An Extensible Rule Transformation Model for XQuery Optimization

Rules Pattern for XQuery Tree Graph View

Nicolas Travers[1] Tuyêt Trâm Dang Ngoc[2]

(1) PRiSM Laboratory-University of Versailles, France

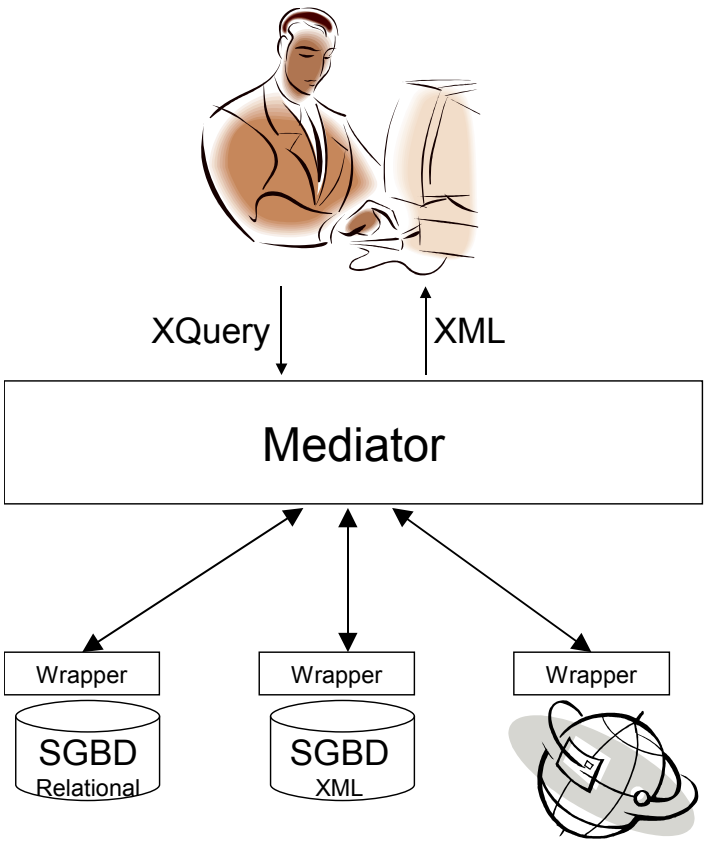
(2) ETIS Laboratory - University of Cergy-Pontoise, France

14 June 2007

ICEIS 2007

Plan

XQuery Evaluation



Data Sources :

- Distributed ;
- Heterogeneous.

Problems :

- XQuery ;
- Data localization ;
- Query optimization ;
- Adaptability.

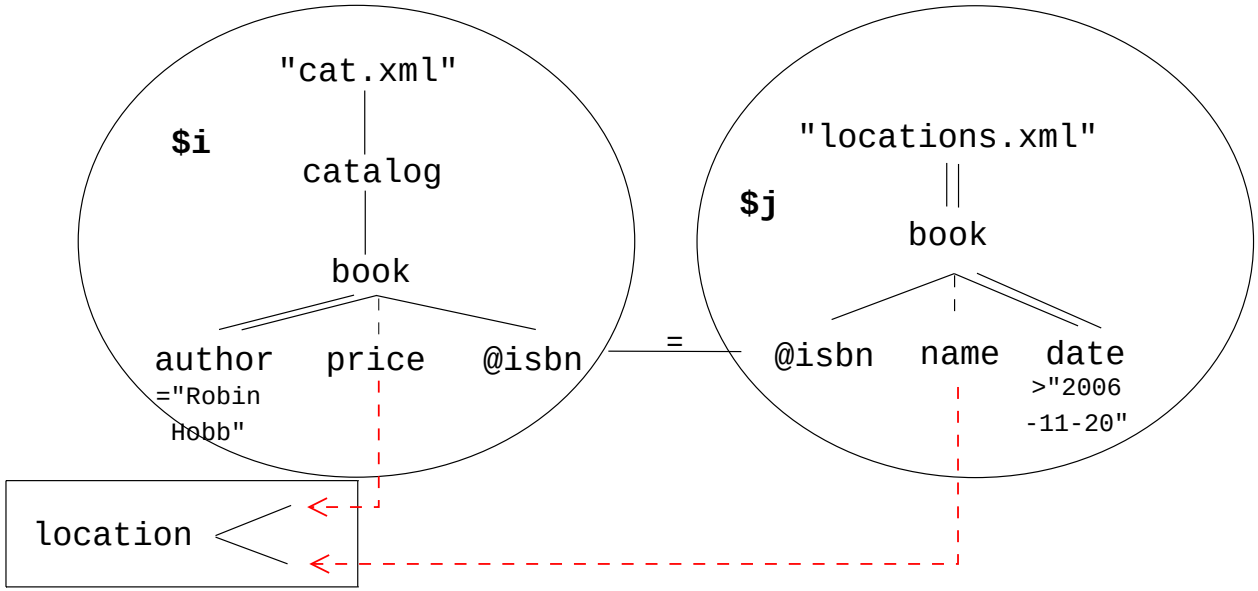
Tree Graph View (TGV) [DASFAA 2007]

TGV : a **query model** for XQuery.

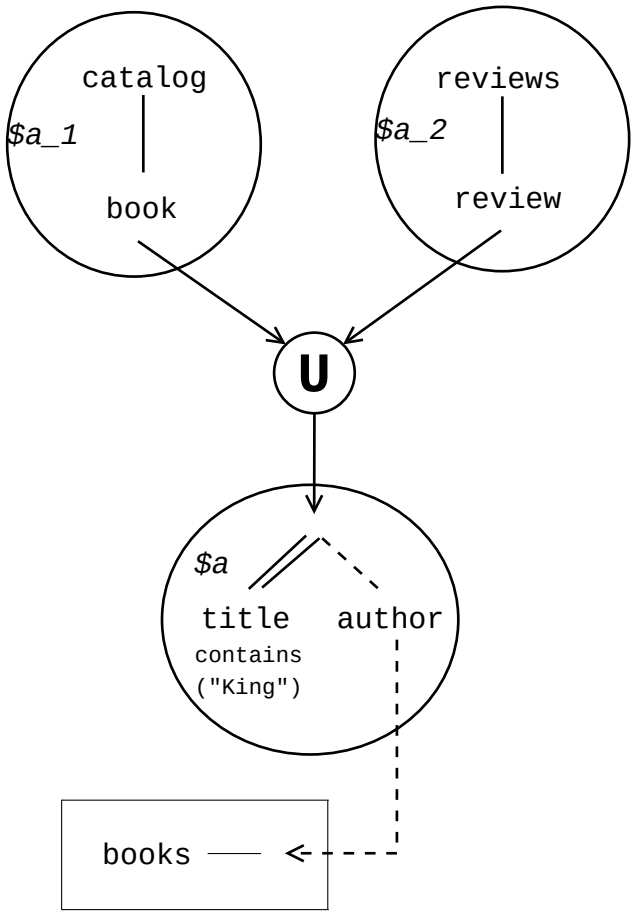
- Based on Tree Pattern matching ;
- Describes selection and construction patterns ;
- Defines projections and associations between patterns ;
- Identifies groups of treatments ;

A TGV is a direct translation of an XQuery query.

TGV examples (Logical TGVs)



TGV examples (Logical TGVs)



Annotation Support (Physical TGVs)

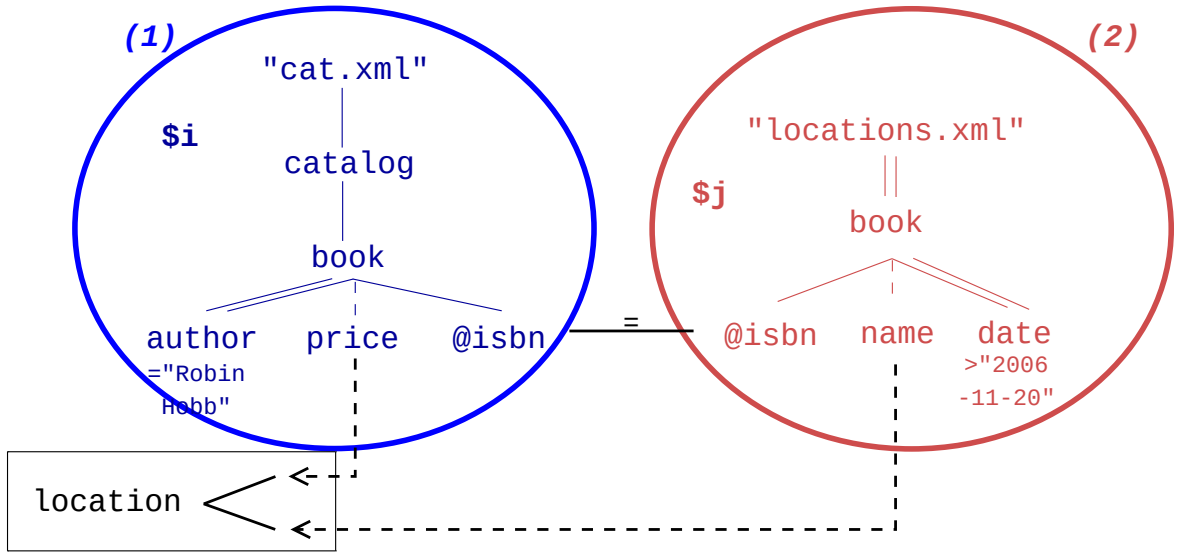
Annotations on TGV :

- Localization.
- Cost model ;

Annotation Support (Physical TGVs)

Annotations on TGV :

- Localization.
- Cost model ;



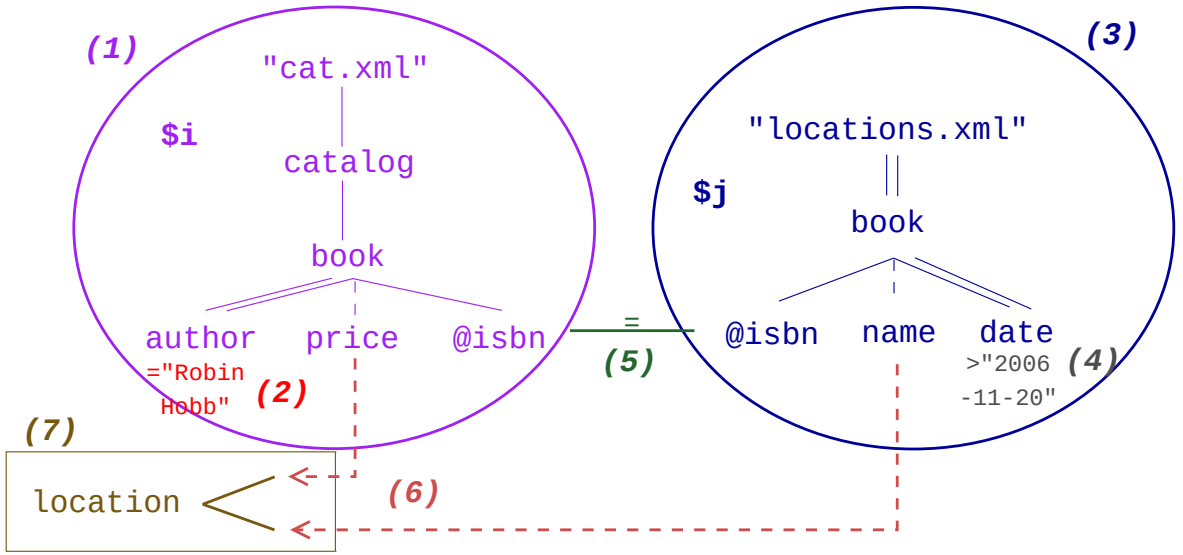
(1) source 1

(2) source 2

Annotation Support (Physical TGVs)

Annotations on TGV :

- Localization.
- Cost model ;



(1) cost_1 = card_i * IO

(2) cost_2 = cost_1 * sel(author)

(3) cost_3 = card_i * IO

(4) cost_4 = cost_3 * sel(date)

(5) cost_5 = CPU * card ((1) x (2))

(6) cost_6 = max (cost_2, cost_4)

(7) cost_7 = const + cost_5 + cost_6

Motivations of TGV optimization

- TGV simplifies XQuery conception, but needs to :
  - Define transformations ;
  - Generate better evaluations ;
- Distributed context :
  - Identify and optimize sub-queries ;
  - Distributed optimization ;
  - Adapt the optimizer (mediation, P2P, mobile agents...).

Motivations of TGV optimization

- TGV simplifies XQuery conception, but needs to :
  - Define transformations ;
  - Generate better evaluations ;
- Distributed context :
  - Identify and optimize sub-queries ;
  - Distributed optimization ;
  - Adapt the optimizer (mediation, P2P, mobile agents...).

An **Extensible Optimizer** corresponds to this needs.

What is needed?

Search Strategy of the best plan, relies on :

- Equivalent representations ;
- Costs types (execution time, communication, price) ;
- Transformation rules on TGVs ;
- Distributed context.

What is needed?

Search Strategy of the best plan, relies on :

- Equivalent representations ;
- Costs types (execution time, communication, price) ;
- Transformation rules on TGVs ;
- Distributed context.

What is needed?

Search Strategy of the best plan, relies on :

- Equivalent representations ;
- Costs types (execution time, communication, price) ;
- Transformation rules on TGVs ;
- Distributed context.

What is needed?

Search Strategy of the best plan, relies on :

- Equivalent representations ;
- Costs types (execution time, communication, price) ;
- Transformation rules on TGVs ;
- Distributed context.

What is needed?

Search Strategy of the best plan, relies on :

- Equivalent representations ;
- Costs types (execution time, communication, price) ;
- Transformation rules on TGVs ;
- Distributed context.

Static and Extensible Optimizers

- Standard optimizers : a set of rules and a strategy ;
- Extensible optimizers : Rules and Strategies can be modified :
  - Emphasize Search Space ;
  - Improve optimizer ;
  - Adapt the system ;

An extensible optimizer needs :

- Transformation rules ;
- Search strategy(ies) ;
- Cost model(s) ;

Static and Extensible Optimizers

- Standard optimizers : a set of rules and a strategy ;
- Extensible optimizers : Rules and Strategies can be modified :
  - Emphasize Search Space ;
  - Improve optimizer ;
  - Adapt the system ;

An extensible optimizer needs :

- Transformation rules ;
- Search strategy(ies) ;
- Cost model(s) ;

Static and Extensible Optimizers

- Standard optimizers : a set of rules and a strategy ;
- Extensible optimizers : Rules and Strategies can be modified :
  - Emphasize Search Space ;
  - Improve optimizer ;
  - Adapt the system ;

An extensible optimizer needs :

- Transformation rules ;
- Search strategy(ies) ;
- Cost model(s) ;

Static and Extensible Optimizers

- Standard optimizers : a set of rules and a strategy ;
- Extensible optimizers : Rules and Strategies can be modified :
  - Emphasize Search Space ;
  - Improve optimizer ;
  - Adapt the system ;

An extensible optimizer needs :

- Transformation rules ;
- Search strategy(ies) ;
- Cost model(s) ;

Static and Extensible Optimizers

- Standard optimizers : a set of rules and a strategy ;
- Extensible optimizers : Rules and Strategies can be modified :
  - Emphasize Search Space ;
  - Improve optimizer ;
  - Adapt the system ;

An extensible optimizer needs :

- Transformation rules ;
- Search strategy(ies) ;
- Cost model(s) ;

Existing Extensible Optimizers

- Exodus [Carey et al. 90]:
  - Search strategy (improvement coefficient);
- Volcano [Graefe et McKenna 93]:
  - Defines search strategies ;
- OPT++ [Kabra et DeWitt 99]:
  - Object Oriented ;
  - Defines search strategies ;
  - *logical* and *physical* optimization.

Existing Extensible Optimizers

- Exodus [Carey et al. 90]:
  - Search strategy (improvement coefficient);
- Volcano [Graefe et McKenna 93]:
  - Defines search strategies ;
- OPT++ [Kabra et DeWitt 99]:
  - Object Oriented ;
  - Defines search strategies ;
  - *logical* and *physical* optimization.

Existing Extensible Optimizers

- Exodus [Carey et al. 90]:
  - Search strategy (improvement coefficient);
- Volcano [Graefe et McKenna 93]:
  - Defines search strategies ;
- OPT++ [Kabra et DeWitt 99]:
  - Object Oriented ;
  - Defines search strategies ;
  - *logical* and *physical* optimization.

Rule Patterns

We propose a framework for defining and modeling rules :

Condition Rule Pattern => Conclusion Rule Pattern

Rule Patterns : must be found into a TGV.

Two types of transformation :

- Logicals ;
- Physicals ;

Transformation Rules

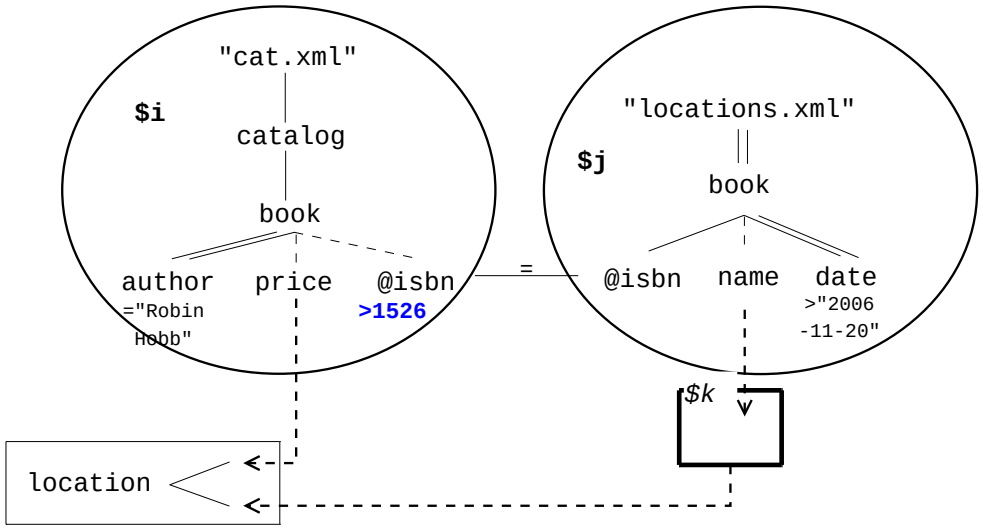
Logical Optimization : predicate duplication

R1 : 

Transformation Rules

Logical Optimization : predicate duplication

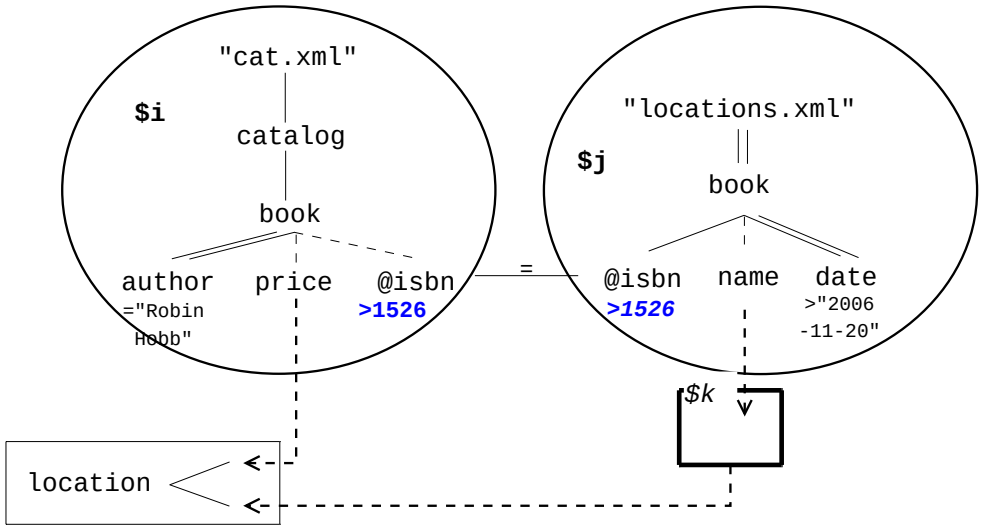
R1 : n_{1} —(c_{1})— =(H_1) —(c_{2})— n_{2} => n_{1} —(c_{1})— =(H_1) —(c_{2})— n_{2}



Transformation Rules

Logical Optimization : predicate duplication

R1 : n_{c_1} —(=, H_1)— n_{c_2} => n_{c_1} —(=, H_1)— n_{c_2}



Transformation Rules

Physical Optimization : Algorithm modification

R2 : 

(1) Left Outer Join
(2) Card
(3) Card

(2) << *(3)*

(1) Bind Left Outer Join
(2) Card
(3) Card

Transformation Rules

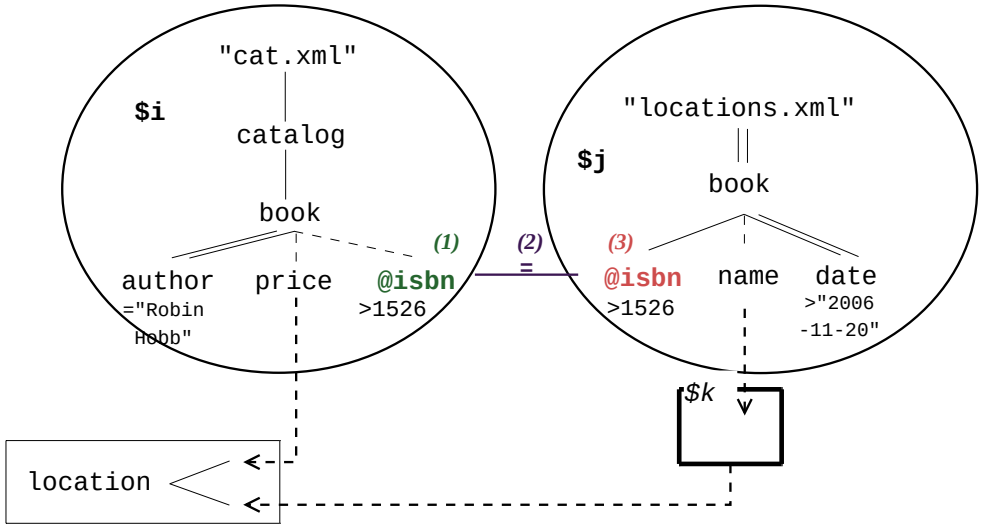
Physical Optimization : Algorithm modification

R2 : $n_1 —(=, $H_1)— $n_2 (2) (1) (3) => $n_1 —(=, $H_1)— $n_2 (2) (1) (3)

(1) Left Outer Join
(2) Card
(3) Card

(2) << (3)

(1) Bind Left Outer Join
(2) Card
(3) Card



(1) Card : 10 *(2)* **Algorithm : Left Outer-Join**
(3) Card : 10000

Transformation Rules

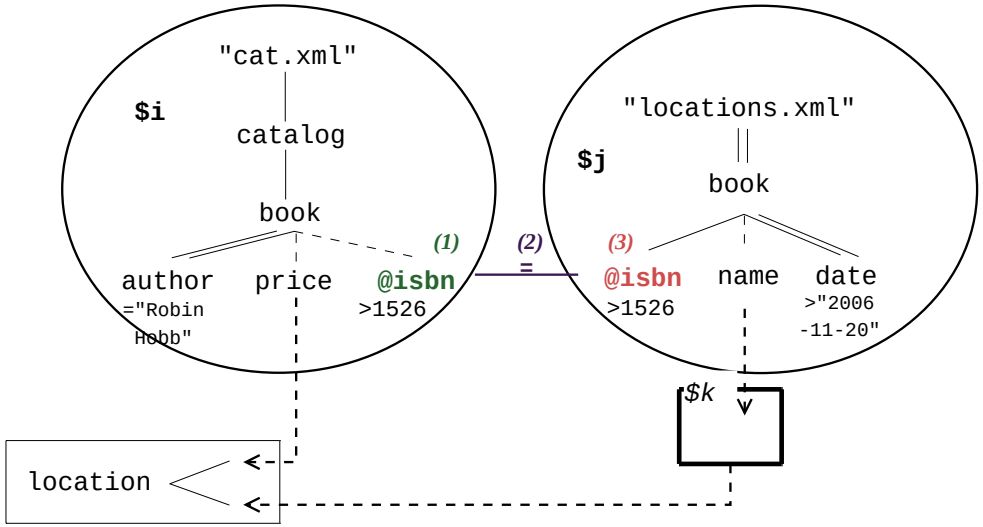
Physical Optimization : Algorithm modification

R2 : $n_1 —(=, $H_1)— $n_2 (2) (1) (3) => $n_1 —(=, $H_1)— $n_2 (2) (1) (3)

(1) Left Outer Join
(2) Card
(3) Card

(2) << (3)

(1) Bind Left Outer Join
(2) Card
(3) Card



(1) Card : 10 *(2)* **Algorithm : Bind Left Outer-Join**

(3) Card : 10000

Search Strategy

Heuristic : **improvement coefficient** on each rule.

- Estimation on : before/after transformation;
- Determined by rules calibration ;

Strategy to direct space generation :

- Incremental ;
- Choose the best applicable coefficient ;

Search Strategy

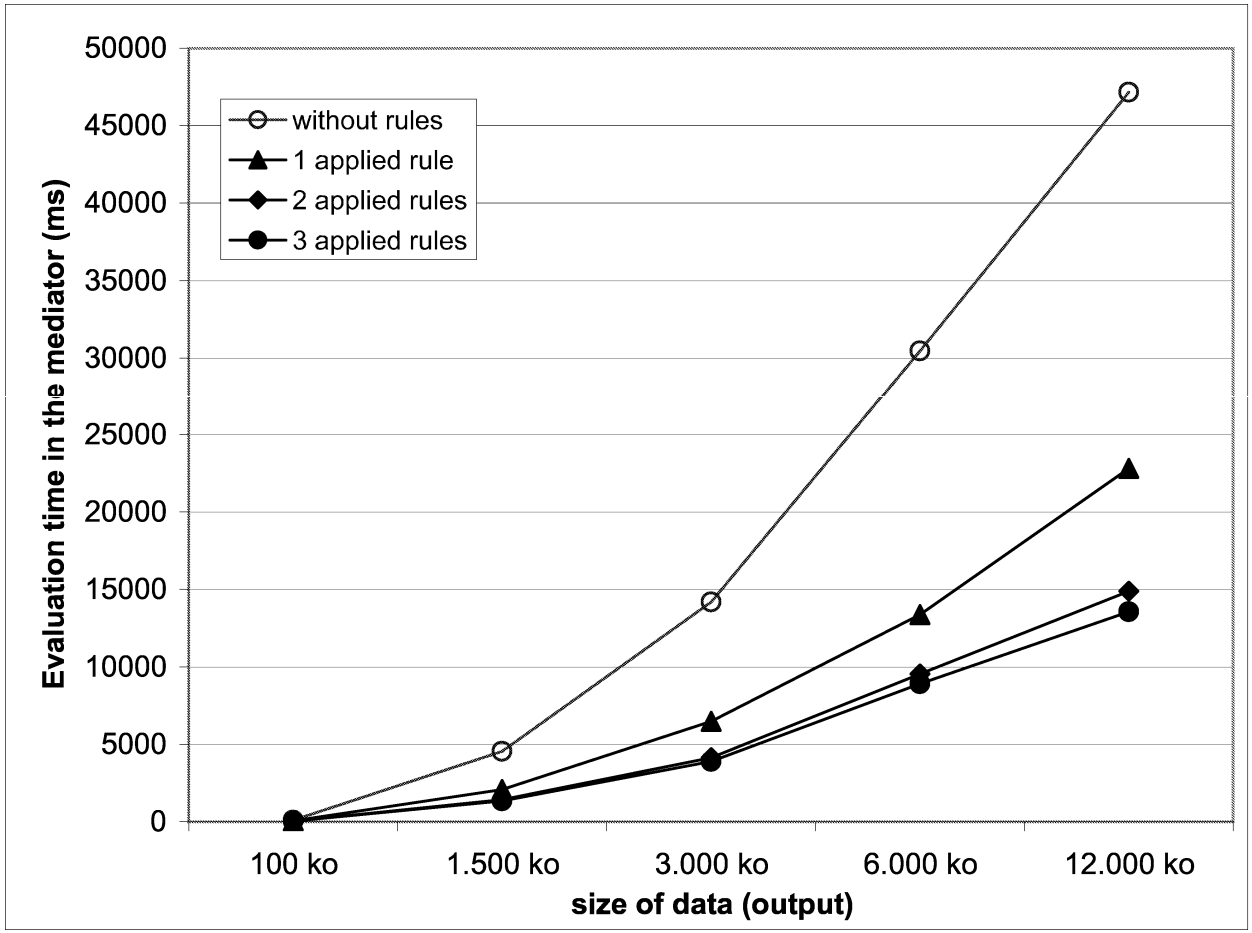
Heuristic : **improvement coefficient** on each rule.

- Estimation on : before/after transformation;
- Determined by rules calibration ;

Strategy to direct space generation :

- Incremental ;
- Choose the best applicable coefficient ;

Rules Application



Conclusion

- Rule Patterns :
  - A framework to define rules ;
  - Makes the optimizer extensible and adaptive ;
  - Rely on TGV (intuitive) ;
  - Rules definition [Cherniak & Zdonik 1998, Lohman 1988, Ali & Moerkotte 2004] ;
- Search Strategy :
  - Improvement Coefficient ;
  - Incremental.