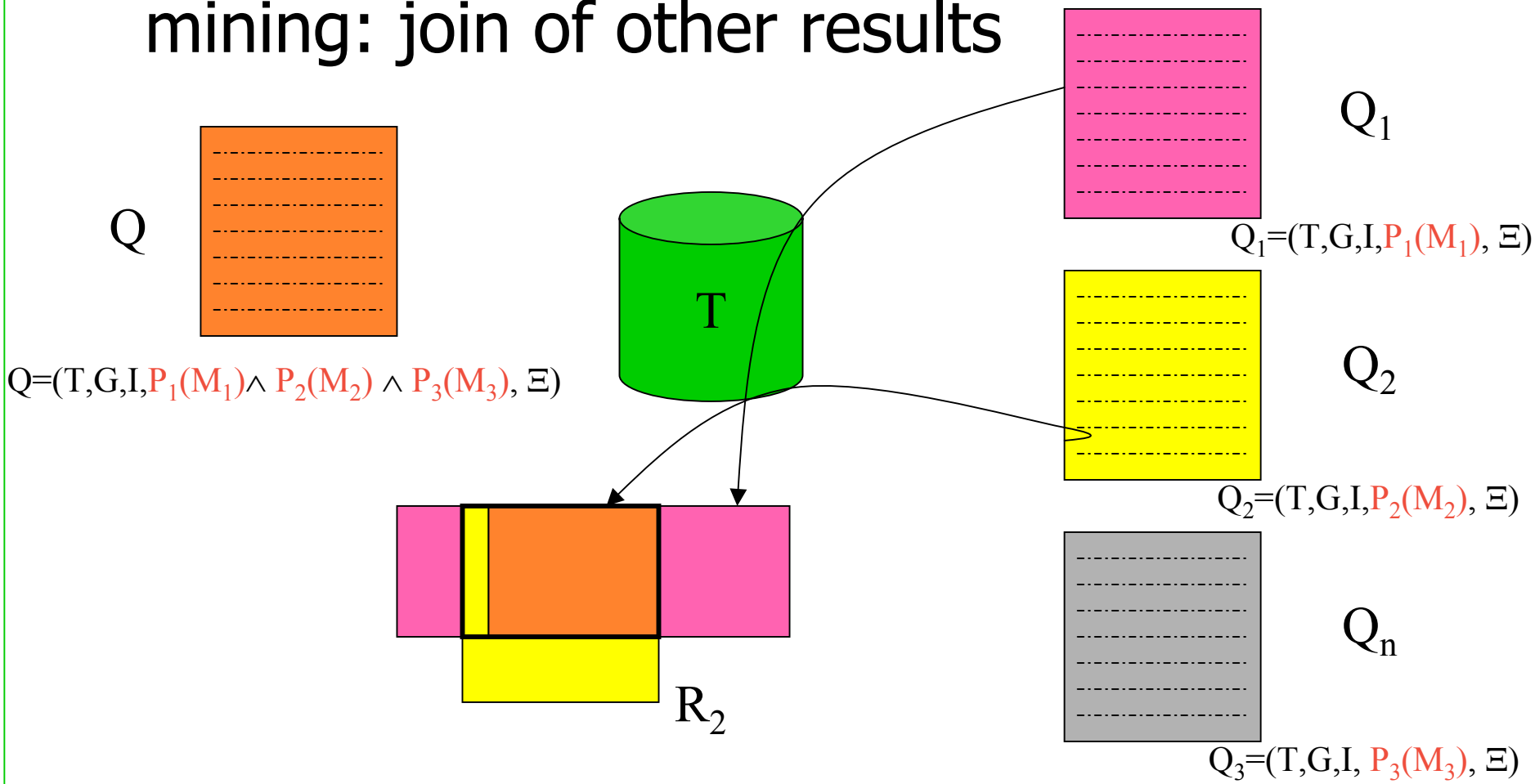
Query Rewriting

- Query rewriting of a query for itemset mining: join of other results



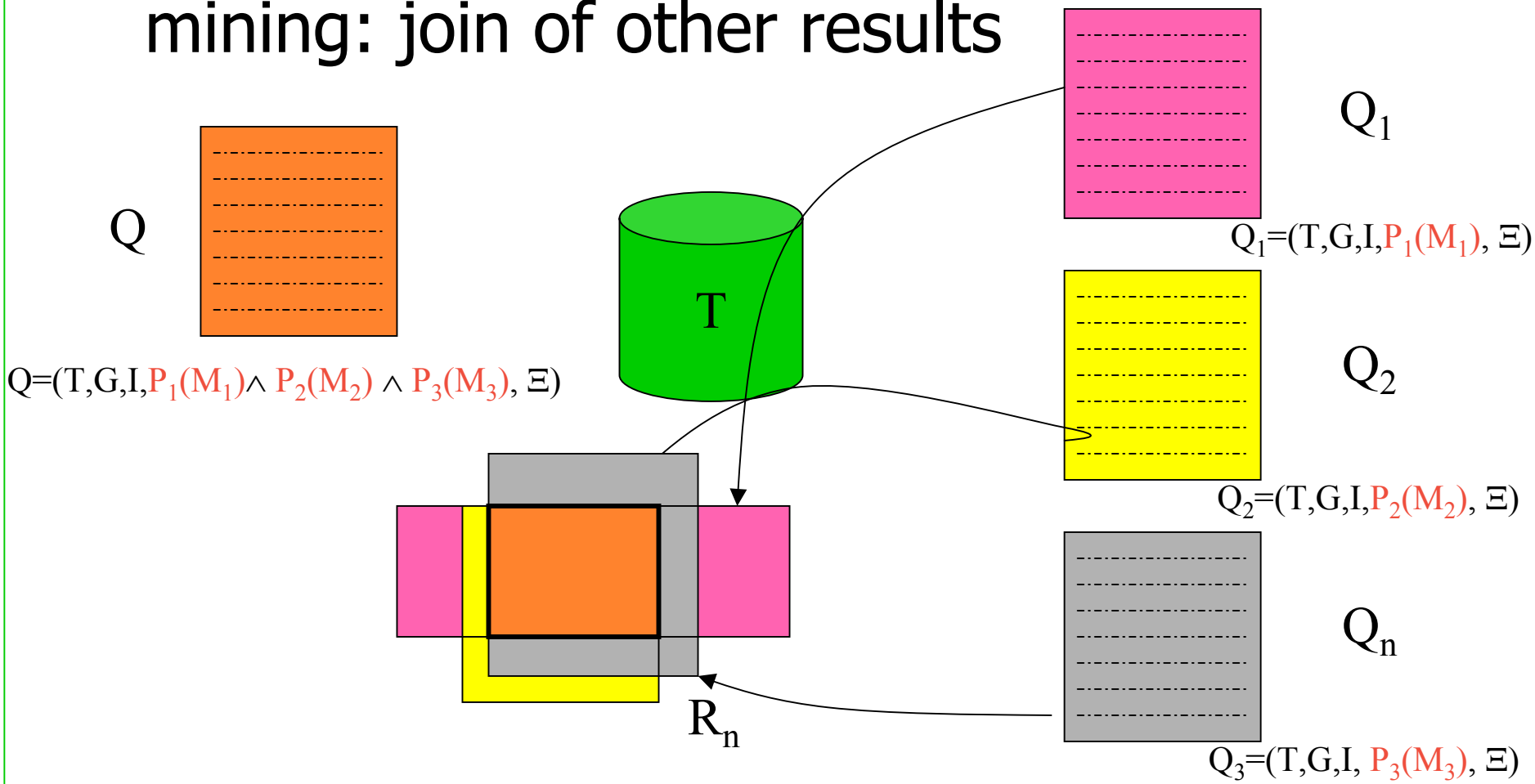
Query Rewriting

- Query rewriting of a query for itemset mining: join of other results



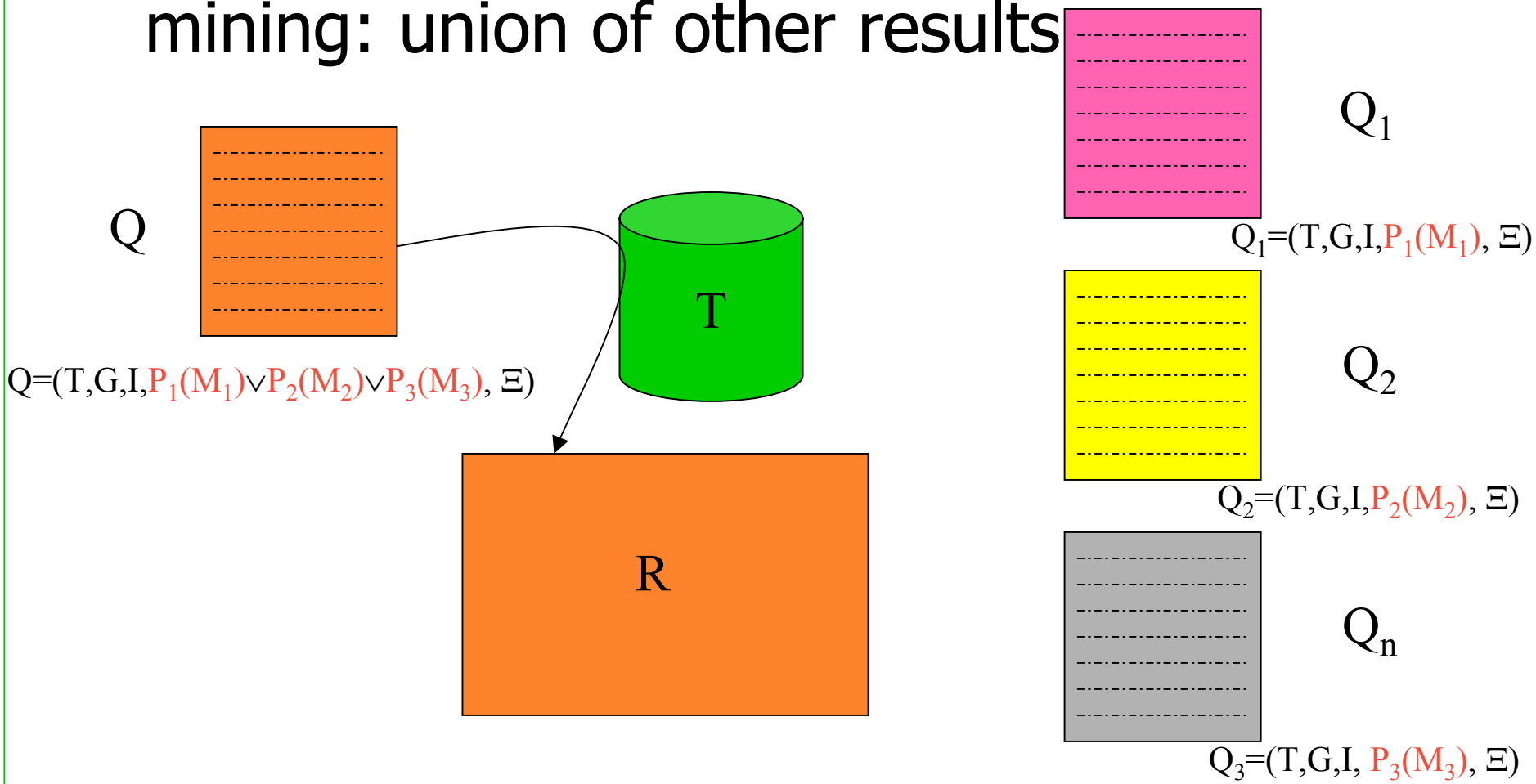
Query Rewriting

- Query rewriting of a query for itemset mining: join of other results



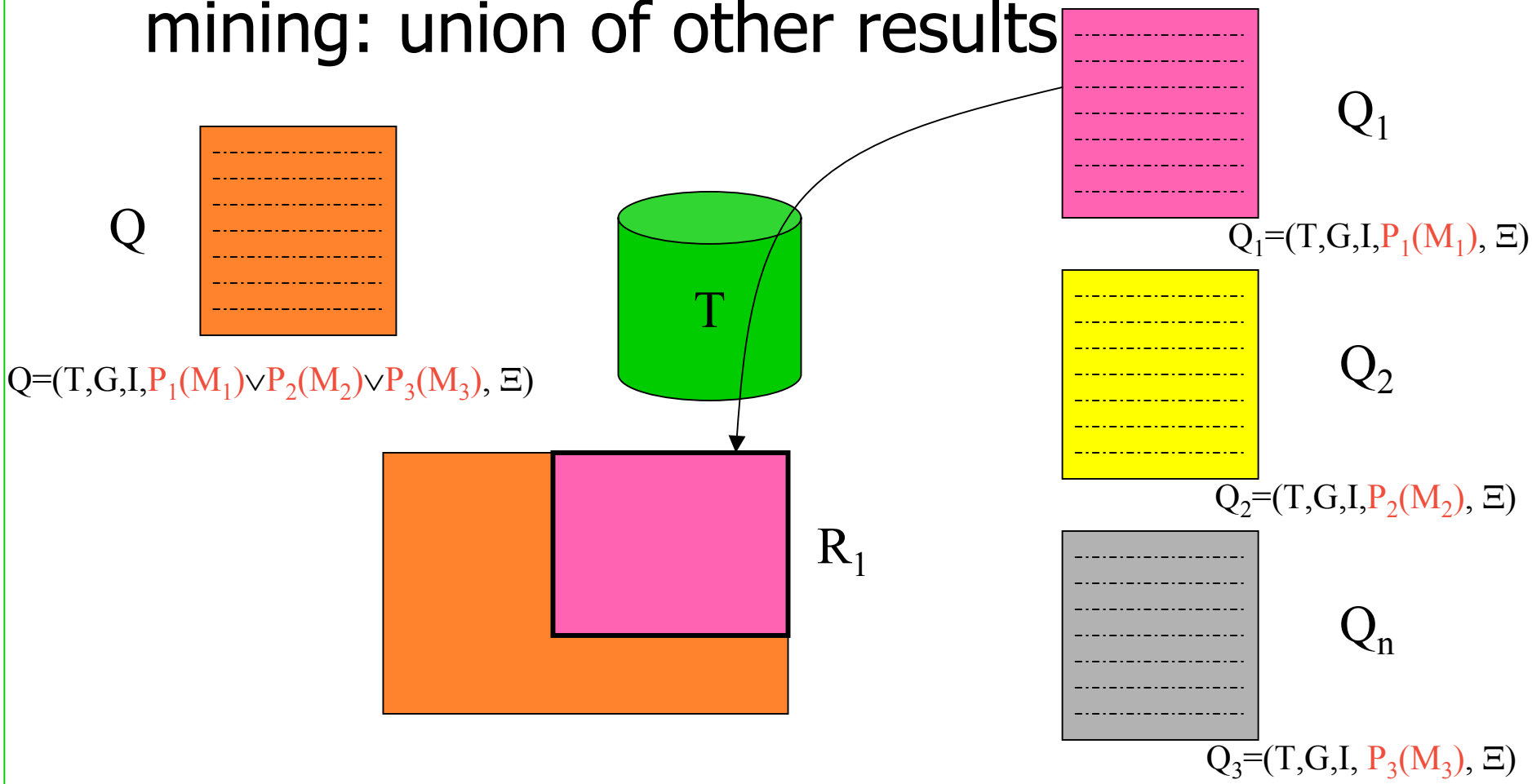
Query Rewriting (2)

- Query rewriting of a query for itemset mining: union of other results



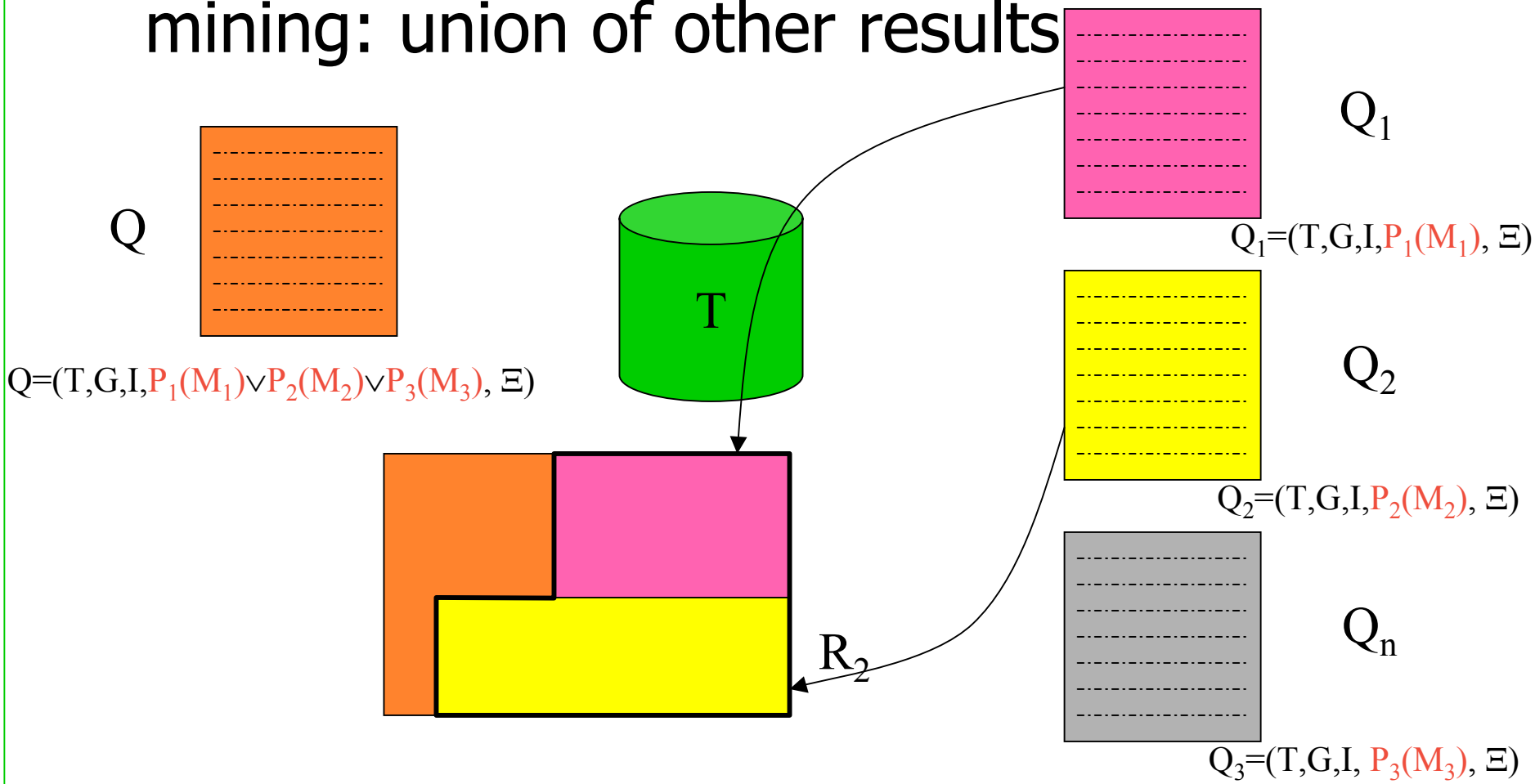
Query Rewriting (2)

- Query rewriting of a query for itemset mining: union of other results



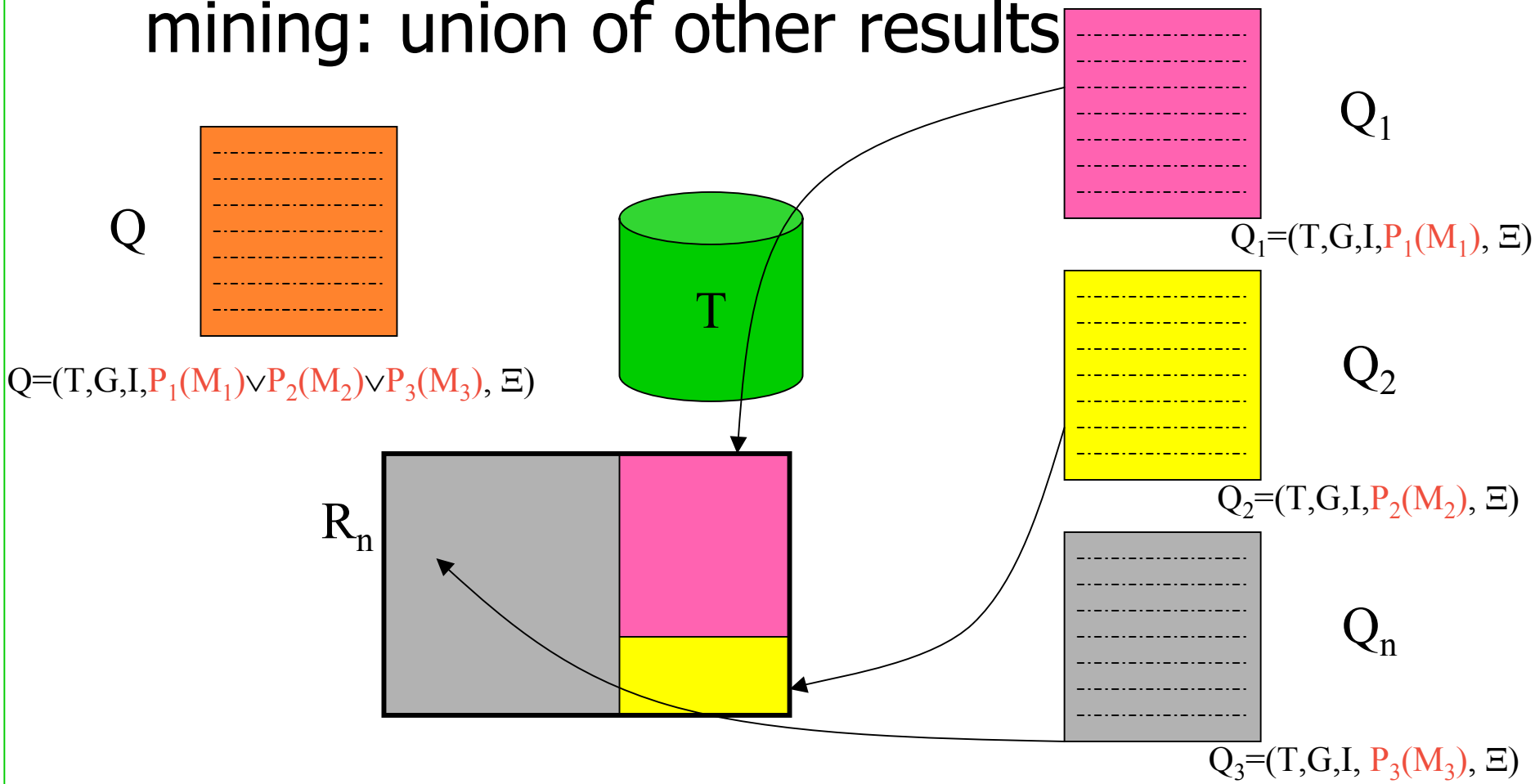
Query Rewriting (2)

- Query rewriting of a query for itemset mining: union of other results



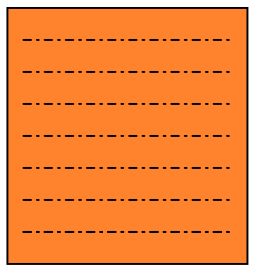
Query Rewriting (2)

- Query rewriting of a query for itemset mining: union of other results



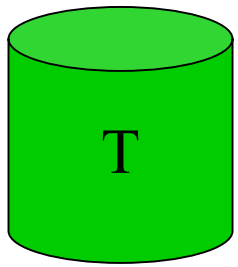
Query Rewriting (2)

- Query rewriting of a query for itemset mining: union of other results

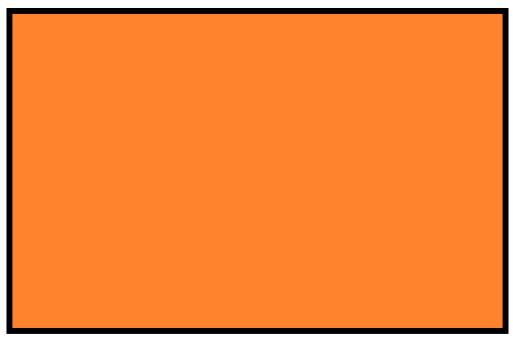


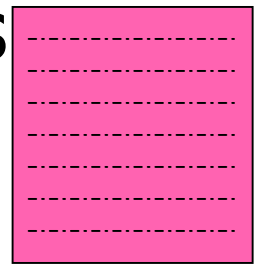
Q

$Q=(T,G,I,P_1(M_1)\vee P_2(M_2)\vee P_3(M_3), \Xi)$



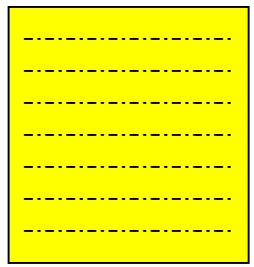
R





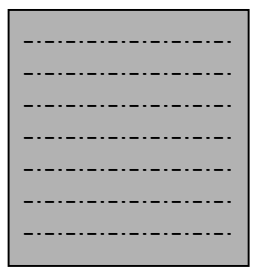
Q_1

$Q_1=(T,G,I,P_1(M_1), \Xi)$



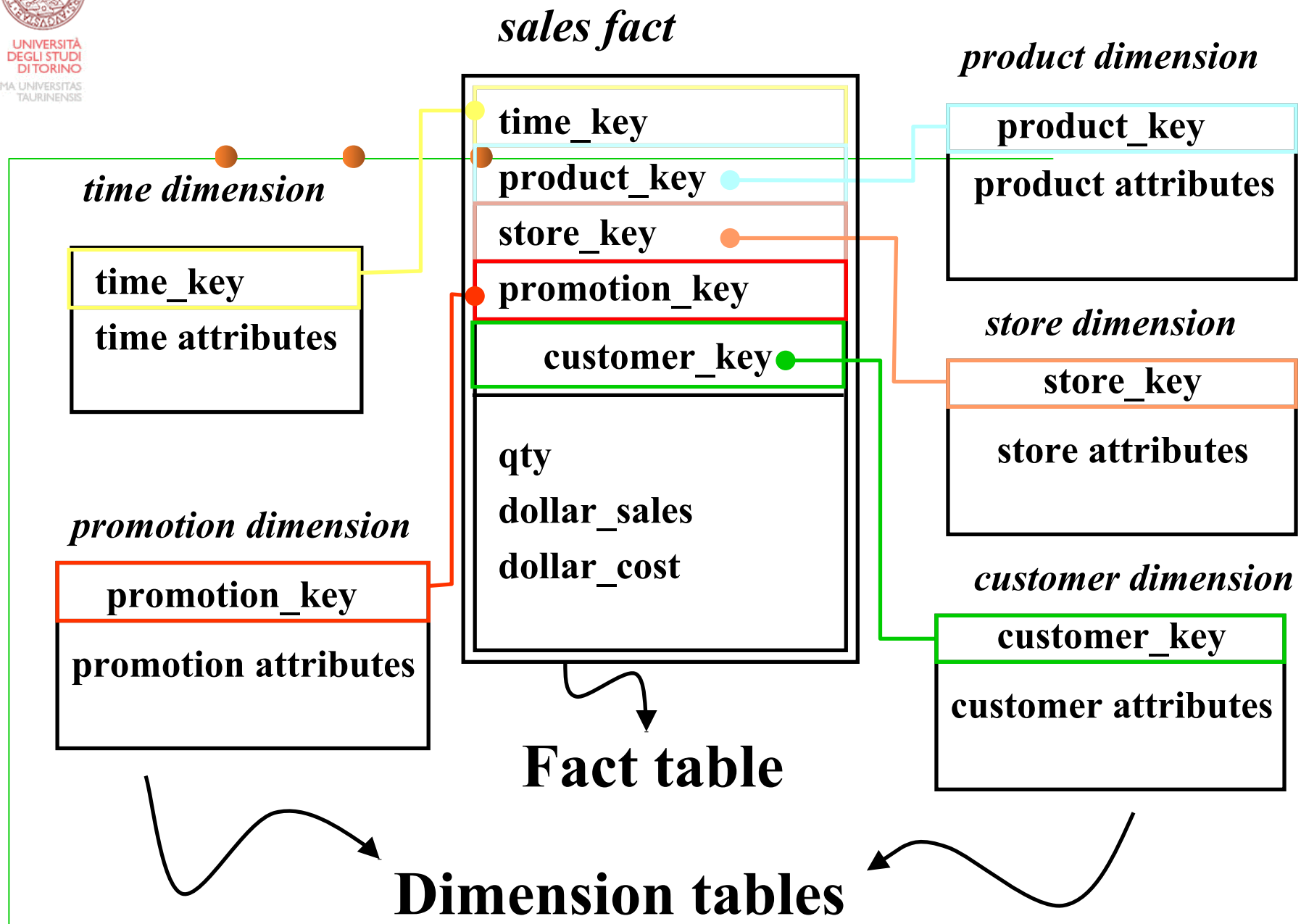
Q_2

$Q_2=(T,G,I,P_2(M_2), \Xi)$



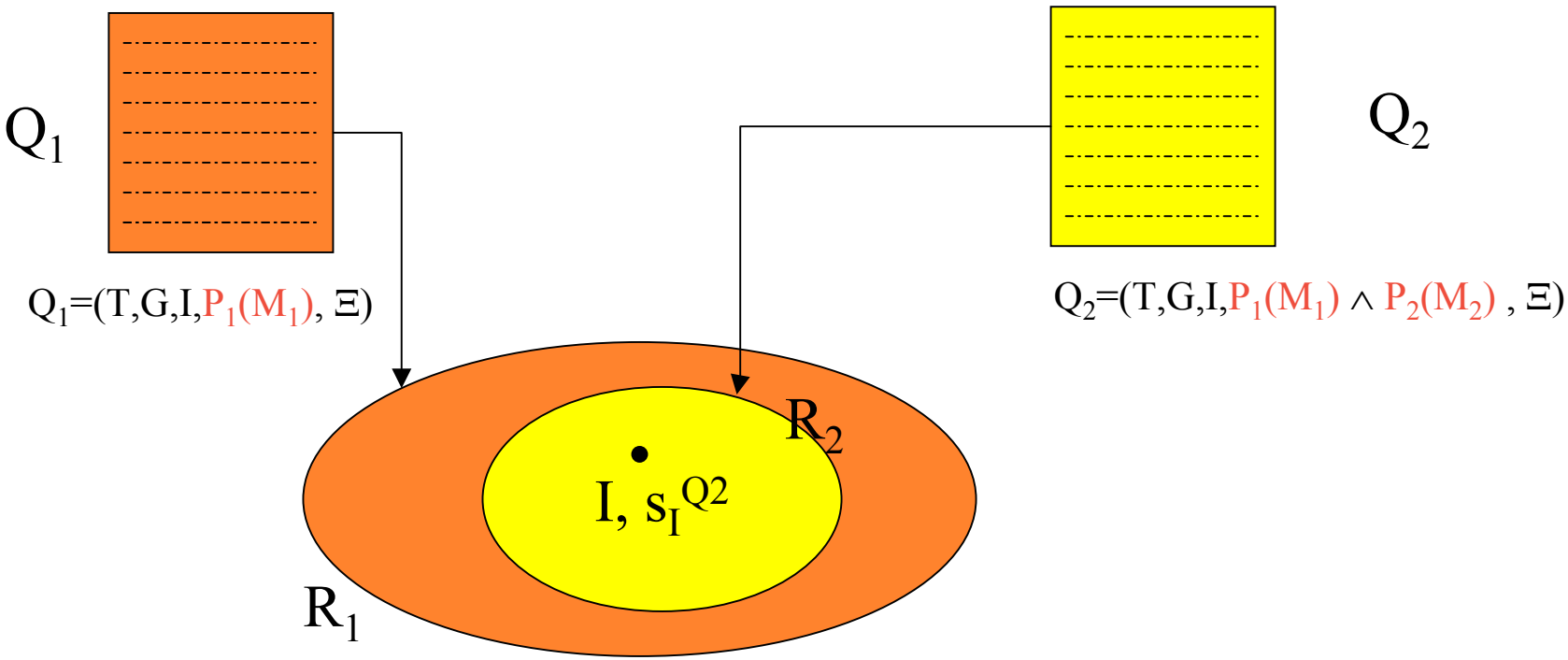
Q_n

$Q_3=(T,G,I, P_3(M_3), \Xi)$


sales fact
time_key
product_key
store_key
promotion_key
customer_key
qty
dollar_sales
dollar_cost
time dimension
time_key
time attributes
product dimension
product_key
product attributes
store dimension
store_key
store attributes
promotion dimension
promotion_key
promotion attributes
customer dimension
customer_key
customer attributes
Fact table
Dimension tables

Query Rewriting

- Can also be used for query containment
  - In the "lucky" cases, we have $R_2 = R_1 \bowtie \sigma_{C2}(D_I)$
  - In the general case, we have $R_2 = \rho\ \sigma_{C2}(R_1 \bowtie F \bowtie D_J)$



IDC vs CDC

- Item Dependent Constraints (***IDC***)
  - are functionally dependent on the item extracted
  - are satisfied for a given itemset either for *all* the groups in the database or for *none*
  - if an itemset is common to R1 and R2, it will have the same support: **inclusion**
- Context Dependent Constraints (***CDC***)
  - depend on the transactions in the database
  - might be satisfied for a given itemset only for *some* groups in the database
  - a common itemset to R1 and R2 might *not* have the same support: **dominance**

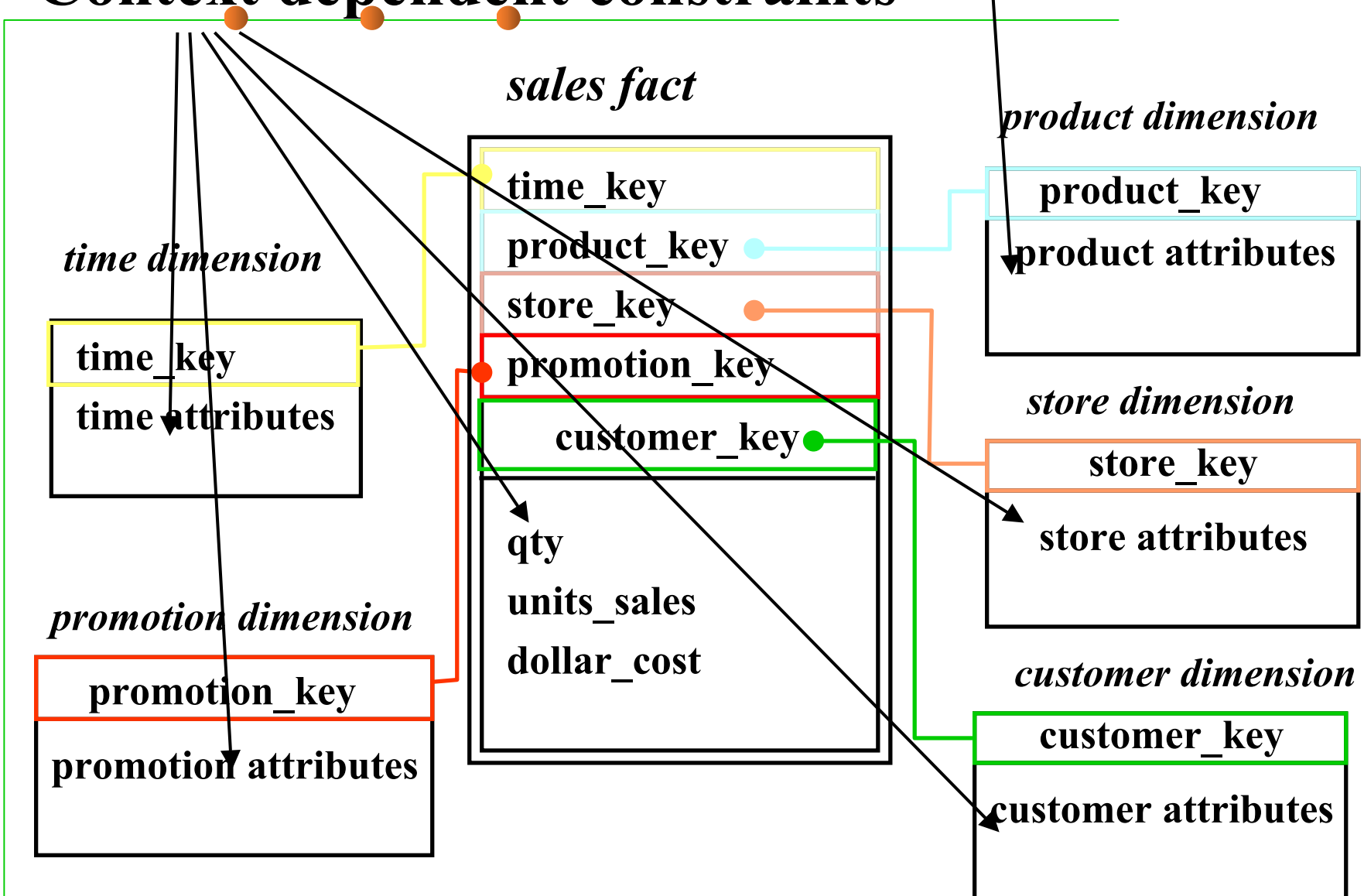
IDC: price > 150

CDC: qty > 1

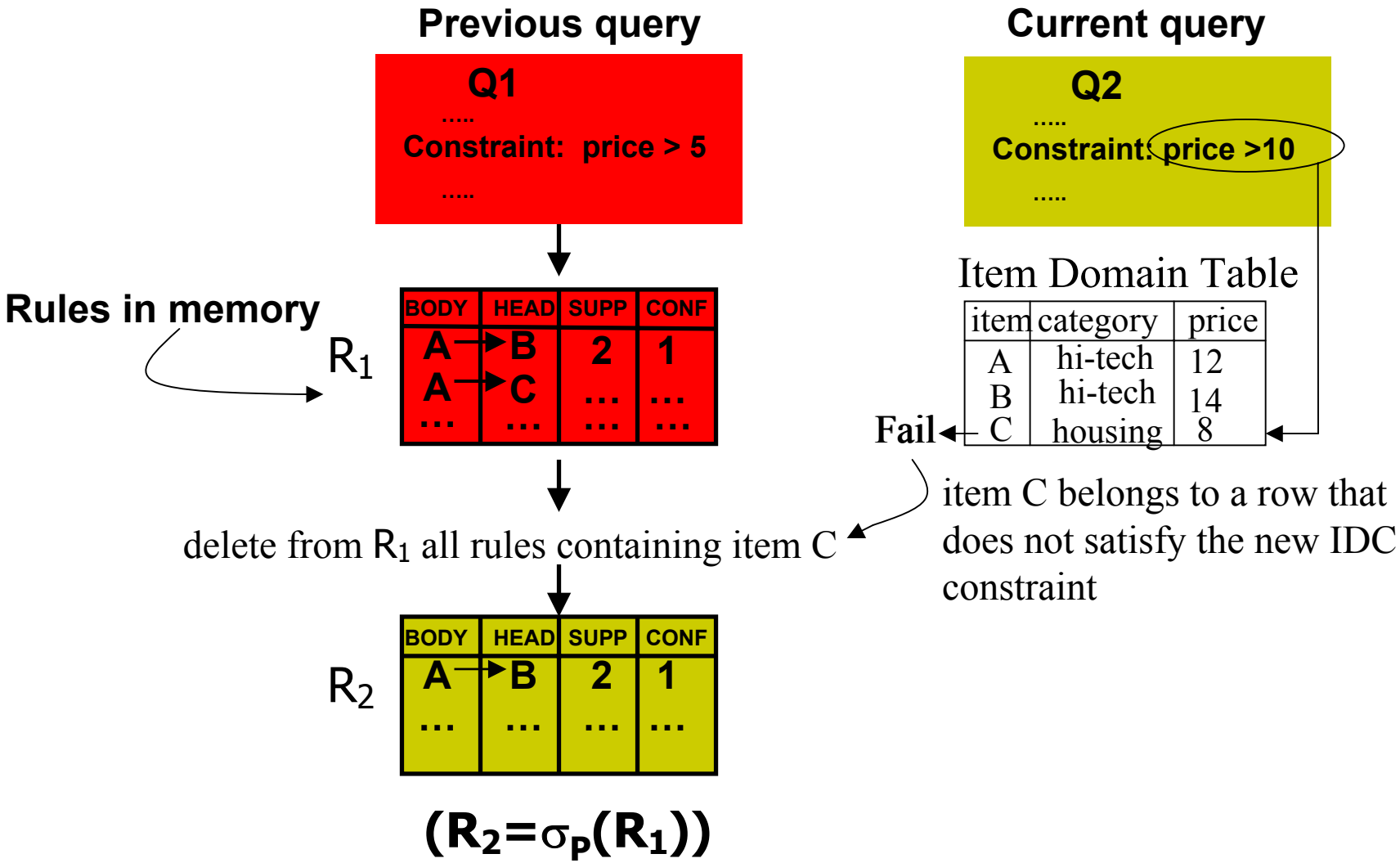
transaction	customer	product	date	price	quantity
1	1001	ski pants	12/7/98	140	1
1	1001	hiking boots	12/7/98	180	1
2	1001	jacket	17/7/98	300	2
2	2256	col shirt	12/7/98	25	2
2	2256	ski pants	13/7/98	140	2
3	2256	jacket	13/7/98	300	1
4	2256	col shirt	13/7/98	25	3
4	2256	jacket	20/8/98	300	2

Item dependent constraints

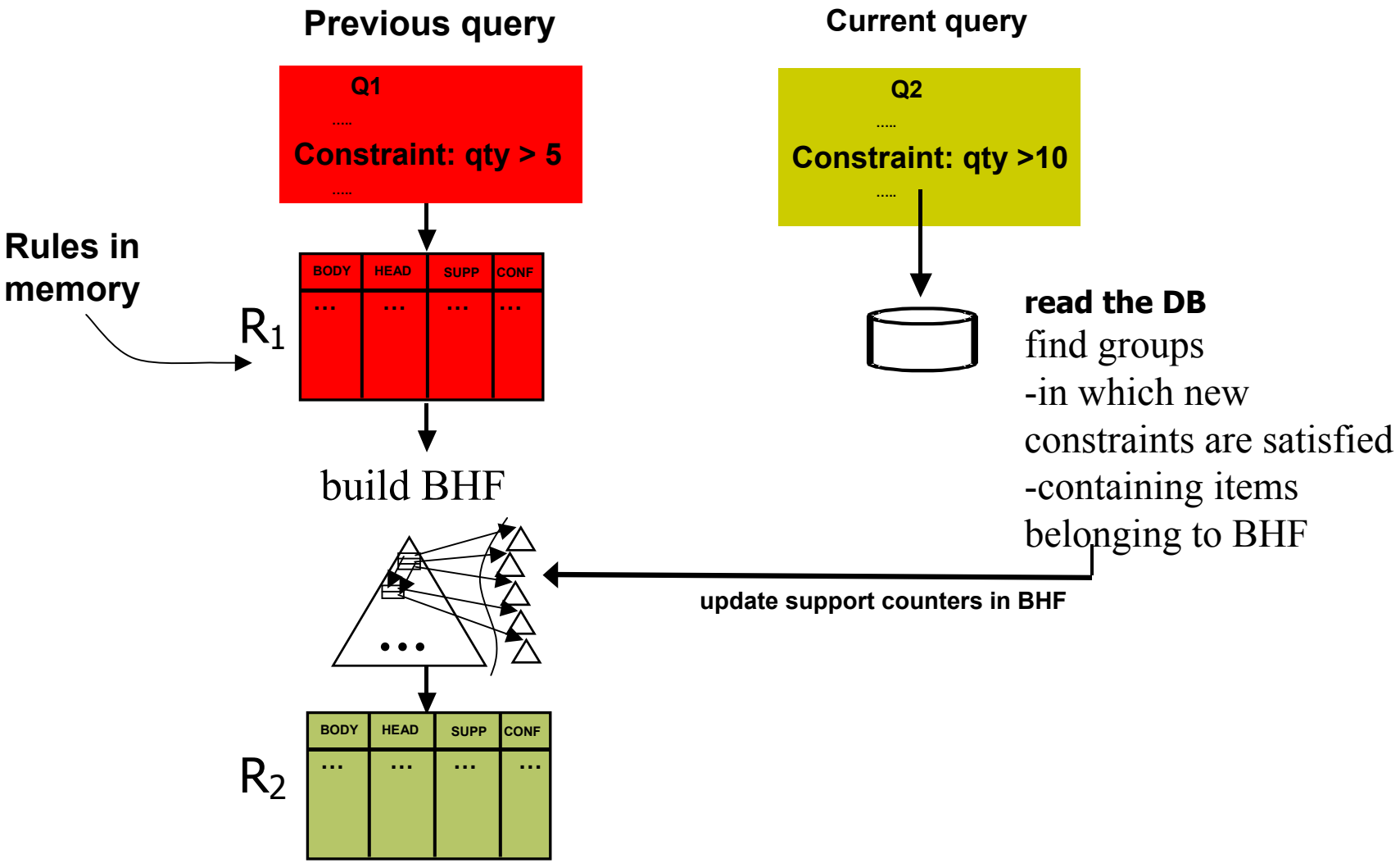
Context dependent constraints



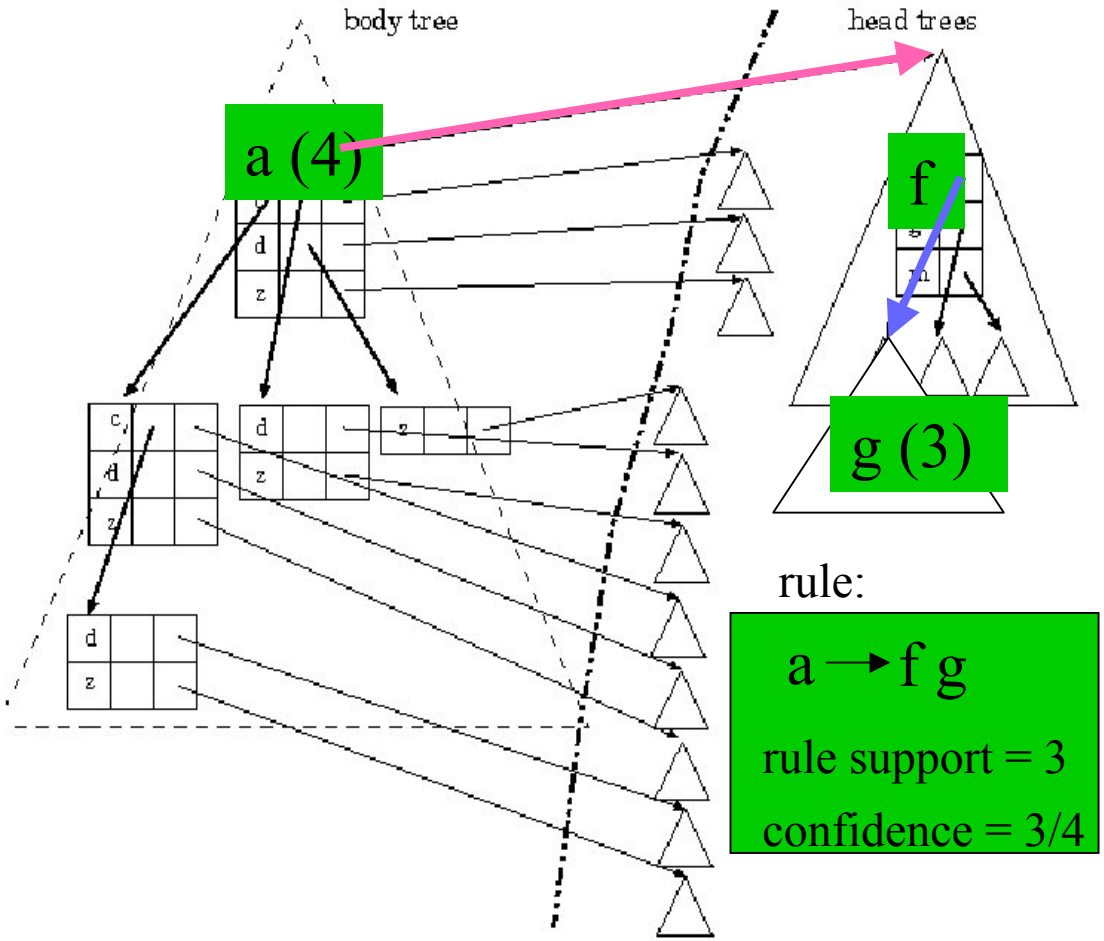
Incremental Algorithm for IDC



Incremental Algorithm for **CDC**



Body-Head Forest (BHF)

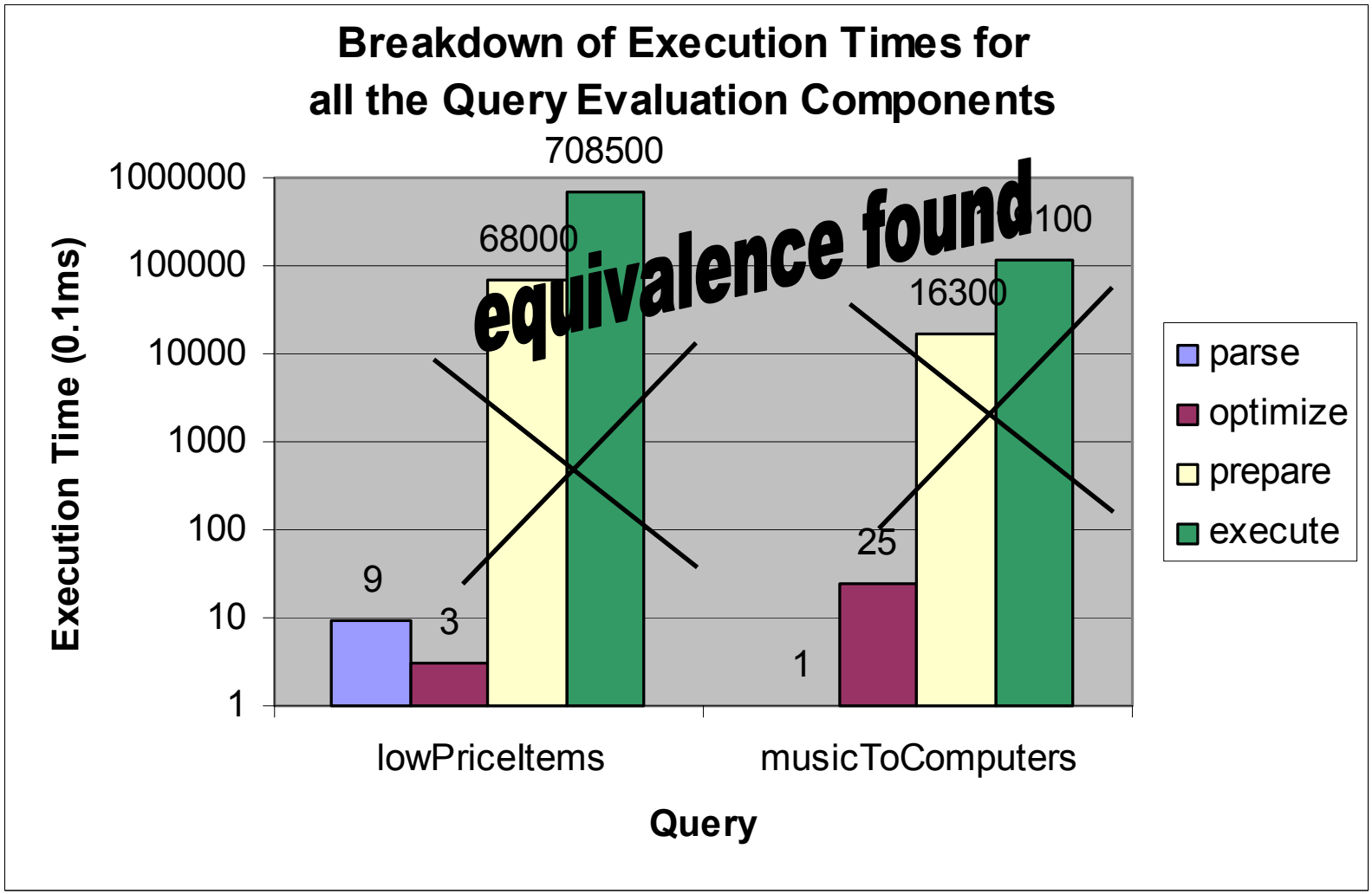


- body (head) tree contains itemsets which are candidates for being in the body (head) part of the rule
- an itemset is represented as a single path in the tree and vice versa
- each path in the body (head) tree is associated to a counter representing the body (rule) support

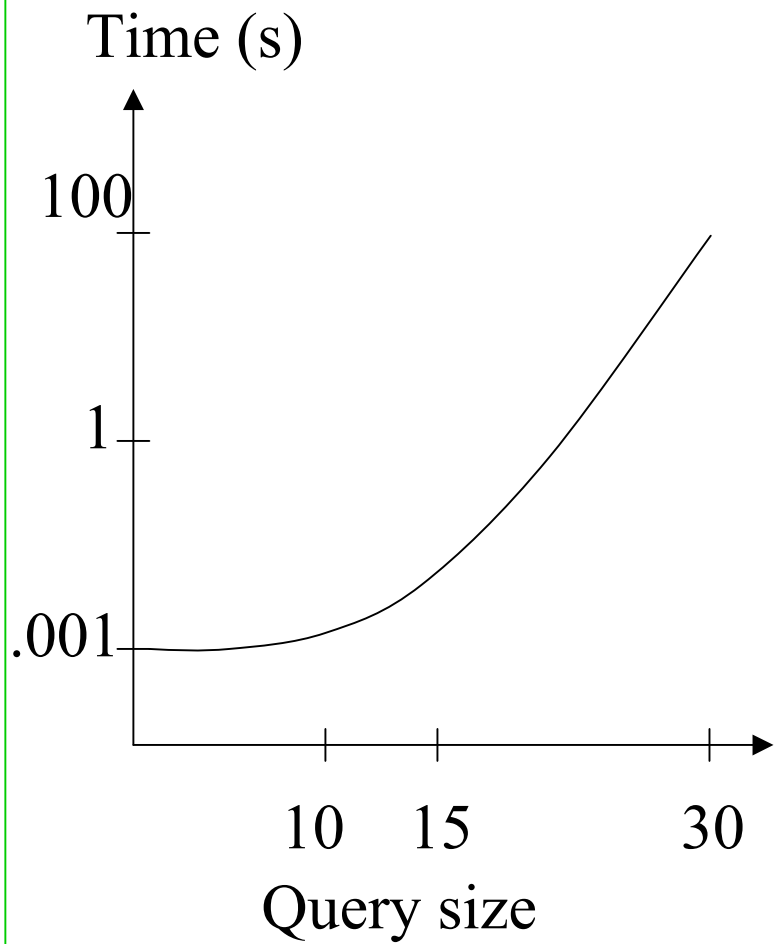
Database for the Experiments

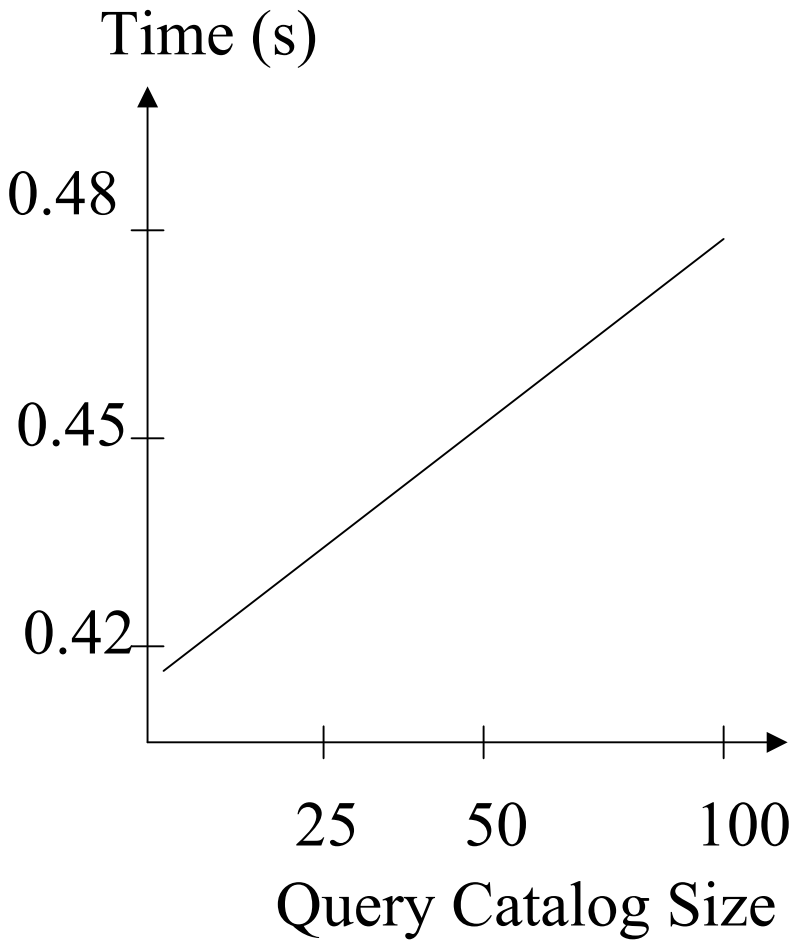
- Retail database
  - Gid, item, price, category, discount, qty, cost
- 125 000 rows
- 10 000 transactions (gid)
- 12.5 items (average) for each transaction
- 22 item categories
- Price uniformly distributed: 100 – 20 000
- price⇔discount (discount=price*0.2)

Experiments on some Typical BI Queries

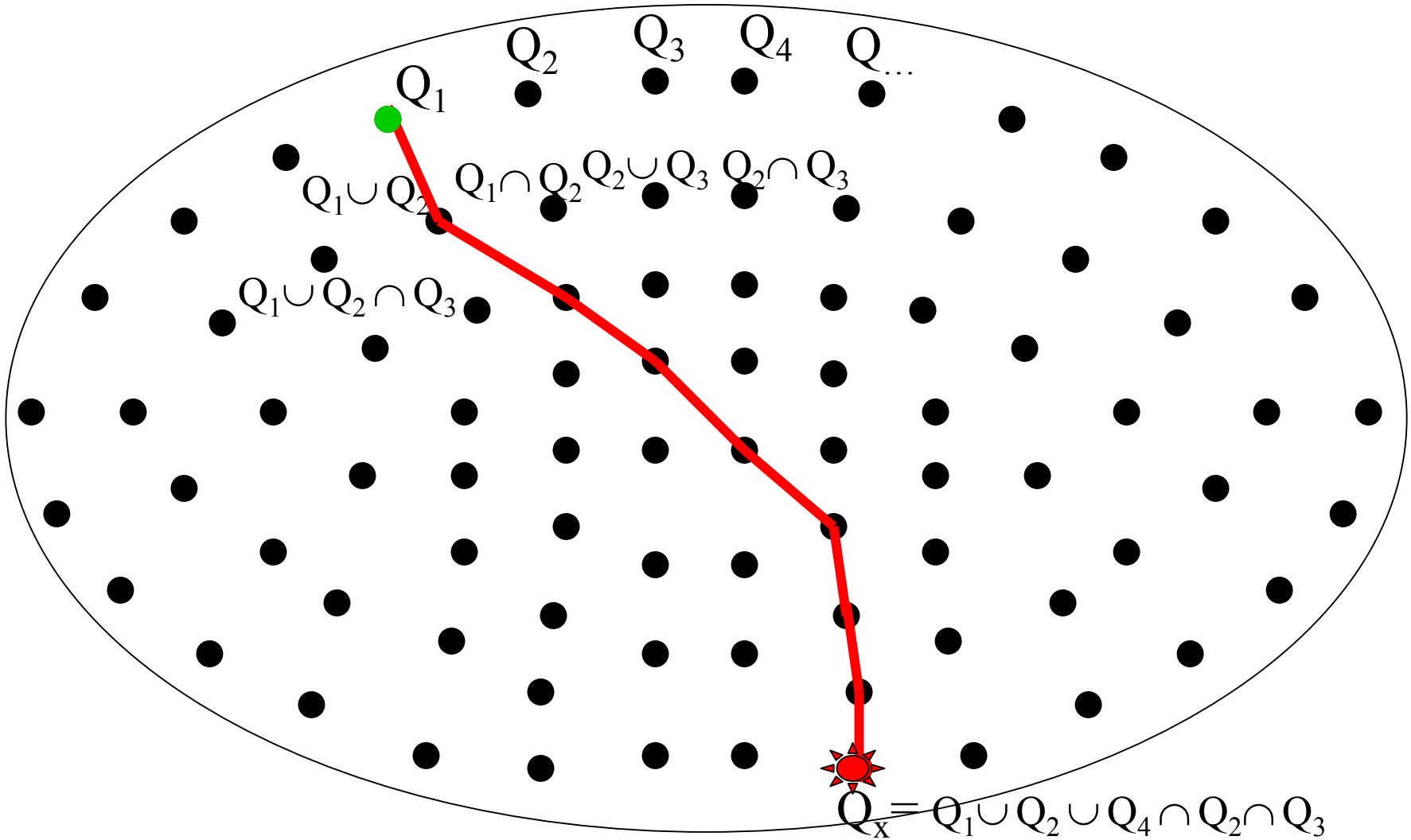


Experiments on the Optimizer

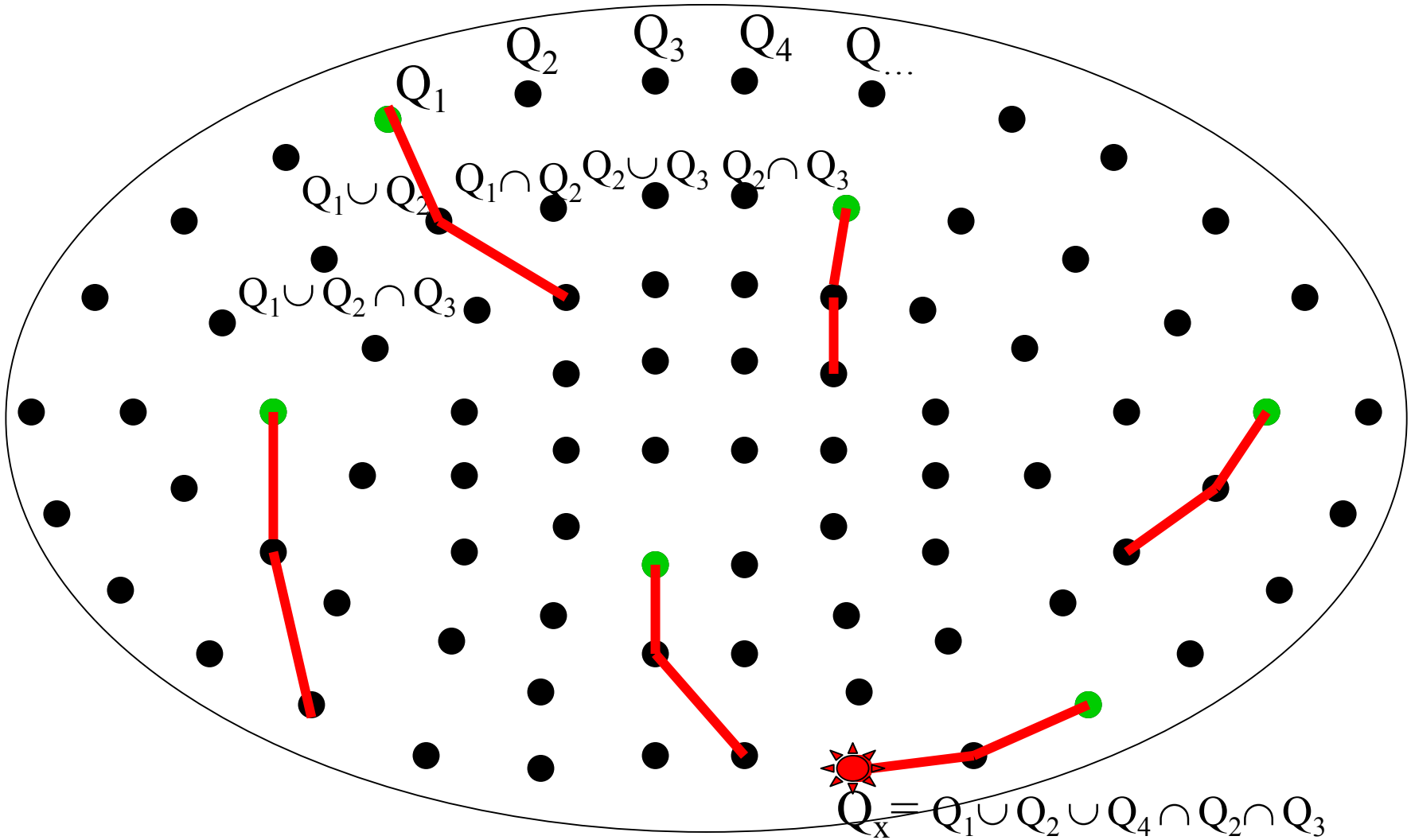




Searching for Equivalence with a Genetic Approach

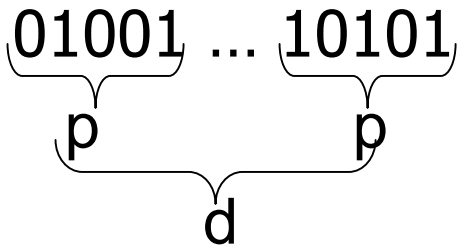


Searching for Equivalence with a Genetic Approach



Searching for Equivalence with a Genetic Approach

- Candidate query rewritings for target query are represented by a bit string (genome). With
  - p: number of predicates
  - d: number of disjuncts
  - genome of a query is: 01001 … 10101

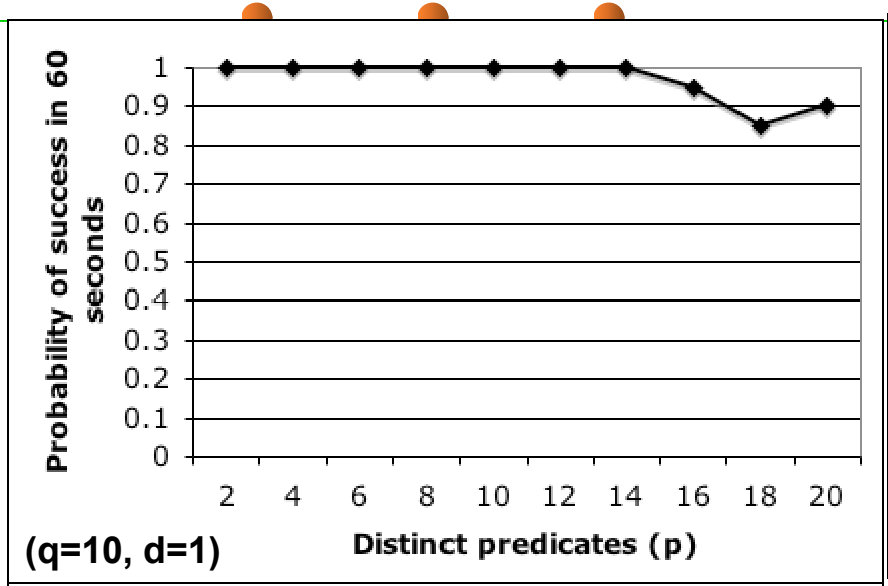


Ex. with predicates A,B,C,D:

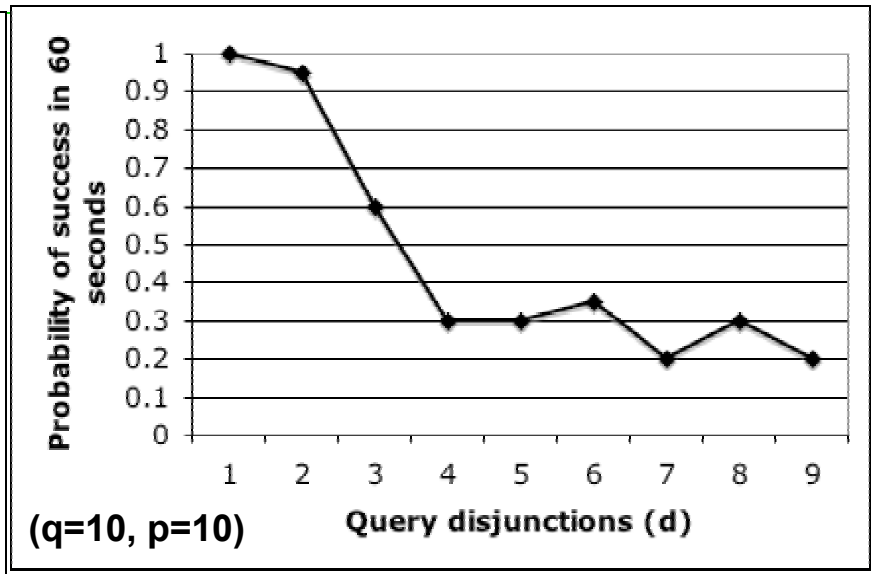
$A \wedge B \vee C \wedge D$ =1100 0011

- Search is guided by the fitness function (Hamming distance between the target query and a candidate query rewriting)

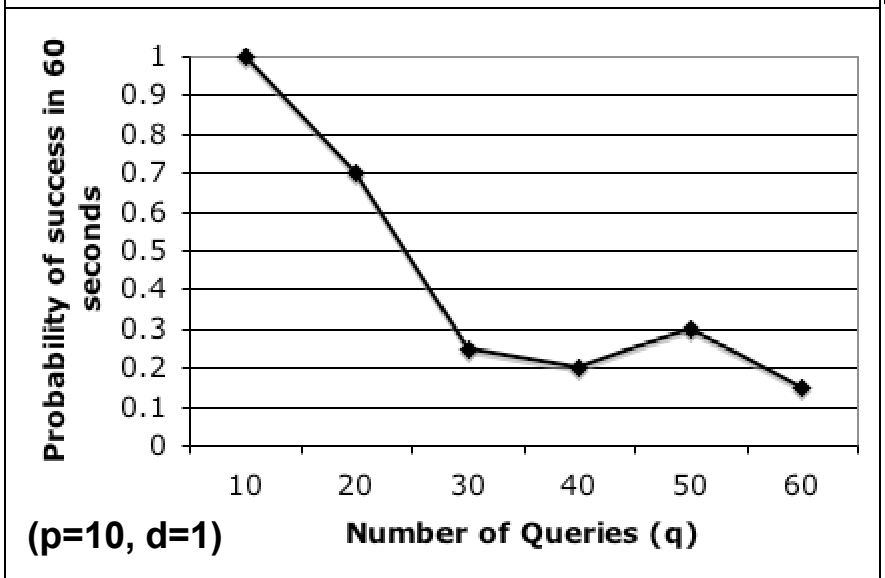
Experiments on the Genetic Search for Equivalence



(q=10, d=1)



(q=10, p=10)



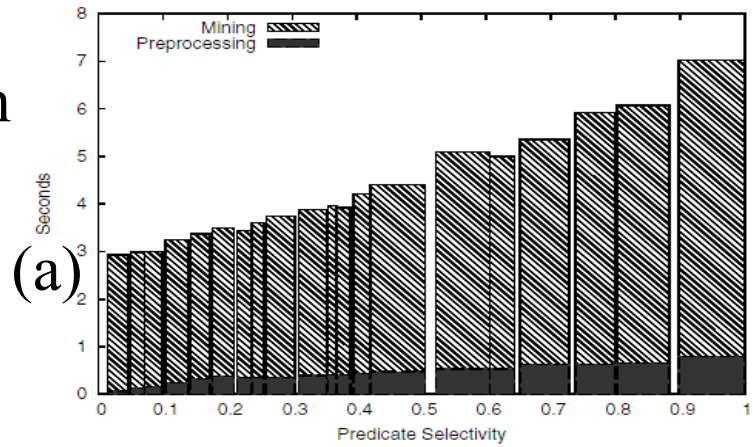
(p=10, d=1)

Experiments on Containment: ID and CD

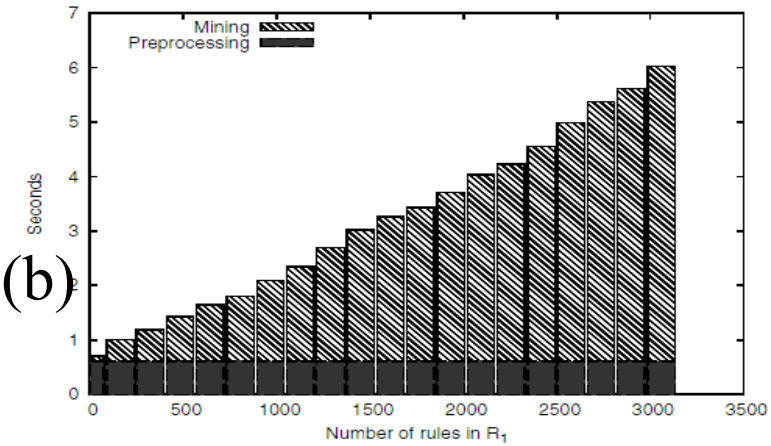
execution time vs constraint selectivity

execution time vs volume of previous result

ID algorithm

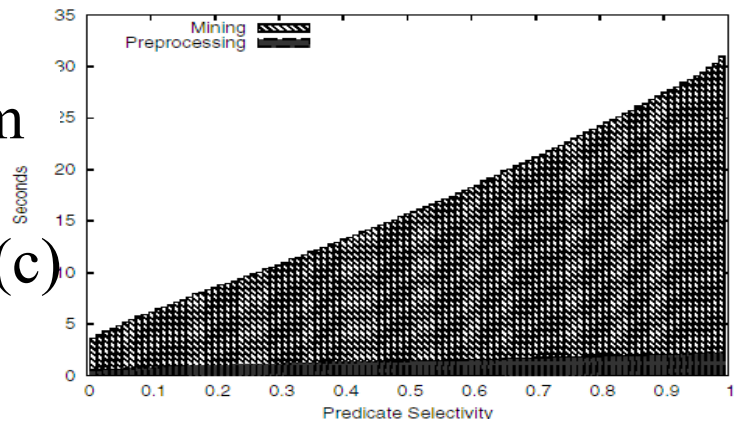


(a)

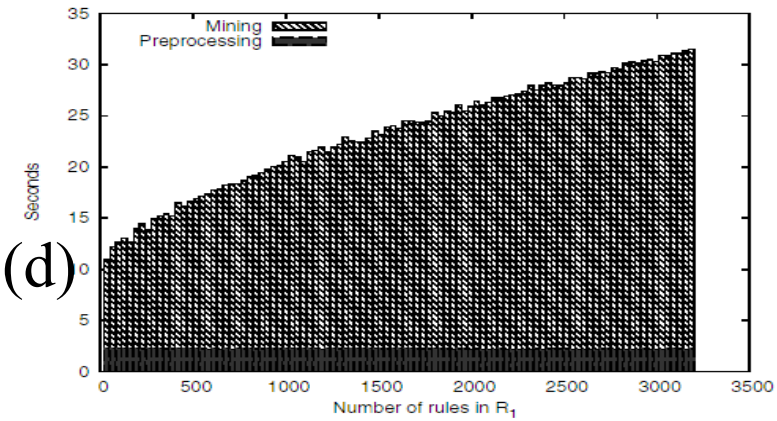


(b)

CD algorithm

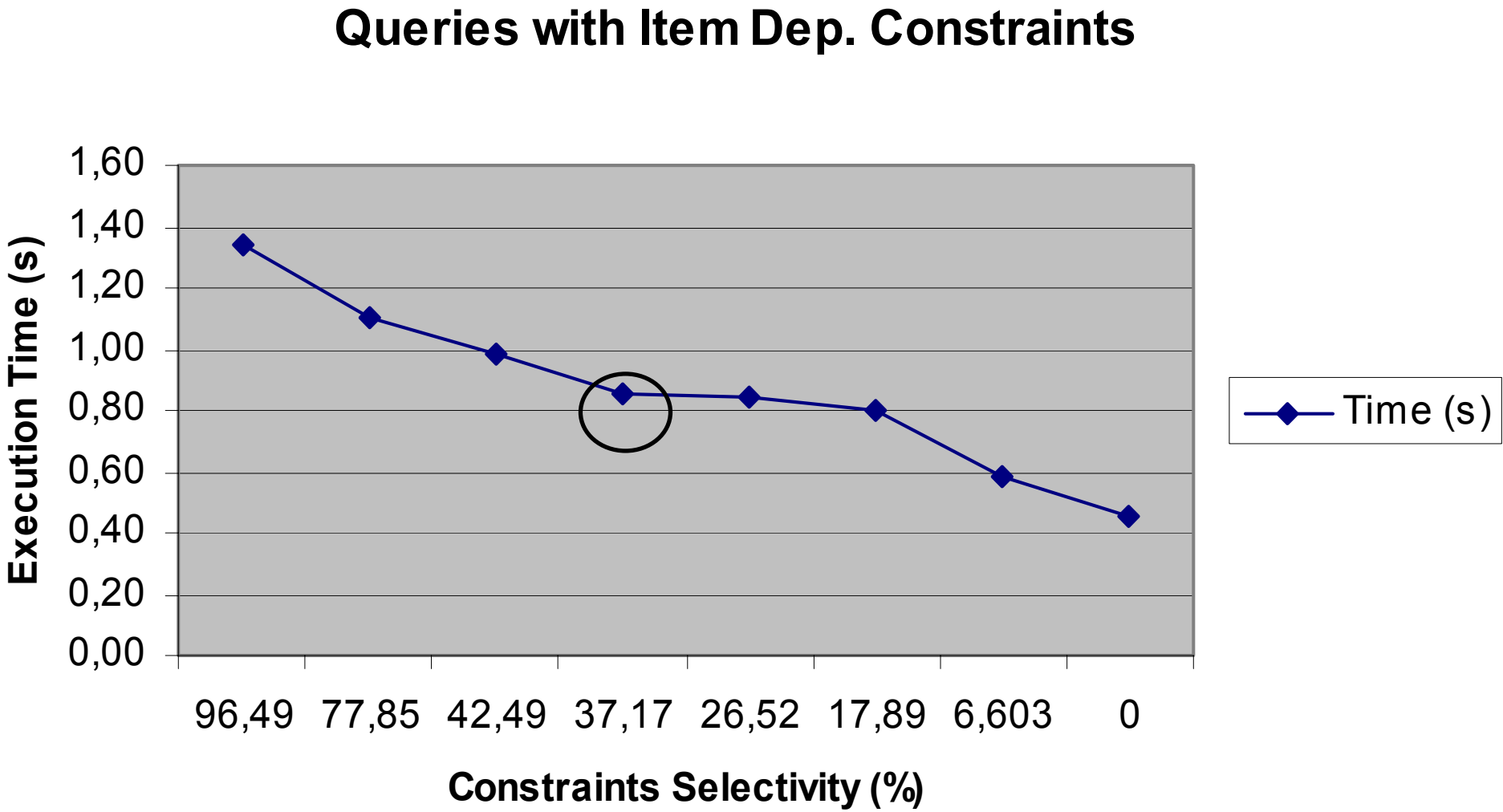


(c)

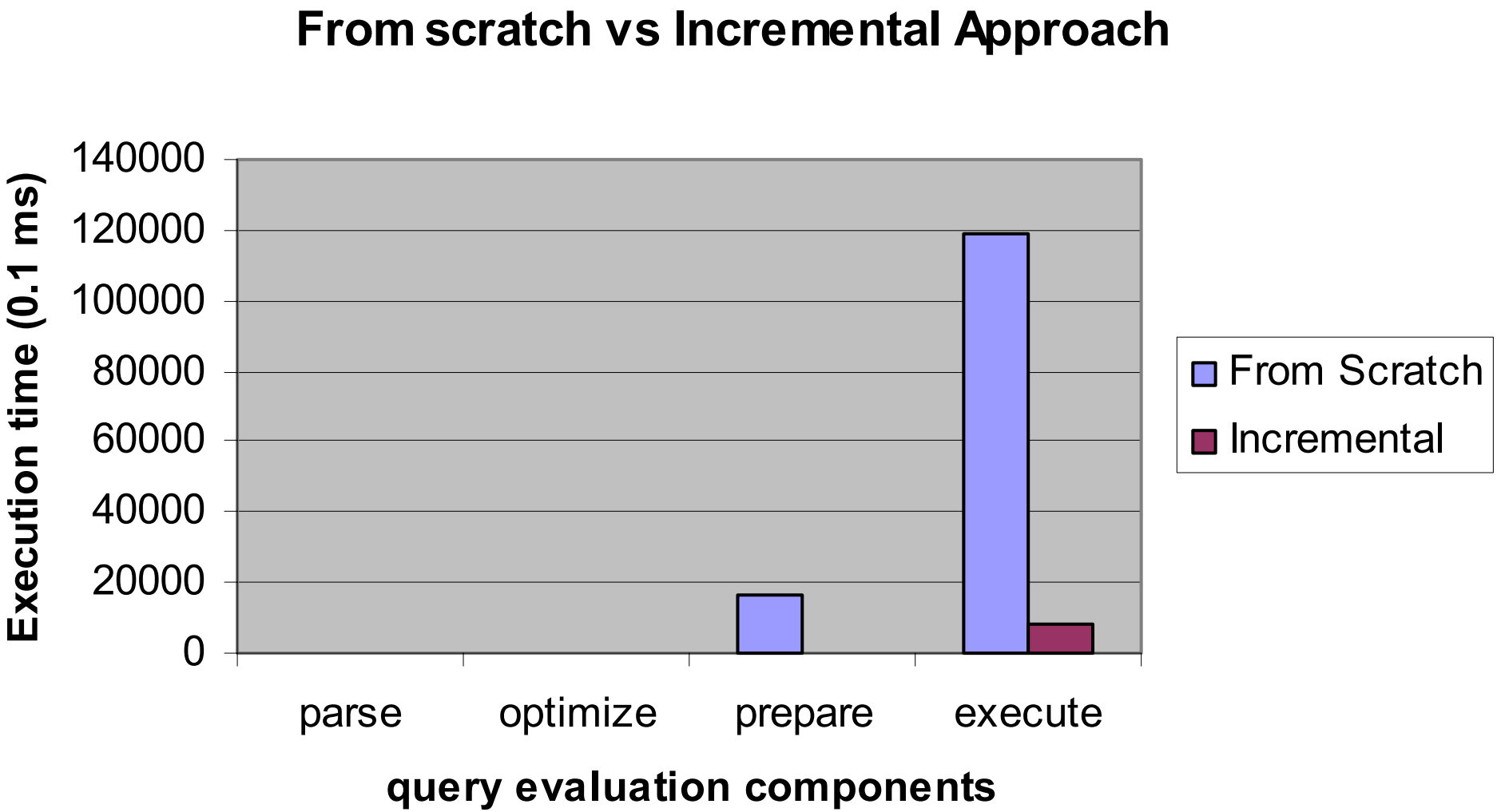


(d)

Experiments on Query Inclusion



Comparison between Traditional vs Incremental



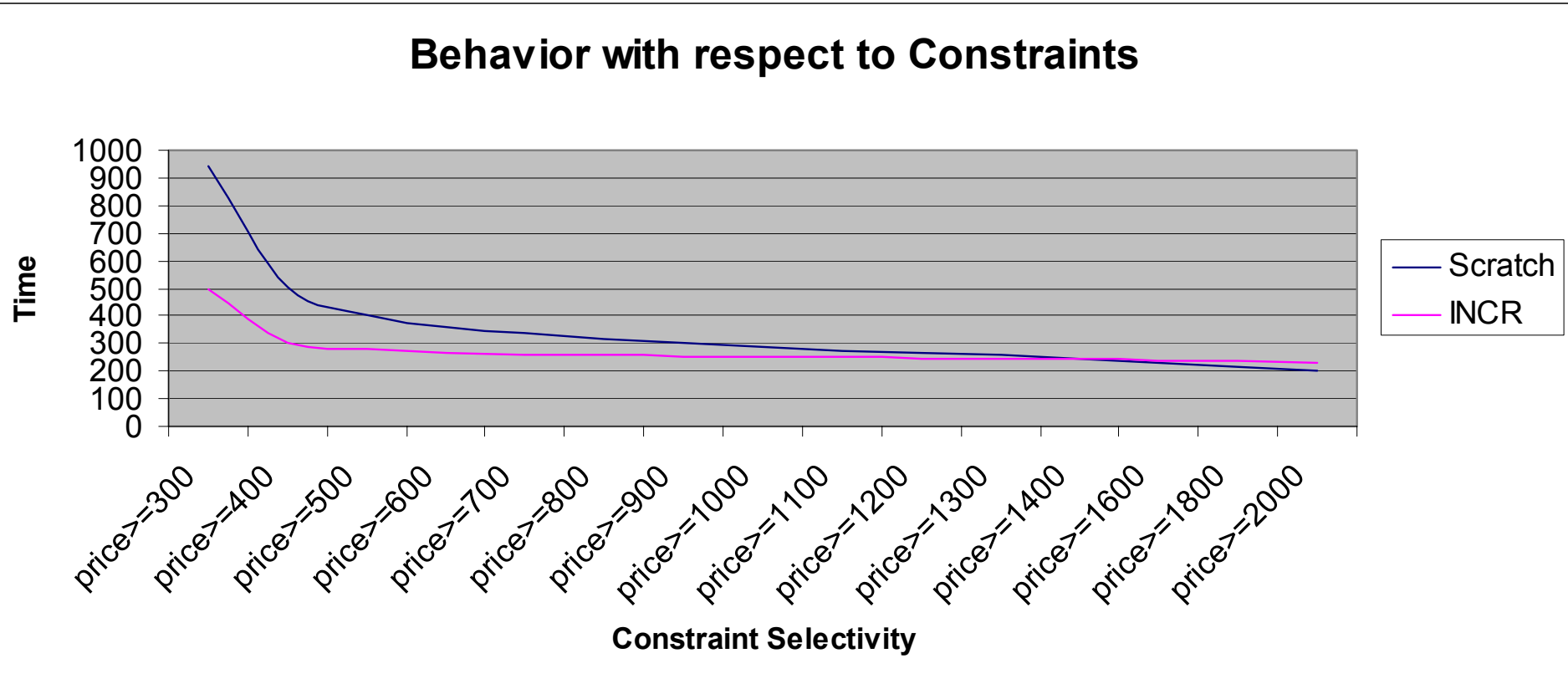
Evaluation of Incremental Algorithm

- **From Scratch:**
  - suitable to treat item and context dep. constraints as well
  - Works in two steps
    1. freq. singletons satisfying constraints
    2. DFS for itemsets generation

- **INCR:**
  - proposed incremental algorithm
  - Works in two steps
    1. Loads in memory previous result R_1
    2. Checks on any $I \in R_1$ the new constraints and update support

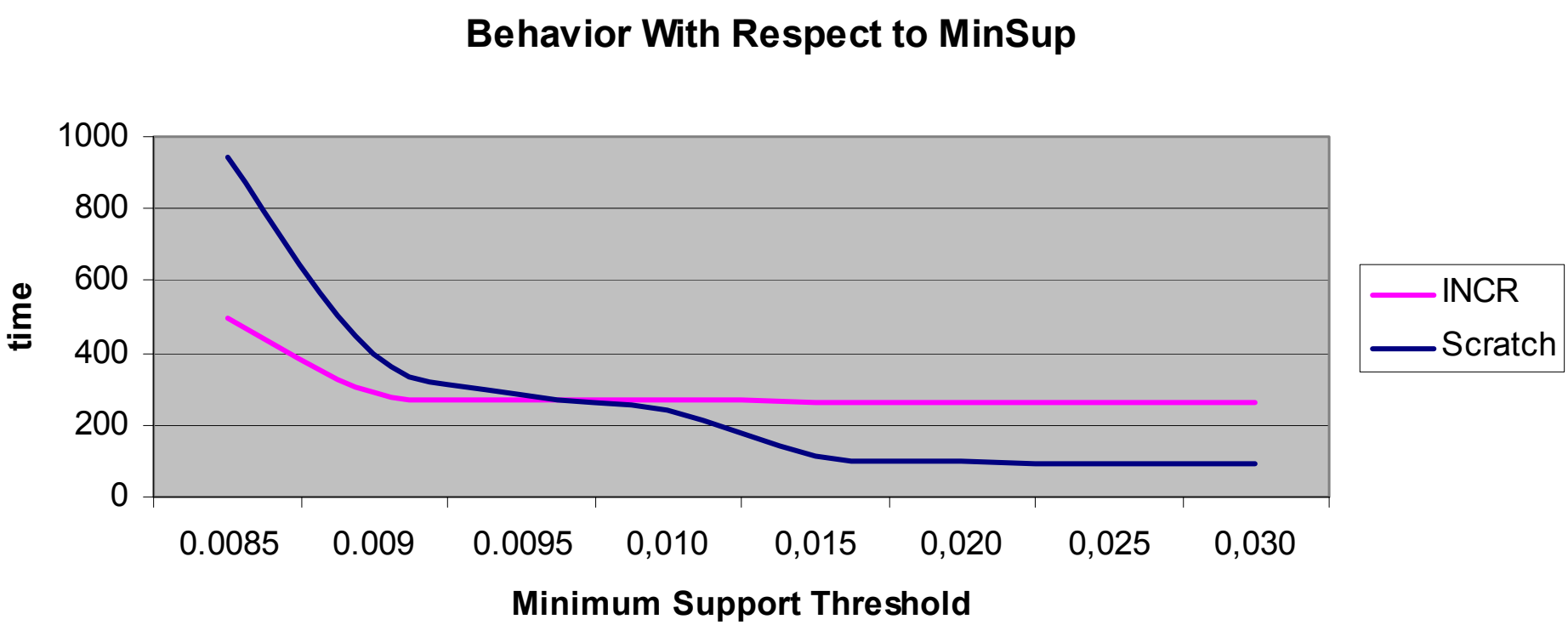
Experiments on Incremental Algorithms (1)

- Evaluation of *Constraints* Factor



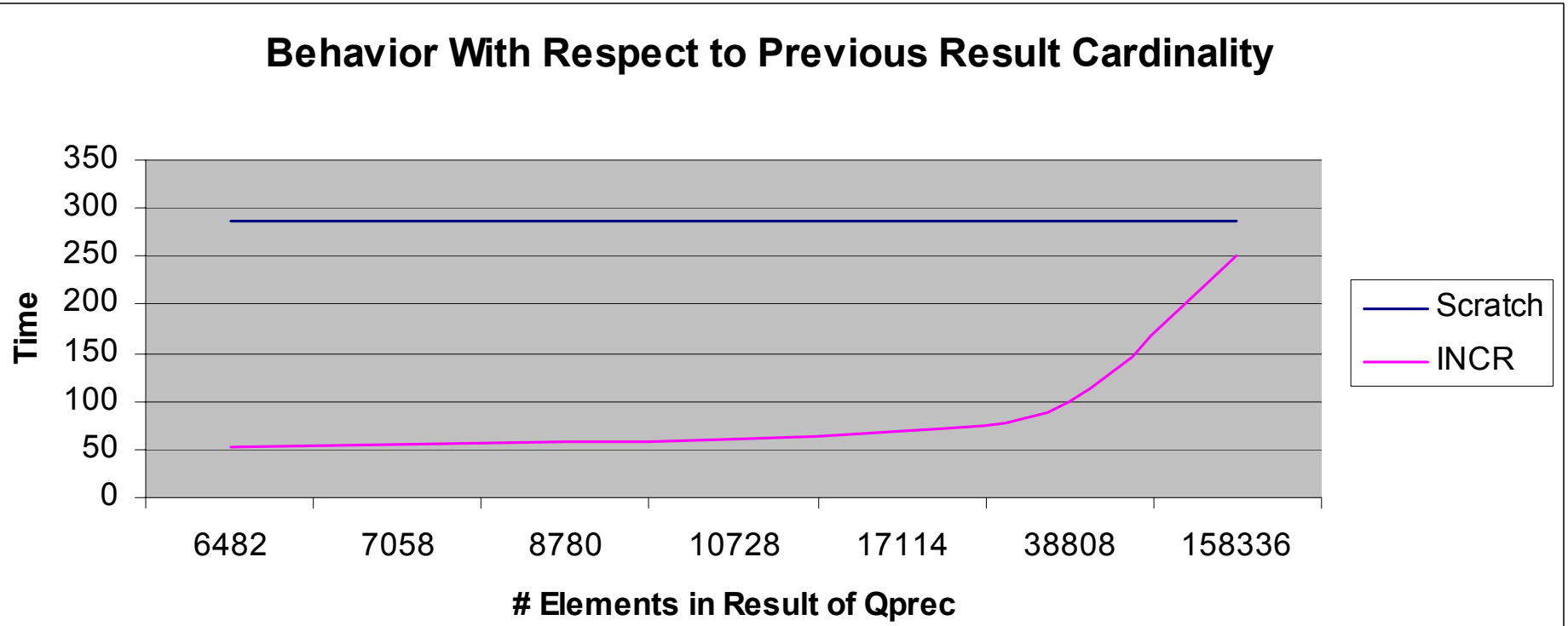
Experiments on Incremental Algorithms (2)

- Evaluation of *MinSup* Factor



Experiments on Incremental Algorithms (3)

- Evaluation of *Result Set Cardinality* Factor



UNIVERSITÀ DEGLI STUDI DI TORINO
ALMA UNIVERSITAS TAURINENSIS

Conclusions

- We have proposed query rewriting for itemset mining as a promising strategy
- We defined item and context dependent constraints and studied their role in query rewriting
- Experiments show that this approach is feasible and in many cases necessary to speed up execution times in inductive databases and data mining.