Optimisation Extensible dans un Médiateur de Données Semi-Structurées

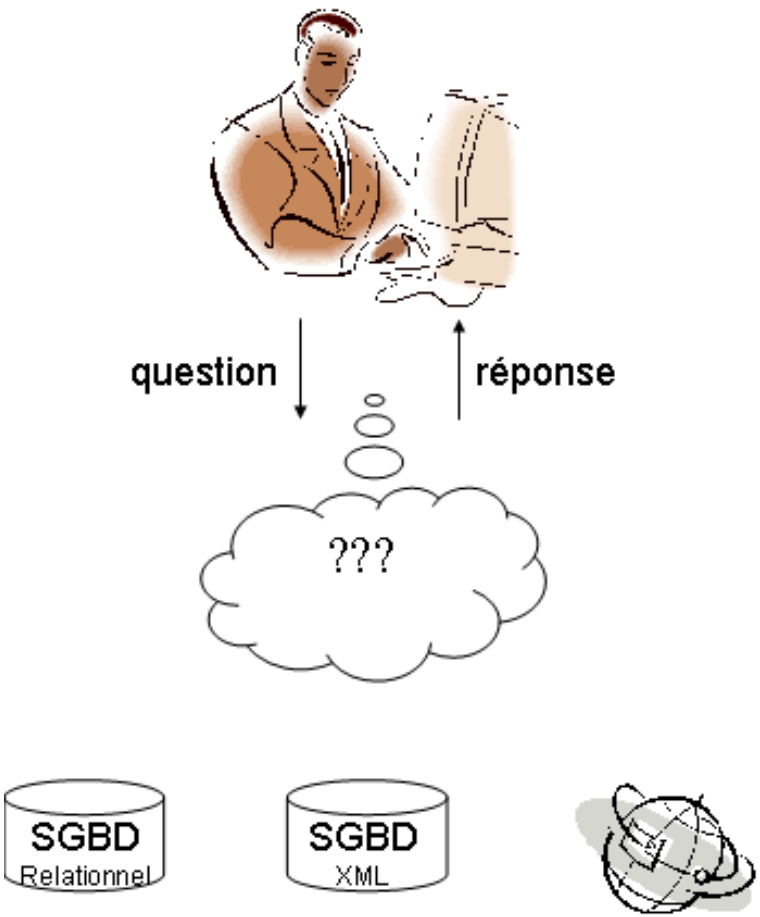
Nicolas Travers

Laboratoire PRiSM
Université de Versailles

12 décembre 2006

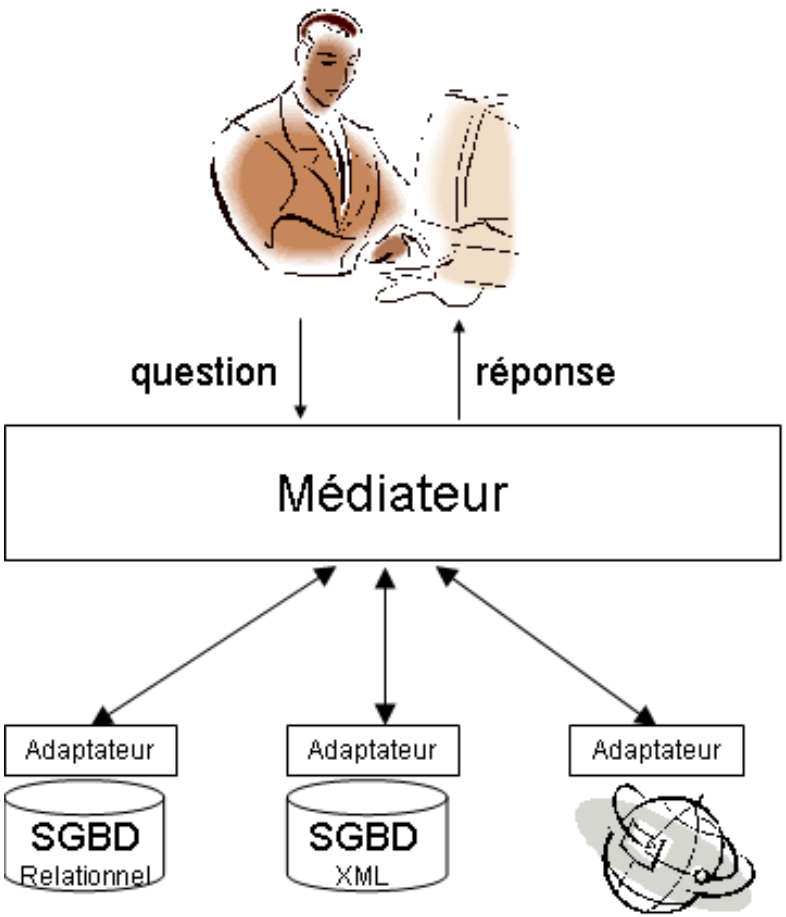
PRiSM

Médiation de données



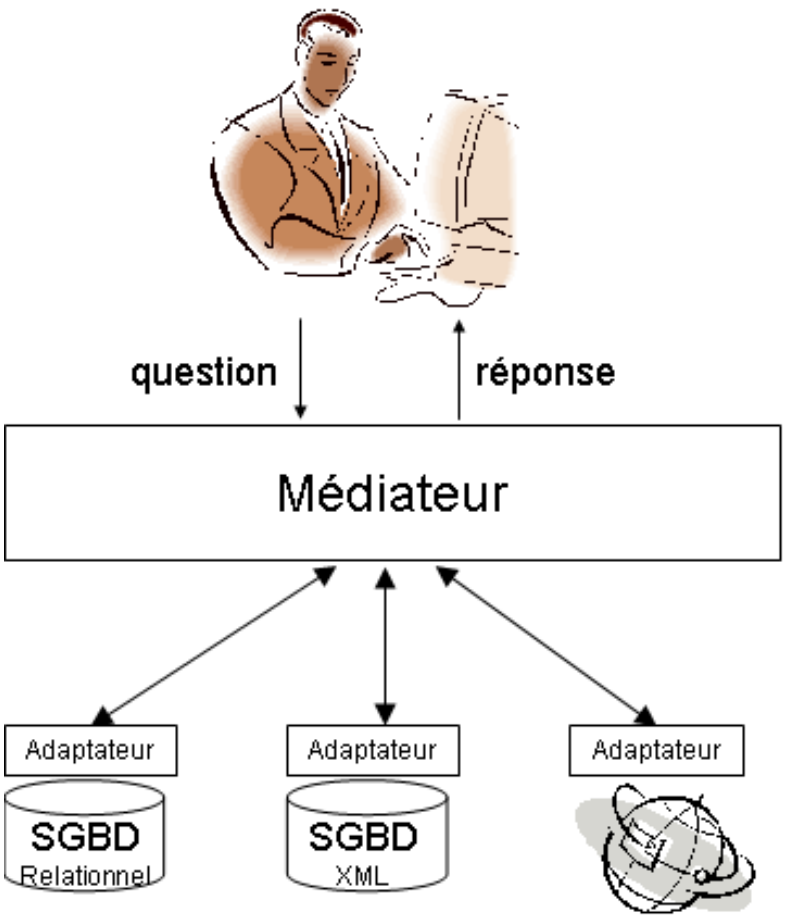
- Sources distribuées :
- Sources hétérogènes :

Médiation de données



- Sources `distribuées` : *Médiateur* ;
- Sources `hétérogènes` : *Adaptateurs*.

Médiation de données



- Sources `distribuées` : *Médiateur* ;
- Sources `hétérogènes` : *Adaptateurs*.

Besoin de :

- Intégrer les données ;
- Utiliser un langage unifié ;
- Produire des résultats efficacement ;
- Passer à l'échelle.

XML

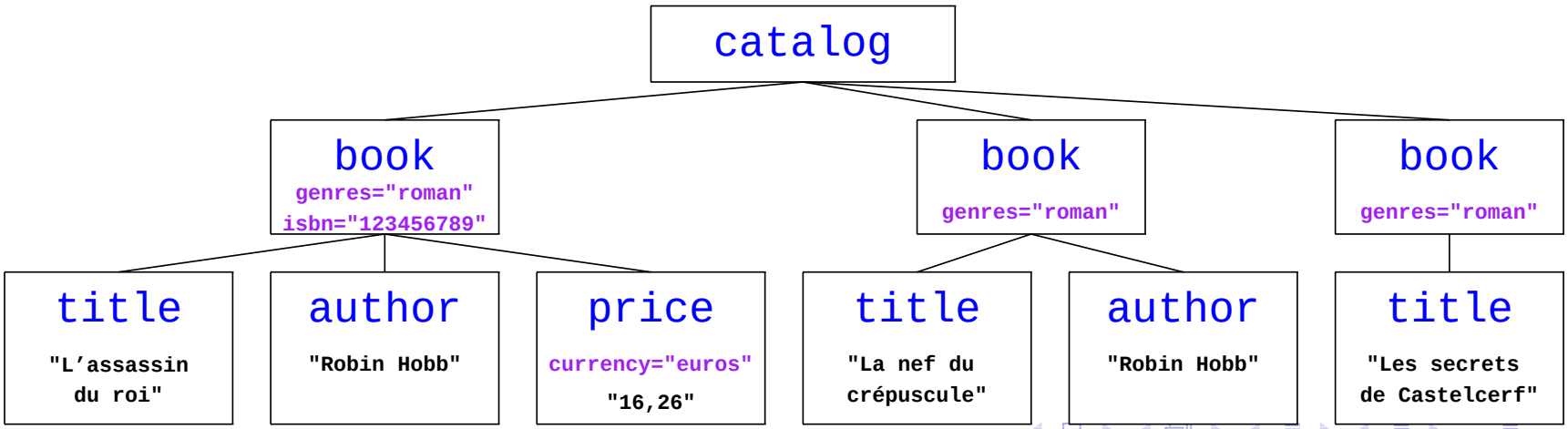
```
<catalog>
  <book genres="roman" isbn="123456789">
    <title> L'assassin du roi </title>
    <author>Robin Hobb </author>
    <price currency="euros"> 16,26 </price>
  </book>
  <book genres="roman">
    <title> La nef du crépuscule </title>
    <author>Robin Hobb </author>
  </book>
  <book genres="roman">
    <title> Les secrets de Castelcerf </title>
  </book>
</catalog>
```

- Données et structures ensembles ;
- Structure variable ;
- Bonne expressivité ;
- Structure arborescente ;

XML

```
<catalog>
  <book genres="roman" isbn="123456789">
    <title> L'assassin du roi </title>
    <author>Robin Hobb </author>
    <price currency="euros"> 16,26 </price>
  </book>
  <book genres="roman">
    <title> La nef du crépuscule </title>
    <author>Robin Hobb </author>
  </book>
  <book genres="roman">
    <title> Les secrets de Castelcerf </title>
  </book>
</catalog>
```

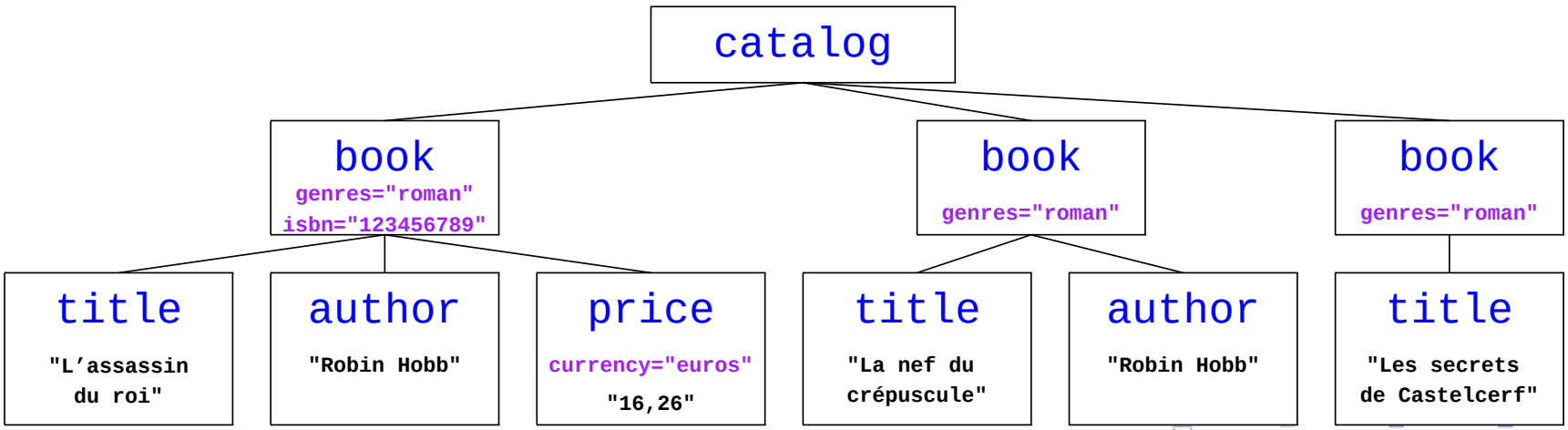
- Données et structures ensembles ;
- Structure variable ;
- Bonne expressivité ;
- Structure arborescente ;



XQuery

```
<catalog>
  <book genres="roman" isbn="123456789">
    <title> L'assassin du roi </title>
    <author>Robin Hobb </author>
    <price currency="euros"> 16,26 </price>
  </book>
  <book genres="roman">
    <title> La nef du crépuscule </title>
    <author>Robin Hobb </author>
  </book>
  <book genres="roman">
    <title> Les secrets de Castelcerf </title>
  </book>
</catalog>
```

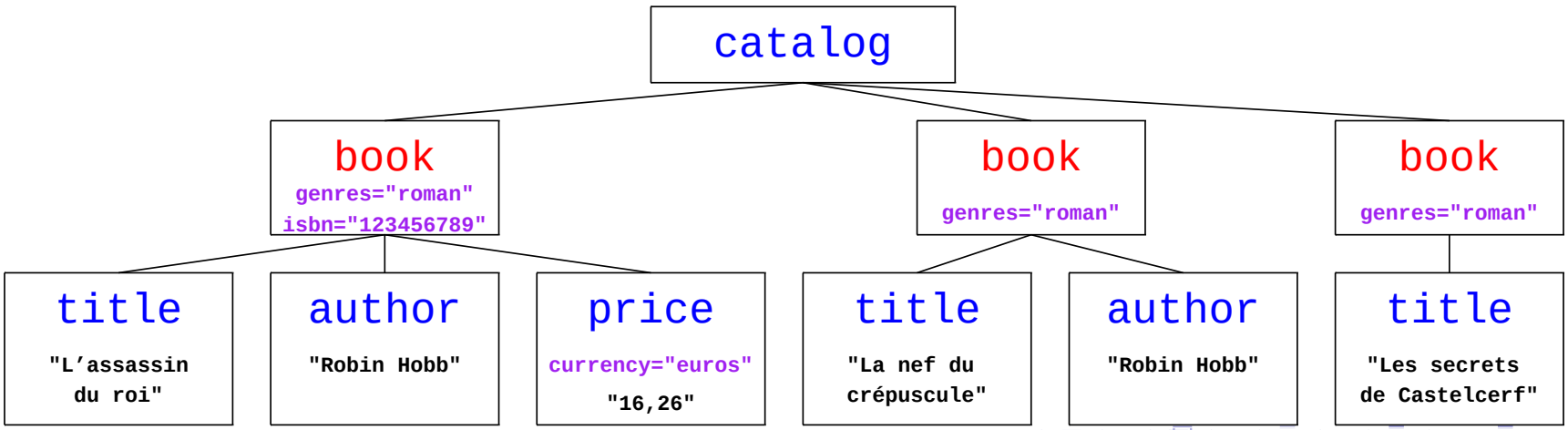
```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```



XQuery

```
<catalog>
  <book genres="roman" isbn="123456789">
    <title> L'assassin du roi </title>
    <author>Robin Hobb </author>
    <price currency="euros"> 16,26 </price>
  </book>
  <book genres="roman">
    <title> La nef du crépuscule </title>
    <author>Robin Hobb </author>
  </book>
  <book genres="roman">
    <title> Les secrets de Castelcerf </title>
  </book>
</catalog>
```

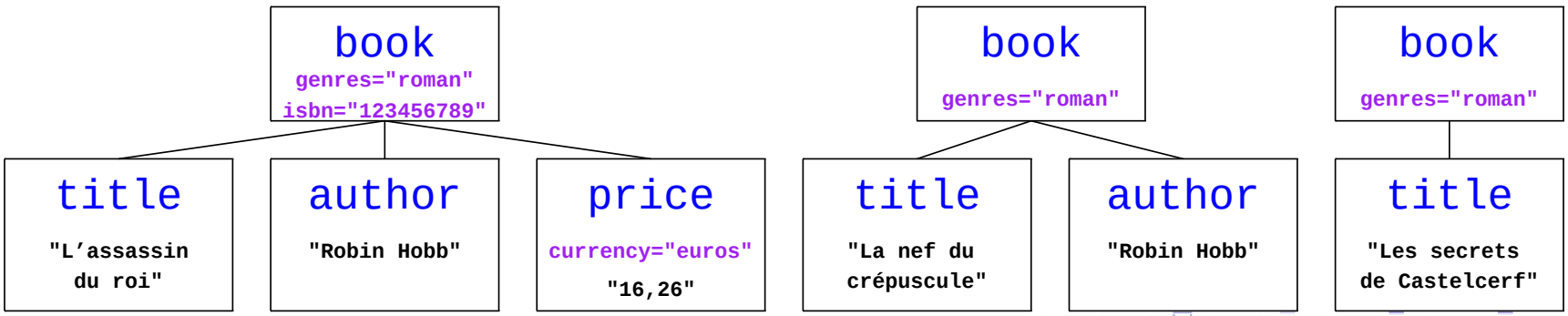
```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
          {$i/price/text()}
      </prix>
```



XQuery

```
<book genres="roman" isbn="123456789">
  <title> L'assassin du roi </title>
  <author>Robin Hobb </author>
  <price currency="euros"> 16,26 </price>
</book>
<book genres="roman">
  <title> La nef du crépuscule </title>
  <author>Robin Hobb </author>
</book>
<book genres="roman">
  <title> Les secrets de Castelcerf </title>
</book>
```

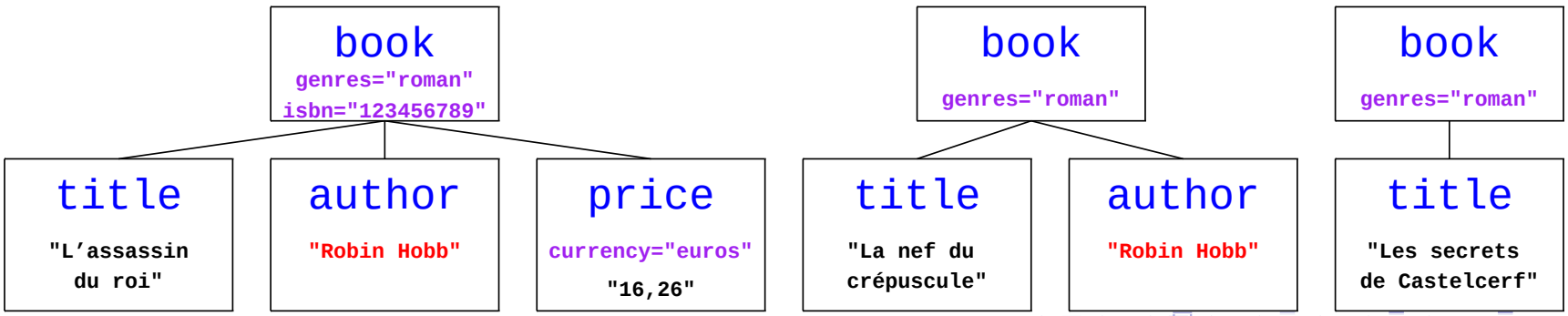
```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```



XQuery

```
<book genres="roman" isbn="123456789">
  <title> L'assassin du roi </title>
  <author>Robin Hobb </author>
  <price currency="euros"> 16,26 </price>
</book>
<book genres="roman">
  <title> La nef du crépuscule </title>
  <author>Robin Hobb </author>
</book>
<book genres="roman">
  <title> Les secrets de Castelcerf </title>
</book>
```

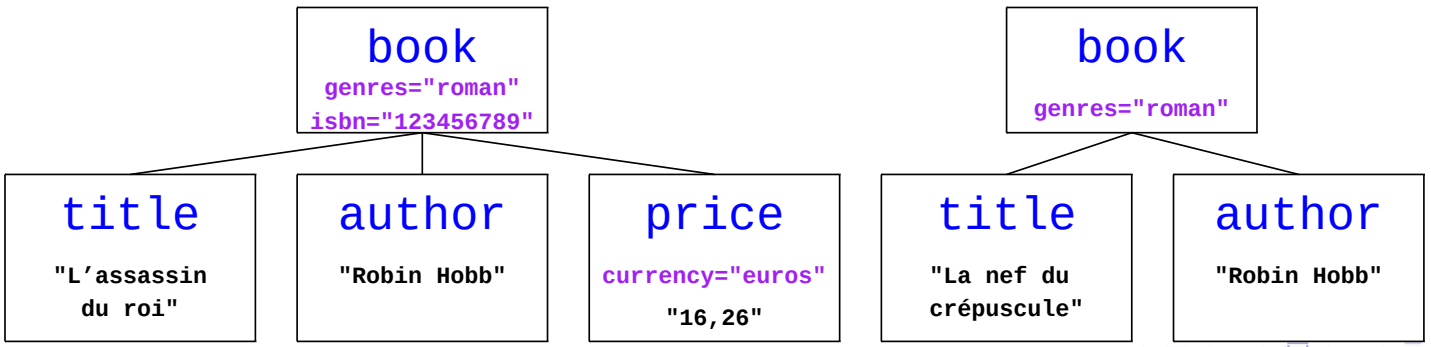
```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```



XQuery

```
<book genres="roman" isbn="123456789">
  <title> L'assassin du roi </title>
  <author>Robin Hobb </author>
  <price currency="euros"> 16,26 </price>
</book>
<book genres="roman">
  <title> La nef du crépuscule </title>
  <author>Robin Hobb </author>
</book>
```

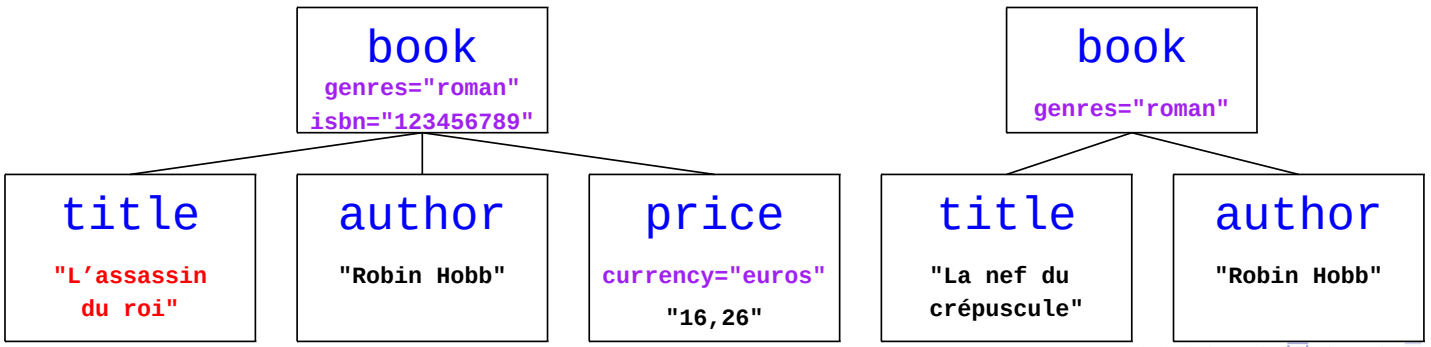
```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```



XQuery

```
<book genres="roman" isbn="123456789">
  <title> L'assassin du roi </title>
  <author>Robin Hobb </author>
  <price currency="euros"> 16,26 </price>
</book>
<book genres="roman">
  <title> La nef du crépuscule </title>
  <author>Robin Hobb </author>
</book>
```

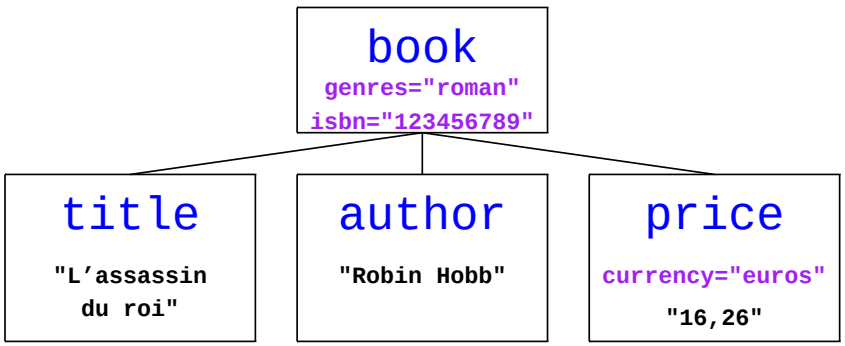
```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```



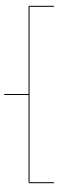
XQuery

```
<book genres="roman" isbn="123456789">
  <title> L'assassin du roi </title>
  <author>Robin Hobb </author>
  <price currency="euros"> 16,26 </price>
</book>
```

```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```



XQuery



16,26

```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```

price

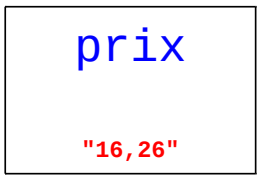
currency="euros"

"16,26"

XQuery

<prix> 16,26 </prix>

```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```



Motivations

```
declare function local :f($doc as xs :string) as element()
{
 for $x in (doc("rev.xml")/review|doc("$doc")/catalog)
       [. contains("Robin Hobb")]/book/[.//price > 15]
 where
       some $y in $x/comments
            satisfies contains ($y, "Excellent")
 order by $x/@isbn
 return
    <livre>
       {$x/@isbn}
       <prix>{$x//price/text()}</price>
       {
          if (count($x/title) > 2)
          then
          {
             for $z in doc("books.xml")/book
             where $z/@isbn = $x/@isbn
             return <titre>{($z/title)[3]}</titre>
          }
          else<titre/>
       }
    </livre>
}
```

- XPath ;
- Contraintes ;
- Filtres ;
- Quantificateurs ;
- Construction de document ;
- Imbrications ;
- Agrégats ;
- Opérations conditionnelles ;
- Opérations ensemblistes ;
- Ordonnancements ;
- Séquences ;
- Fonctions ;

Motivations

```
declare function local :f($doc as xs :string) as element()
{
 for $x in (doc("rev.xml")/review|doc("$doc")/catalog)
       [. contains("Robin Hobb")]/book/[.//price > 15]
 where
       some $y in $x/comments
            satisfies contains ($y, "Excellent")
 order by $x/@isbn
 return
    <livre>
       {$x/@isbn}
       <prix>{$x//price/text()}</price>
       {
          if (count($x/title) > 2)
          then
          {
             for $z in doc("books.xml")/book
             where $z/@isbn = $x/@isbn
             return <titre>{($z/title)[3]}</titre>
          }
          else<titre/>
       }
    </livre>
}
```

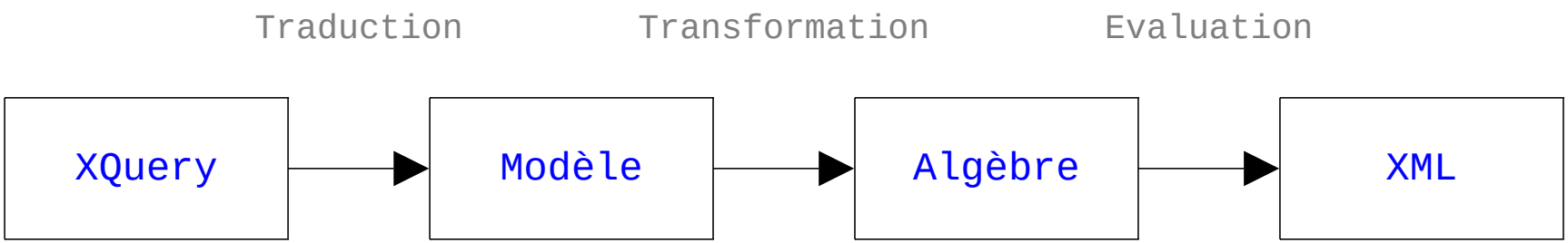
- XPath ;
- Contraintes ;
- Filtres ;
- Quantificateurs ;
- Construction de document ;
- Imbrications ;
- Agrégats ;
- Opérations conditionnelles ;
- Opérations ensemblistes ;
- Ordonnancements ;
- Séquences ;
- Fonctions ;

Besoin d'un modèle pour représenter les requêtes XQuery.

Problématique

Pour créer un modèle de représentation de requêtes, il faut :

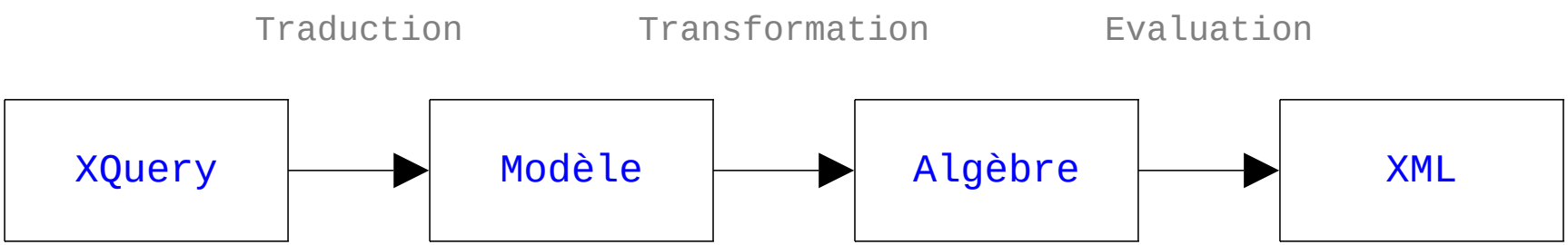
- Tenir compte des caractéristiques du langage ;
- Prendre en compte le contexte de médiation ;
- Définir une évaluation (Algèbre).



Problématique

Pour créer un modèle de représentation de requêtes, il faut :

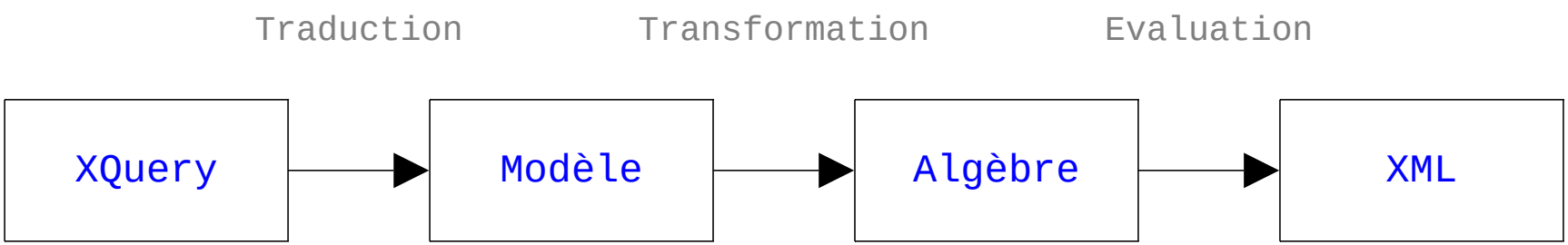
- Tenir compte des caractéristiques du langage ;
- Prendre en compte le contexte de médiation ;
- Définir une évaluation (Algèbre).



Problématique

Pour créer un modèle de représentation de requêtes, il faut :

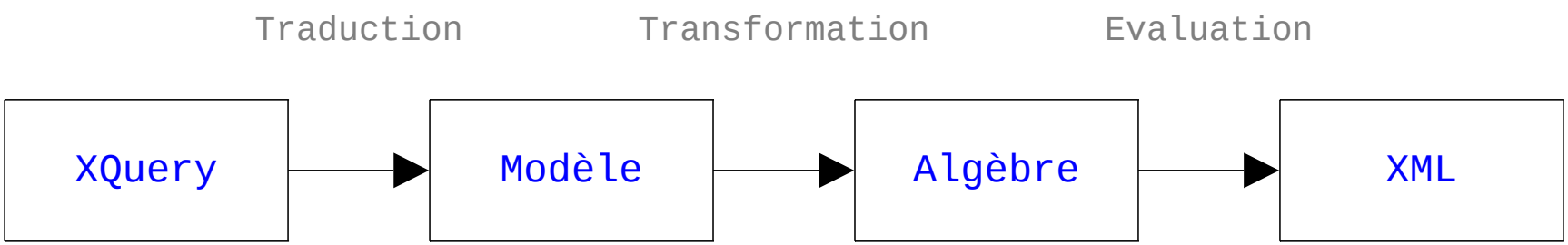
- Tenir compte des caractéristiques du langage ;
- Prendre en compte le contexte de médiation ;
- Définir une évaluation (Algèbre).



Problématique

Pour créer un modèle de représentation de requêtes, il faut :

- Tenir compte des caractéristiques du langage ;
- Prendre en compte le contexte de médiation ;
- Définir une évaluation (Algèbre).



Canonisation des requêtes

Caractéristiques de XQuery :

- Syntaxe très riche ;
- Formes équivalentes (spécifications) ;

```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```

Est équivalente à :

```
for $i in doc("cat.xml")/catalog/book[./title = "L'assassin du roi"]
where $i//author = "Robin Hobb"
return
      <prix>
            {$i/price/text()}

      </prix>
```

Canonisation des requêtes

Caractéristiques de XQuery :

- Syntaxe très riche ;
- Formes équivalentes (spécifications) ;

```
for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```

Est équivalente à :

```
for $i in doc("cat.xml")/catalog/book[./title = "L'assassin du roi"]
where $i//author = "Robin Hobb"
return
      <prix>
            {$i/price/text()}

      </prix>
```

Canonisation des requêtes

Caractéristiques de XQuery :

- Syntaxe très riche ;
- Formes équivalentes (spécifications) ;

```
 for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```

Est équivalente à :

```
for $i in doc("cat.xml")/catalog/book[./title = "L'assassin du roi"]
where $i//author = "Robin Hobb"
return
      <prix>
            {$i/price/text()}

      </prix>
```

Canonisation des requêtes

Caractéristiques de XQuery :

- Syntaxe très riche ;
- Formes équivalentes (spécifications) ;

```
 for $i in doc("cat.xml")/catalog/book
where $i//author = "Robin Hobb"
      and $i/title = "L'assassin du roi"
return
      <prix>
            {$i/price/text()}
      </prix>
```

Est équivalente à :

```
for $i in doc("cat.xml")/catalog/book[./title = "L'assassin du roi"]
where $i//author = "Robin Hobb"
return
      <prix>
            {$i/price/text()}

      </prix>
```

Besoin d'une canonisation pour transformer les requêtes en un format unique.

Règles de canonisation [Chen 04]

Règles de canonisation existantes :

- Filtres ;
- Imbrications ;
- Agrégats ;
- Quantificateurs.

Requête XQuery	Requête XQuery Canonique

Règles de canonisation [Chen 04]

Règles de canonisation existantes :

- Filtres ;
- Imbrications ;
- Agrégats ;
- Quantificateurs.

Requête XQuery	Requête XQuery Canonique
for $i in doc("cat.xml")/catalog/book [@isbn="12351234"]/title return {$i}	for $j in doc("cat.xml")/catalog/book **for $i in $j/title** where **$j/@isbn = "12351234"** return {$i}

Règles de canonisation [Chen 04]

Règles de canonisation existantes :

- Filtres ;
- Imbrications ;
- Agrégats ;
- Quantificateurs.

Requête XQuery	Requête XQuery Canonique
for $i in doc("cat.xml")/catalog/book return <livre> {for $j in $i/title return {$j}} </livre>	for $i in doc("cat.xml")/catalog/book **let $l :=** (for $j in $i/title return {$j}) return <livre>{**$l**}</livre>

Règles de canonisation [Chen 04]

Règles de canonisation existantes :

- Filtres ;
- Imbrications ;
- Agrégats ;
- Quantificateurs.

Requête XQuery	Requête XQuery Canonique
for $i in collection("catalog")/catalog/book return <count>{count($i/author)}</count>	for $i in collection("catalog")/catalog/book **let $l :=** count($i/author) return <count>{**$l**}</count>

Règles de canonisation [Chen 04]

Règles de canonisation existantes :

- Filtres ;
- Imbrications ;
- Agrégats ;
- Quantificateurs.

Requête XQuery	Requête XQuery Canonique
for $i in doc("cat.xml")/catalog/book where every $s in $i/price satisfies $s > 15 return {$i}	for $i in doc("cat.xml")/catalog/book **let $l :=** (for $j in $i/price where **$j <= 15** return {$j}) where **count($l) = 0** return {$i}

XQuery canonique non-typée

XQuery canonique [Chen 04] :

- XPath ;
- Contraintes ;
- Filtres ;
- Quantificateurs ;
- Construction de document ;
- Imbrications ;
- Agrégats ;

Manque de règles pour :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

XQuery canonique non-typée

XQuery canonique [Chen 04] :

- XPath ;
- Contraintes ;
- Filtres ;
- Quantificateurs ;
- Construction de document ;
- Imbrications ;
- Agrégats ;

Manque de règles pour :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

Règles de canonisation

Nouvelles règles de canonisation :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

Requête XQuery	Requête XQuery Canonique

Règles de canonisation

Nouvelles règles de canonisation :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

Requête XQuery	**Requête XQuery Canonique**
for $i in /catalog/book order by $i/title return $i/title	for $i in /catalog/book let $j := **orderby ($i, $i/title)** **for $k in $j** return $k/title

Règles de canonisation

Nouvelles règles de canonisation :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

Requête XQuery	**Requête XQuery Canonique**
for $i in (/catalog | /review)/book return $i/title	let i_3$:= **for $**$i_1$ **in /catalog** **for $**$i_2$ **in /review** **return ($**$i_1$ | **$**$i_2$**)** for $i in i_3$/book return $i/title

Règles de canonisation

Nouvelles règles de canonisation :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

Requête XQuery	**Requête XQuery Canonique**
for $i in /catalog/book return {if contains ($i/author, "Hobb") then (for $j in $i//title return $j) else ($i/author)}	for $i in /catalog/book let $l := **for $j in $i//title return $j** return {if contains ($i/author, "Hobb") then ($l) else ($i/author)}

Règles de canonisation

Nouvelles règles de canonisation :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

Requête XQuery	**Requête XQuery Canonique**
for $i in (/catalog/book)[2] return $i/title	let i_1$:= **for $x in /catalog/book** **return $x** for $i in i_1$ where **$i/position() == 2** return $i/title

Règles de canonisation

Nouvelles règles de canonisation :

- Ordonnancements ;
- Opérations ensemblistes ;
- Opérations conditionnelles ;
- Séquences ;
- Fonctions ;

Requête XQuery	Requête XQuery Canonique

declare function local :section
($i as element()) as element ()*
{
for $j in $i/book
return
<book>
{$j/title}
{for $s in $i/section/title
return <section>{$s/text()}</section>}
</book>
}
for $f in doc("catalog.xml")/catalog
return local:section($f)
	declare function local :section
	($i as element()) as element ()*
	{
	for $j in $i/book
	let $l := (for $s in $i/section/title
	return <section>
	{$s/text()}
	</section>)
	return
	<book> {$j/title} {$l} </book>
	}
	for $f in doc("catalog.xml")/catalog
	return local:section($f)

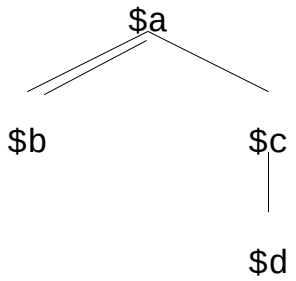
XQuery Canonique non typée

Grâce à ces règles de canonisation :

- Tout XQuery non-typé ;
- Forme canonique utile pour la modélisation.

Motifs d'arbre

TPQ (ou Tree Pattern Queries) [Amer-Yahia et al. 01]



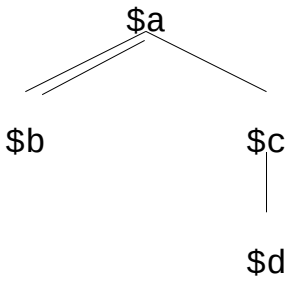
$a.tag = catalog &
$b.tag = author &
$b.content = "Robin Hobb" &
$c.tag = book &
$d.tag = title &
$d.content = "L'assassin du roi"

Modélise :

- arborescente (XPath) ;
- contraintes.

Motifs d'arbre

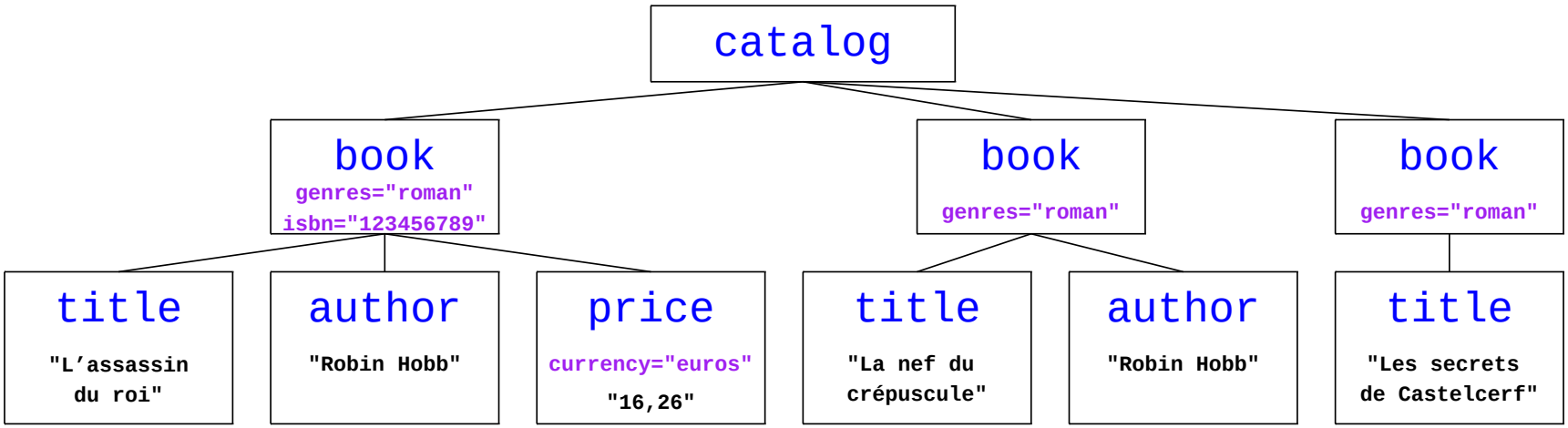
TPQ (ou Tree Pattern Queries) [Amer-Yahia et al. 01]



$a.tag = catalog &
$b.tag = author &
$b.content = "Robin Hobb" &
$c.tag = book &
$d.tag = title &
$d.content = "L'assassin du roi"

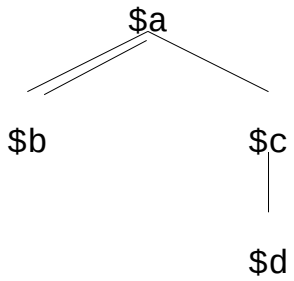
Modélise :

- arborescente (XPath) ;
- contraintes.



Motifs d'arbre

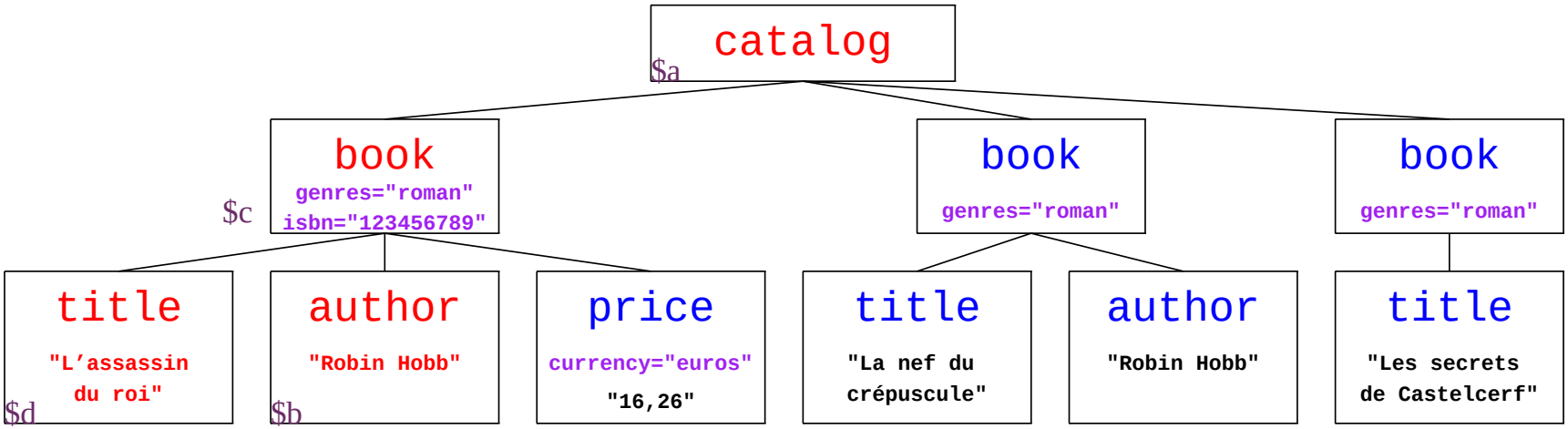
TPQ (ou Tree Pattern Queries) [Amer-Yahia et al. 01]



$a.tag = catalog &
$b.tag = author &
$b.content = "Robin Hobb" &
$c.tag = book &
$d.tag = title &
$d.content = "L'assassin du roi"

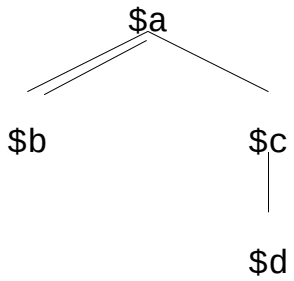
Modélise :

- arborescente (XPath) ;
- contraintes.



Motifs d'arbre

TPQ (ou Tree Pattern Queries) [Amer-Yahia et al. 01]

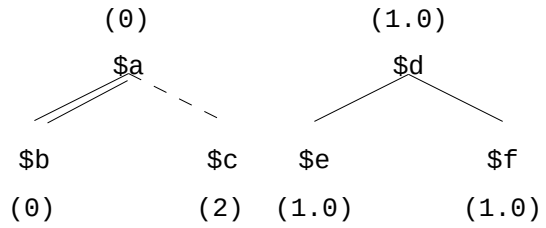


$a.tag = catalog &
$b.tag = author &
$b.content = "Robin Hobb" &
$c.tag = book &
$d.tag = title &
$d.content = "L'assassin du roi"

Modélise :

- arborescente (XPath) ;
- contraintes.

GTP (ou Generalized Tree Pattern) [Chen et al. 03]



$a.tag = catalog & $b.tag = author &
$e.tag = title & $c.tag = book &
$d.tag = book & $e.tag = author &
$f.tag = price &
$e.content = "L'assassin du roi" &
$b.content = "Robin Hobb" &
$f.content > 14 & $c.id = $d.id

Ajoute au modèle :

- Canonisation [Chen 04] ;
- Optionalité (return) ;
- Jointure ;
- Imbrication.

Problèmes

GTP inadapté avec la problématique :

- Modèle incomplet :
  - Documents ciblés ;
  - Clause let ;
  - Reconstruction de résultat ;
  - Agrégats ;
- XQuery incomplet :
  - Ordonnancements ;
  - Opérations ensemblistes ;
  - Opérations conditionnelles ;
  - Séquences ;
  - Fonctions ;
- Ne permet pas la médiation de requêtes :
  - Identification des sources de données ;
  - Localisation des sous-requêtes ;
- Ne facilite pas l'optimisation :
  - Formule avec un seul prédicat ;
  - Support d'information absent.

Problèmes

GTP inadapté avec la problématique :

- Modèle incomplet :
  - Documents ciblés ;
  - Clause let ;
  - Reconstruction de résultat ;
  - Agrégats ;
- XQuery incomplet :
  - Ordonnancements ;
  - Opérations ensemblistes ;
  - Opérations conditionnelles ;
  - Séquences ;
  - Fonctions ;
- Ne permet pas la médiation de requêtes :
  - Identification des sources de données ;
  - Localisation des sous-requêtes ;
- Ne facilite pas l'optimisation :
  - Formule avec un seul prédicat ;
  - Support d'information absent.

Problèmes

GTP inadapté avec la problématique :

- Modèle incomplet :
  - Documents ciblés ;
  - Clause let ;
  - Reconstruction de résultat ;
  - Agrégats ;
- XQuery incomplet :
  - Ordonnancements ;
  - Opérations ensemblistes ;
  - Opérations conditionnelles ;
  - Séquences ;
  - Fonctions ;
- Ne permet pas la médiation de requêtes :
  - Identification des sources de données ;
  - Localisation des sous-requêtes ;
- Ne facilite pas l'optimisation :
  - Formule avec un seul prédicat ;
  - Support d'information absent.

Problèmes

GTP inadapté avec la problématique :

- Modèle incomplet :
  - Documents ciblés ;
  - Clause let ;
  - Reconstruction de résultat ;
  - Agrégats ;
- XQuery incomplet :
  - Ordonnancements ;
  - Opérations ensemblistes ;
  - Opérations conditionnelles ;
  - Séquences ;
  - Fonctions ;
- Ne permet pas la médiation de requêtes :
  - Identification des sources de données ;
  - Localisation des sous-requêtes ;
- Ne facilite pas l'optimisation :
  - Formule avec un seul prédicat ;
  - Support d'information absent.

Problèmes

GTP inadapté avec la problématique :

- Modèle incomplet :
  - Documents ciblés ;
  - Clause let ;
  - Reconstruction de résultat ;
  - Agrégats ;
- XQuery incomplet :
  - Ordonnancements ;
  - Opérations ensemblistes ;
  - Opérations conditionnelles ;
  - Séquences ;
  - Fonctions ;
- Ne permet pas la médiation de requêtes :
  - Identification des sources de données ;
  - Localisation des sous-requêtes ;
- Ne facilite pas l'optimisation :
  - Formule avec un seul prédicat ;
  - Support d'information absent.

Problèmes

GTP inadapté avec la problématique :

- Modèle incomplet :
  - Documents ciblés ;
  - Clause let ;
  - Reconstruction de résultat ;
  - Agrégats ;
- XQuery incomplet :
  - Ordonnancements ;
  - Opérations ensemblistes ;
  - Opérations conditionnelles ;
  - Séquences ;
  - Fonctions ;
- Ne permet pas la médiation de requêtes :
  - Identification des sources de données ;
  - Localisation des sous-requêtes ;
- Ne facilite pas l'optimisation :
  - Formule avec un seul prédicat ;
  - Support d'information absent.

Besoin d'un modèle plus riche : TGV

Tree Graph View (TGV)

Nous proposons un modèle à base de motifs d'arbre :

- Supporte tout XQuery non typé ;
- Représentation intuitive et visuelle ;
- Support pour l'optimisation.

Modèle formalisé avec Type Abstraits de Données :

- Définition rigoureuse TGV ;
- Implémentation.

Tree Graph View (TGV)

Nous proposons un modèle à base de motifs d'arbre :

- Supporte tout XQuery non typé ;
- Représentation intuitive et visuelle ;
- Support pour l'optimisation.

Modèle formalisé avec Type Abstraits de Données :

- Définition rigoureuse TGV ;
- Implémentation.

Tree Graph View (TGV)

Nous proposons un modèle à base de motifs d'arbre :

- Supporte tout XQuery non typé ;
- Représentation intuitive et visuelle ;
- Support pour l'optimisation.

Modèle formalisé avec Type Abstraits de Données :

- Définition rigoureuse TGV ;
- Implémentation.

Tree Graph View (TGV)

Nous proposons un modèle à base de motifs d'arbre :

- Supporte tout XQuery non typé ;
- Représentation intuitive et visuelle ;
- Support pour l'optimisation.

Modèle formalisé avec Type Abstraits de Données :

- Définition rigoureuse TGV ;
- Implémentation.

Tree Graph View (TGV)

Nous proposons un modèle à base de motifs d'arbre :

- Supporte tout XQuery non typé ;
- Représentation intuitive et visuelle ;
- Support pour l'optimisation.

Modèle formalisé avec Type Abstraits de Données :

- Définition rigoureuse TGV ;
- Implémentation.

Définition du modèle

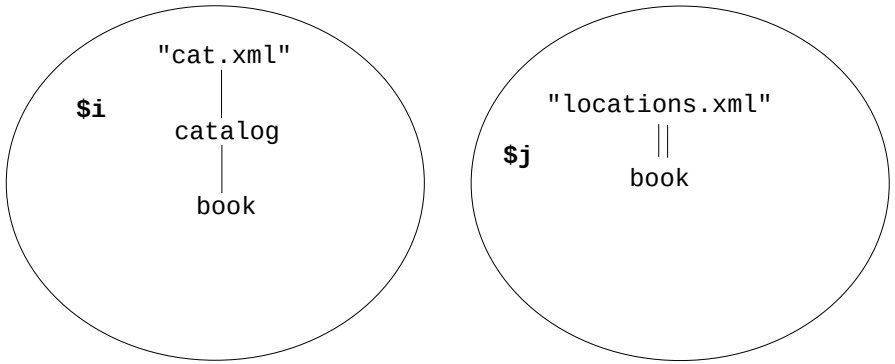
- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```

Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

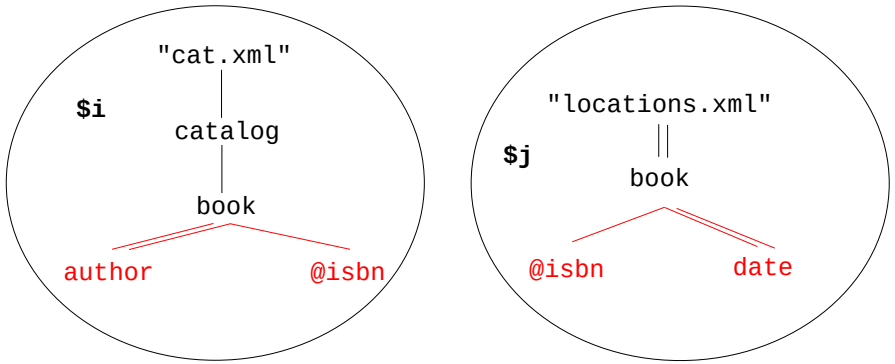
```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```



Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

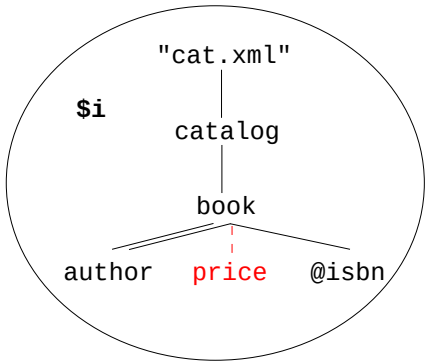
```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```

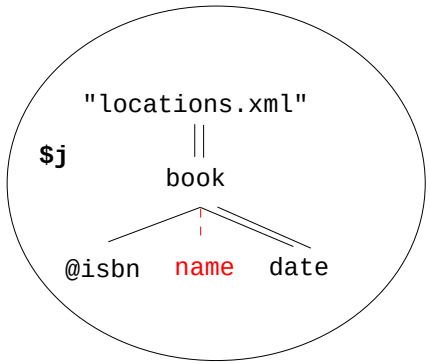


Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```

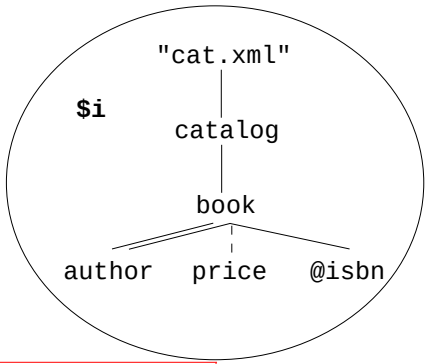


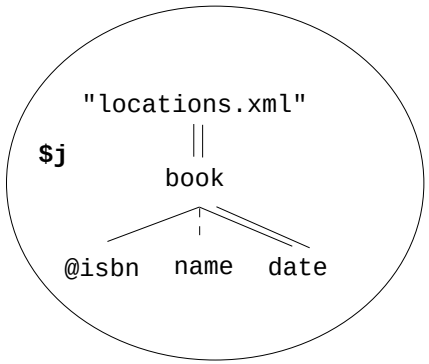


Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```

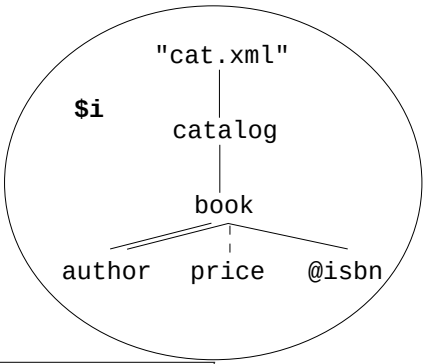


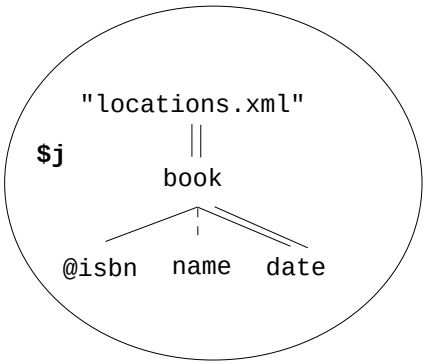


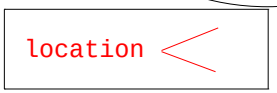
Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```



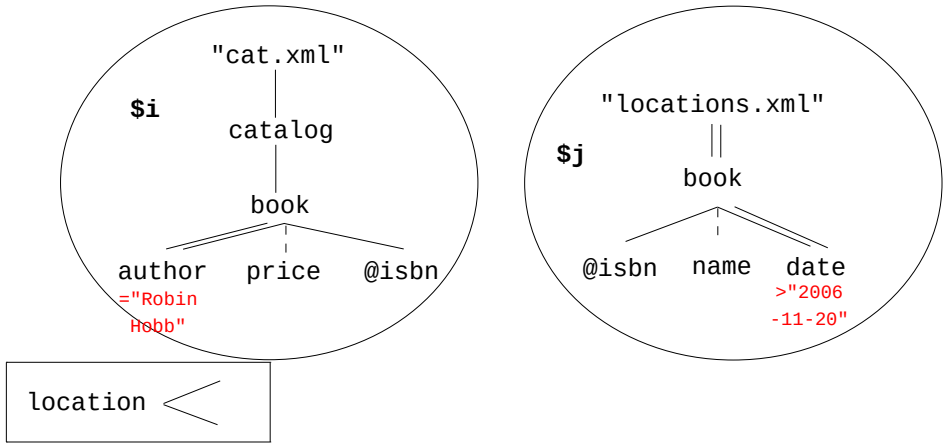




Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

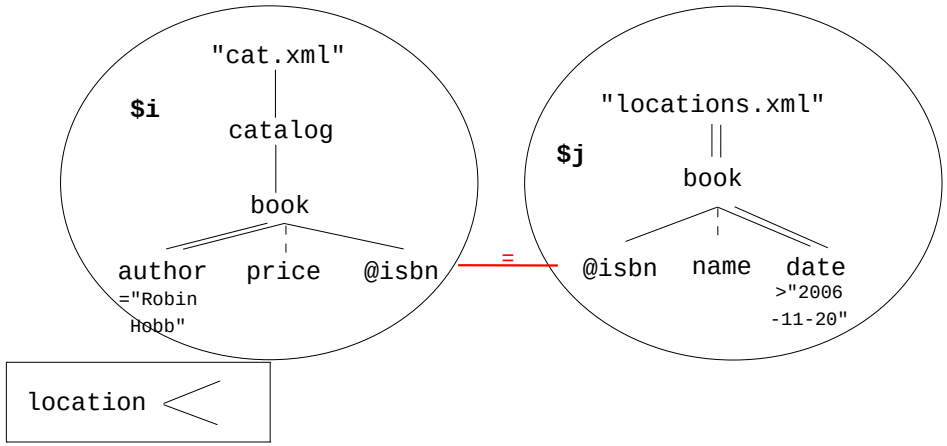
```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```



Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

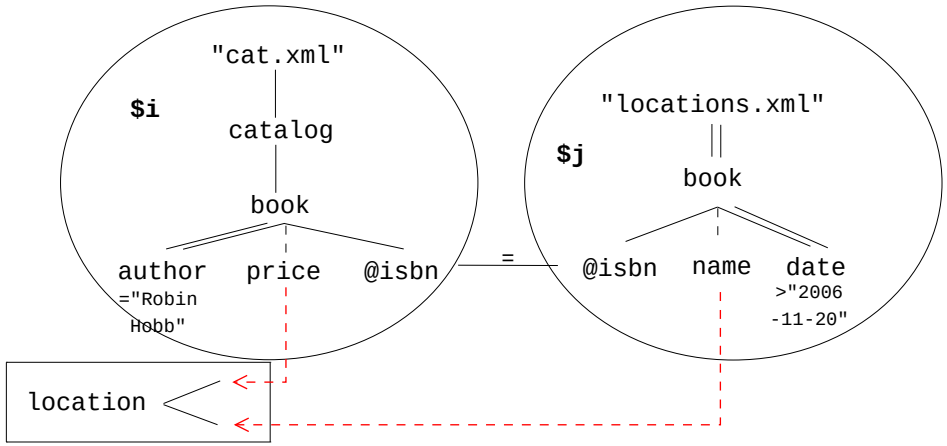
```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```



Définition du modèle

- Motif d'arbre :
  - Source ;
  - Résultat ;
- Contraintes ;
- Hyperliens ;
  - Jointure ;
  - Projection ;

```
for $i in doc("cat.xml")/catalog/book
for $j in doc("locations.xml")//book
where $i//author = "Robin Hobb"
      and $j//date > "2006-11-20"
      and $i/@isbn = $j/@isbn
return
      <location>
            {$i/price}
            {$j/name}
      </location>
```

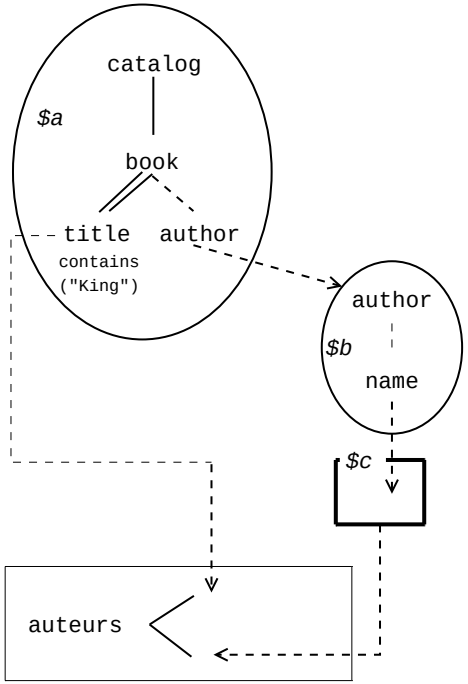


Exemples de TGV

- Imbrication ;
- Opération Ensembliste ;
- Opération Conditionnelle ;

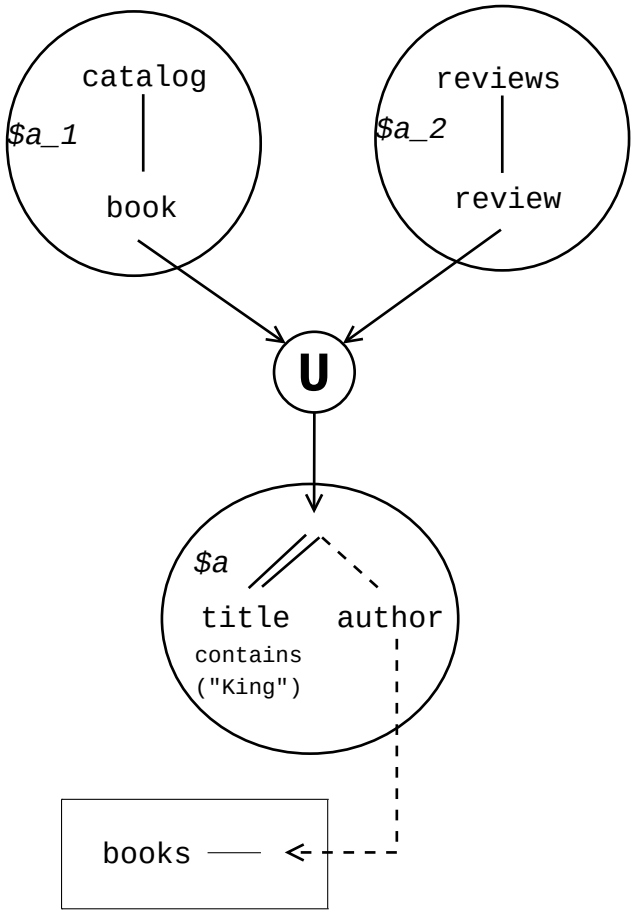
Exemples de TGV

- Imbrication ;
- Opération Ensembliste ;
- Opération Conditionnelle ;



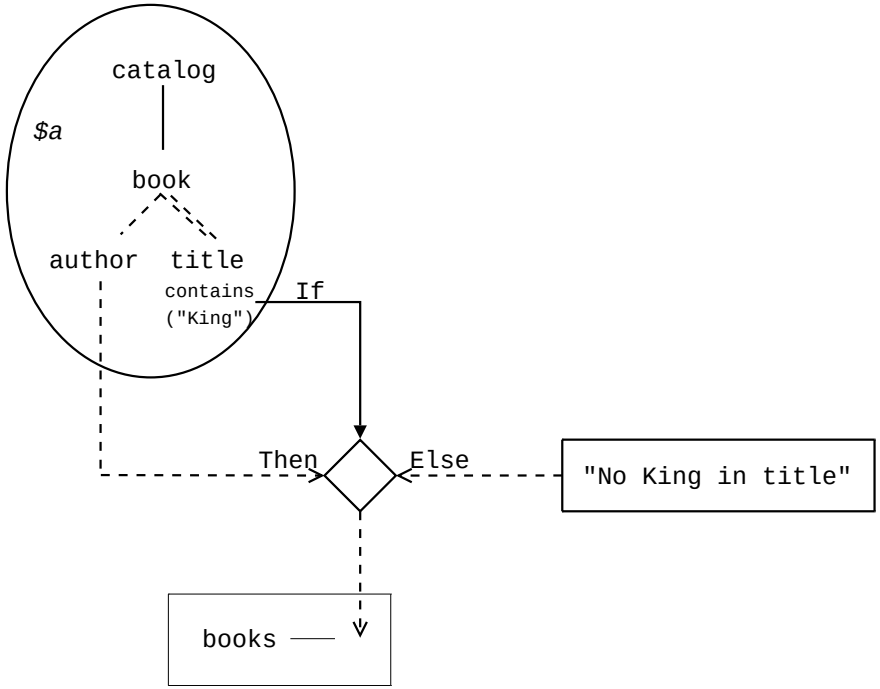
Exemples de TGV

- Imbrication ;
- Opération Ensembliste ;
- Opération Conditionnelle ;



Exemples de TGV

- Imbrication ;
- Opération Ensembliste ;
- Opération Conditionnelle ;



Annotations des TGV

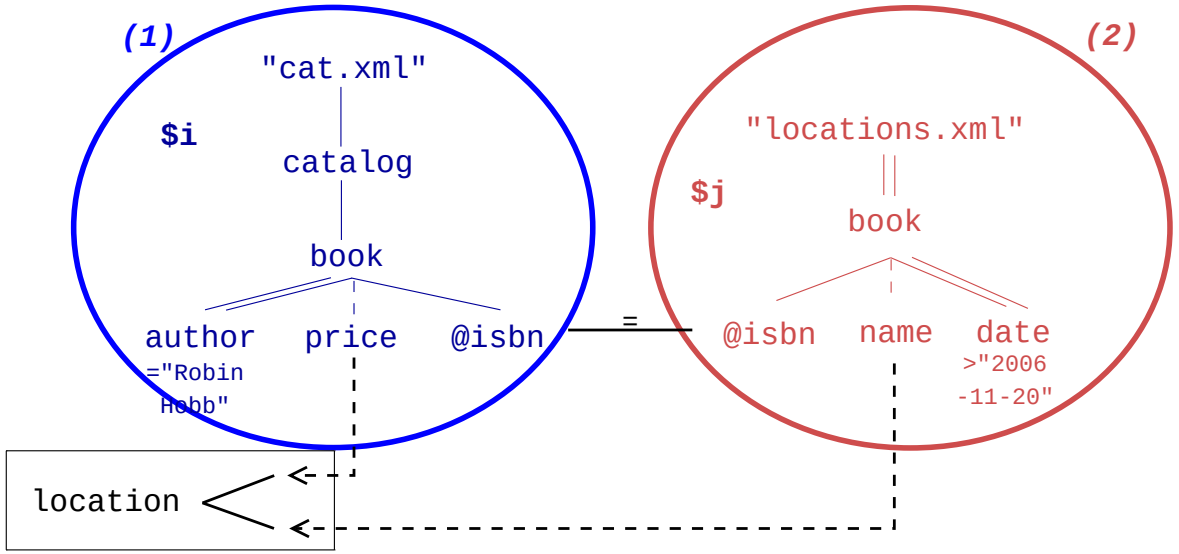
Base d'annotation :

- Localisation.
- Algorithmes ;
- Modèle de coût ;

Annotations des TGV

Base d'annotation :

- Localisation.
- Algorithmes ;
- Modèle de coût ;



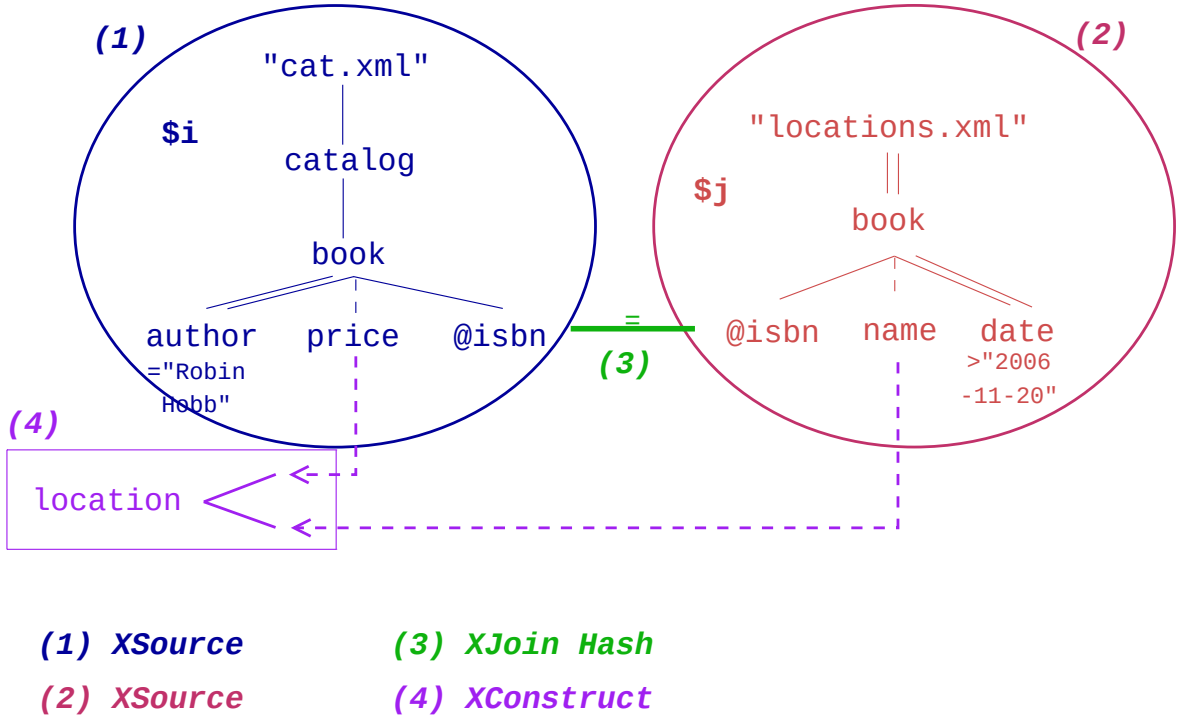
(1) source 1

(2) source 2

Annotations des TGV

Base d'annotation :

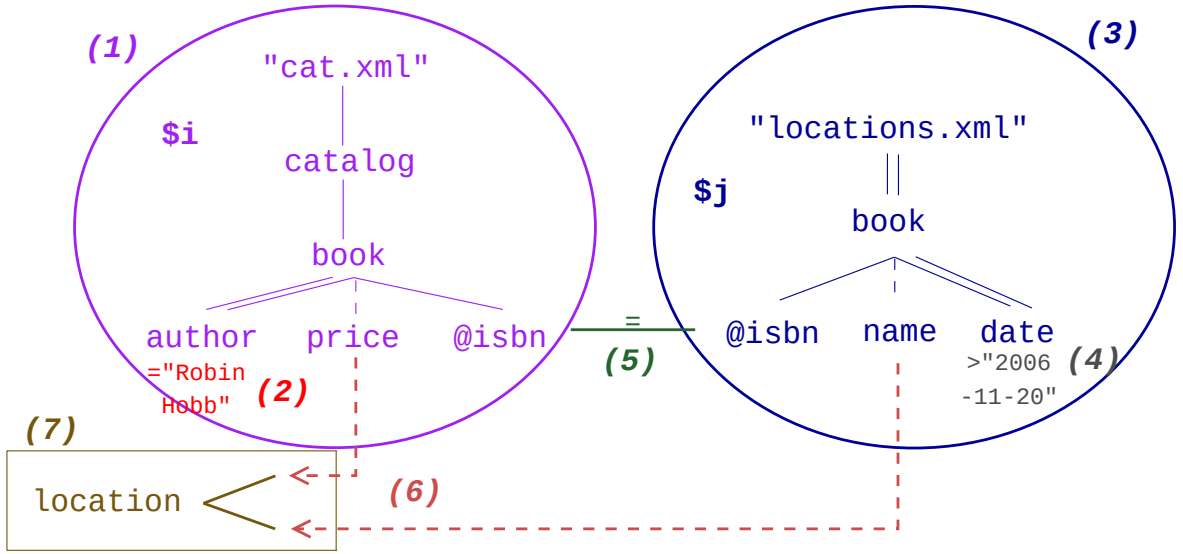
- Localisation.
- Algorithmes ;
- Modèle de coût ;



Annotations des TGV

Base d'annotation :

- Localisation.
- Algorithmes ;
- Modèle de coût ;



(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)

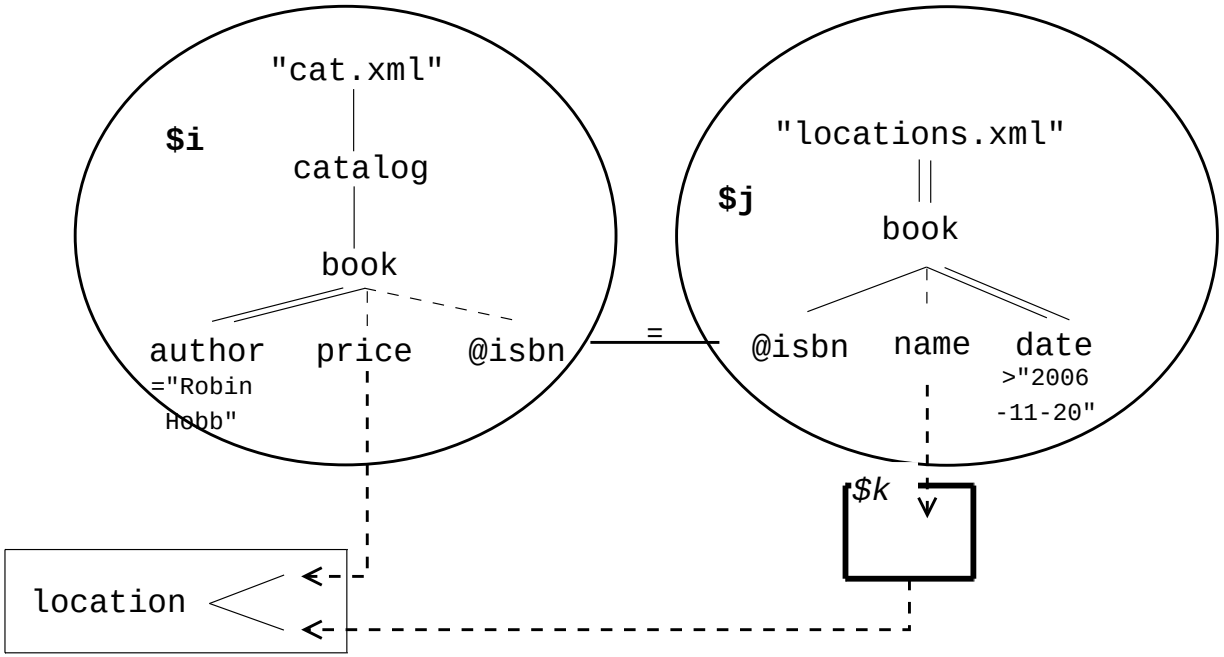
(5) cost_5 = CPU * card ((1) x (2))

(6) cost_6 = max (cost_2, cost_4)

(7) cost_7 = const + cost_5 + cost_6

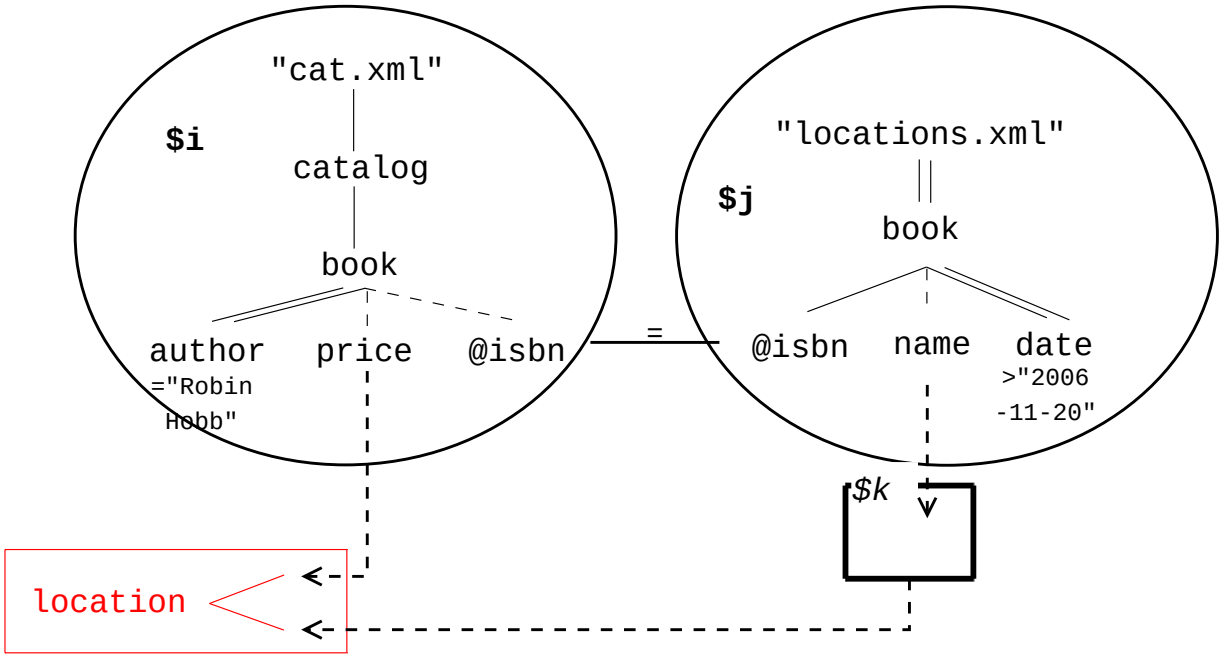
Évaluation des TGV

β construction π projection σ contraintes α agrégation ϕ filtres



Évaluation des TGV

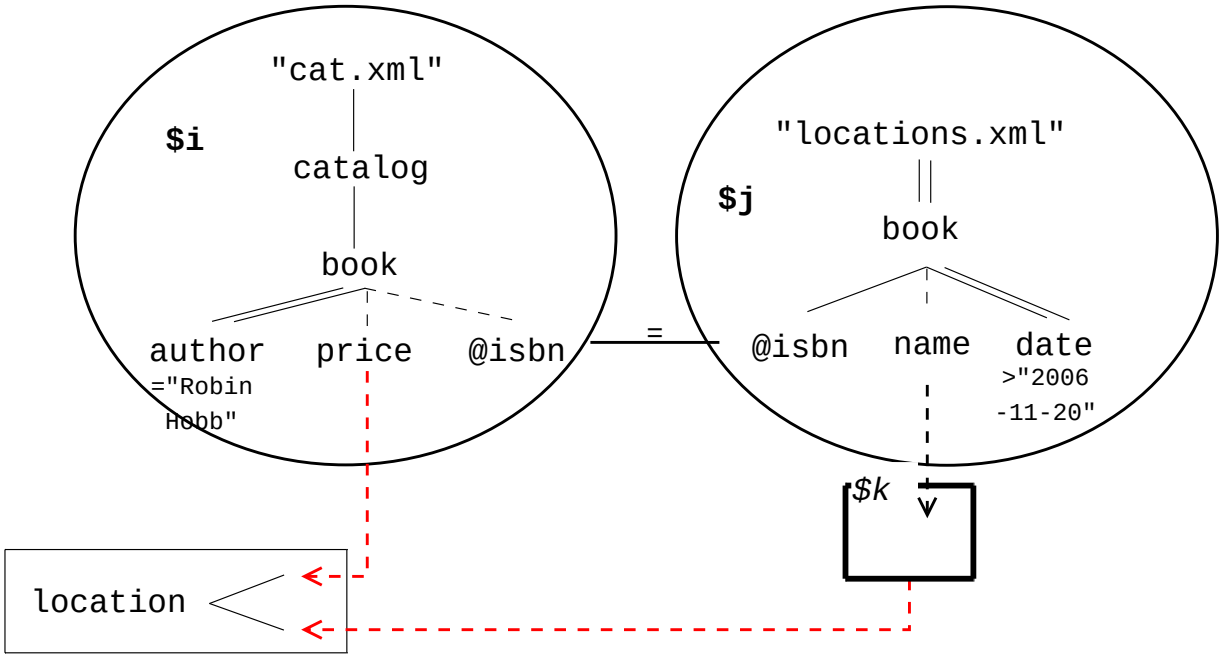
β construction — π projection — σ contraintes — α agrégation — ϕ filtres

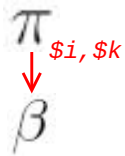


β

Évaluation des TGV

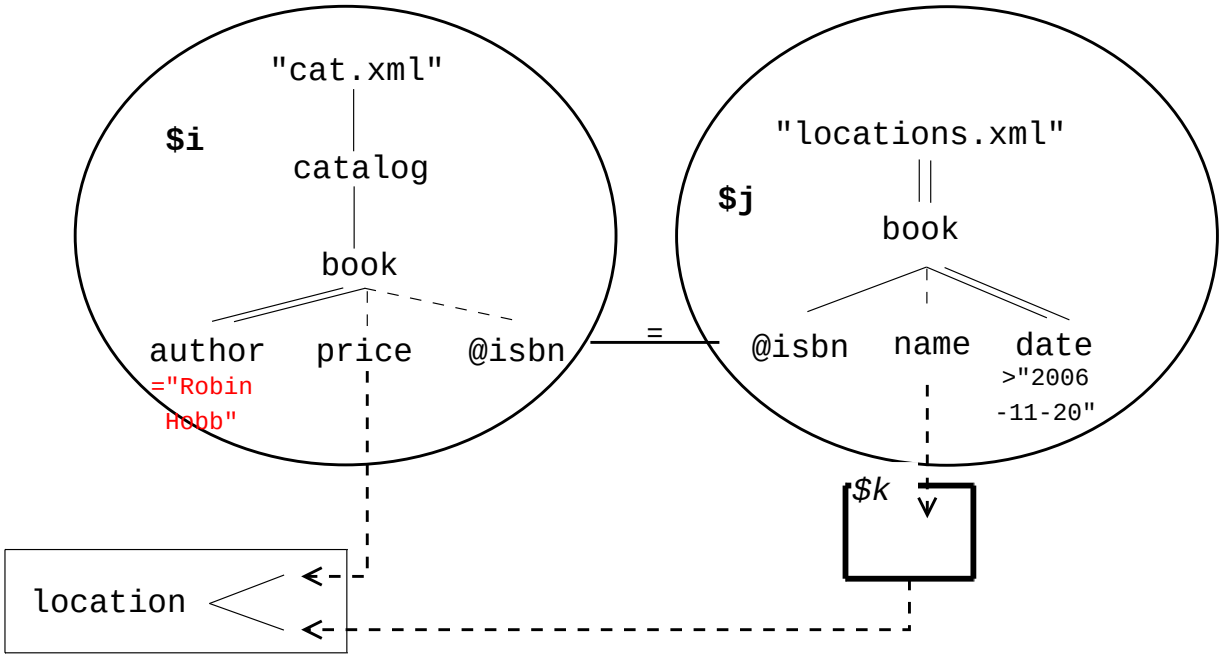
β construction — π projection — σ contraintes — α agrégation — ϕ filtres

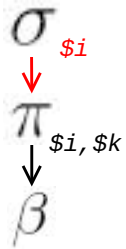




Évaluation des TGV

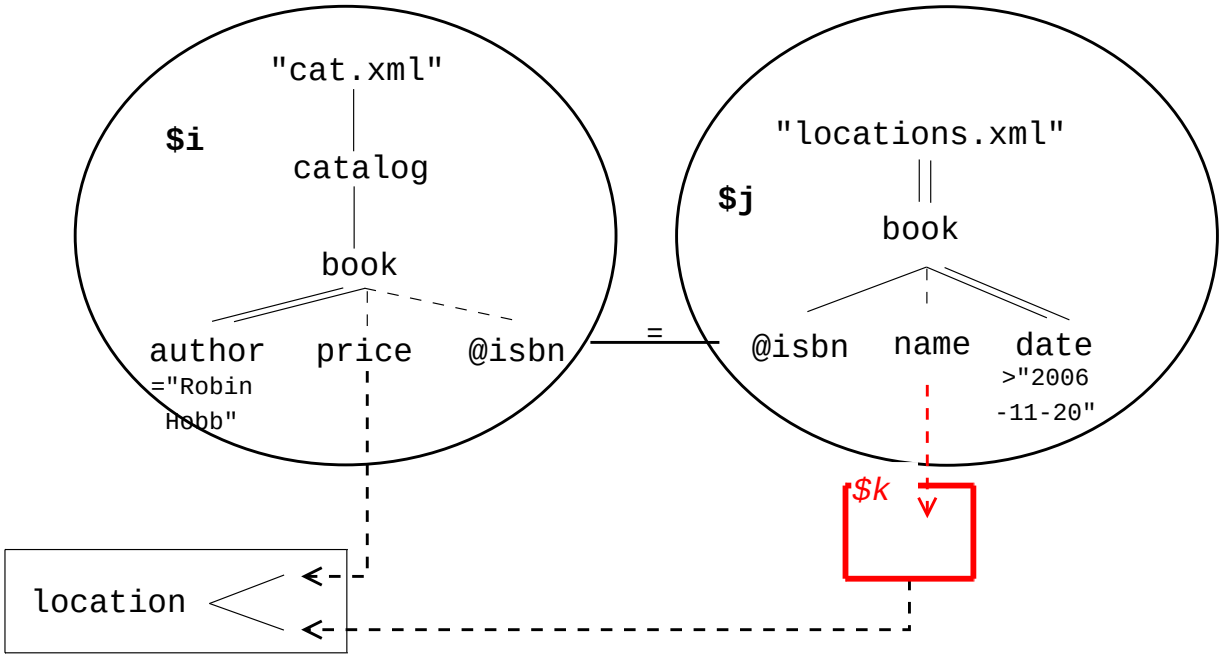
β construction π projection σ contraintes α agrégation ϕ filtres

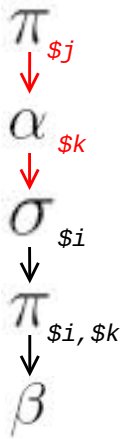




Évaluation des TGV

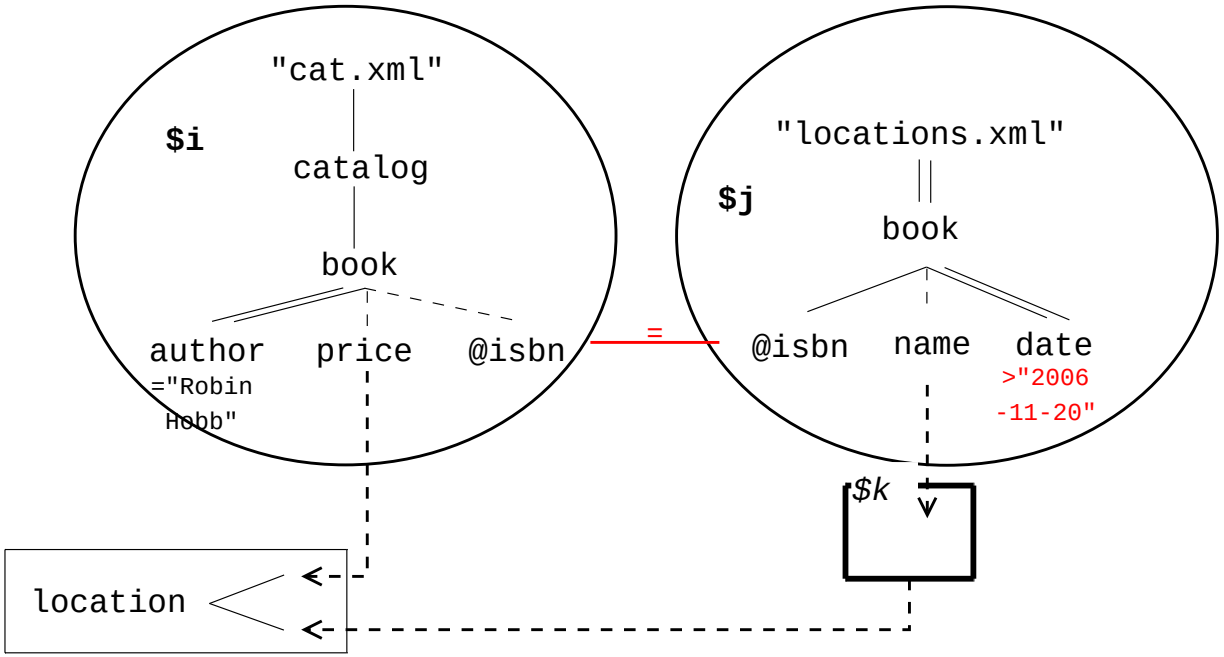
β construction | π projection | σ contraintes | α agrégation | ϕ filtres

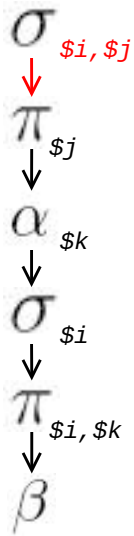




Évaluation des TGV

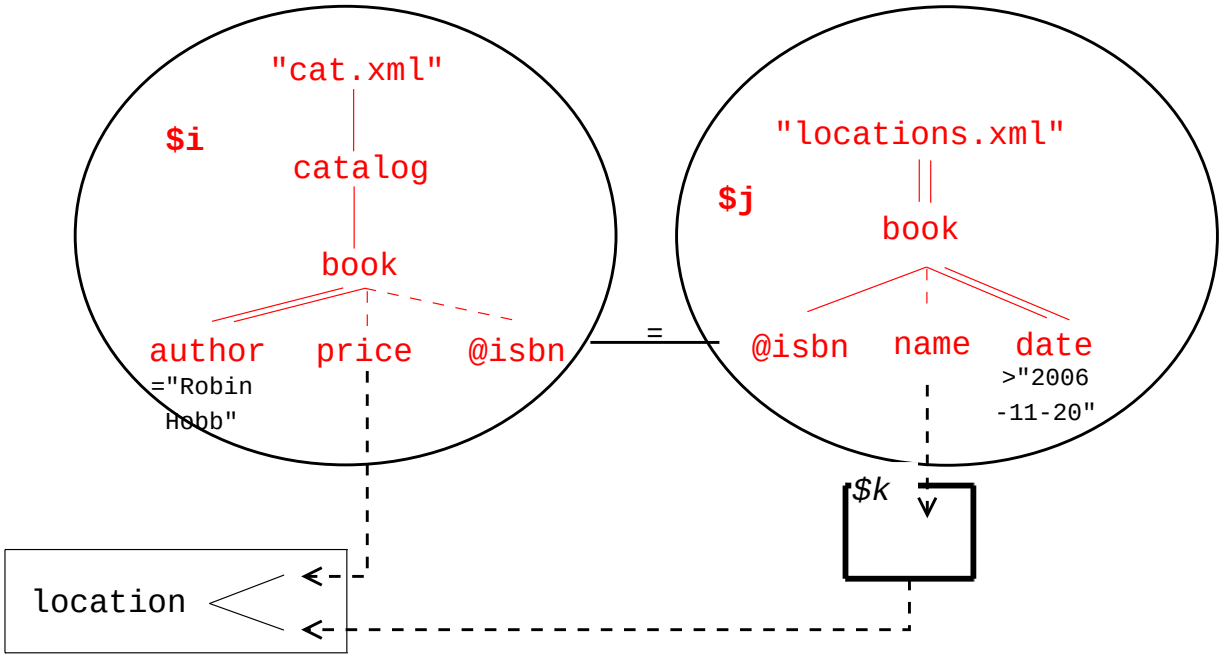
β construction π projection σ contraintes α agrégation ϕ filtres

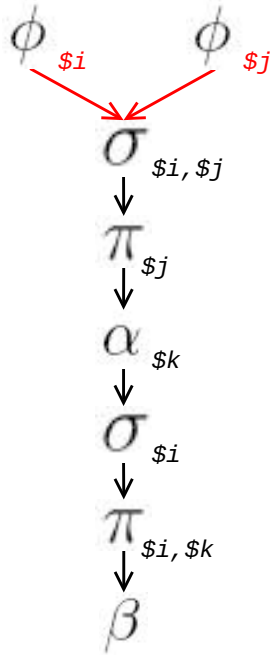




Évaluation des TGV

β construction π projection σ contraintes α agrégation ϕ filtres





Bilan TGV

TGV :

- Modèle de représentation graphique et intuitif ;
- Permet de faciliter la médiation ;
- Base d'évaluation ;
- Base d'annotation.

Conception d'un optimiseur

Stratégie de recherche du meilleur plan :

- Représentation équivalente pour un contexte donné ;
- Type de coût différents (temps d'exécution, communication, prix) ;

Besoin d'un optimiseur permettant :

- Règles de transformations sur le modèle ;
- Tenir compte du contexte de médiation.

Conception d'un optimiseur

Stratégie de recherche du meilleur plan :

- Représentation équivalente pour un contexte donné ;
- Type de coût différents (temps d'exécution, communication, prix) ;

Besoin d'un optimiseur permettant :

- Règles de transformations sur le modèle ;
- Tenir compte du contexte de médiation.

Conception d'un optimiseur

Stratégie de recherche du meilleur plan :

- Représentation équivalente pour un contexte donné ;
- Type de coût différents (temps d'exécution, communication, prix) ;

Besoin d'un optimiseur permettant :

- Règles de transformations sur le modèle ;
- Tenir compte du contexte de médiation.

Conception d'un optimiseur

Stratégie de recherche du meilleur plan :

- Représentation équivalente pour un contexte donné ;
- Type de coût différents (temps d'exécution, communication, prix) ;

Besoin d'un optimiseur permettant :

- Règles de transformations sur le modèle ;
- Tenir compte du contexte de médiation.

Conception d'un optimiseur

Stratégie de recherche du meilleur plan :

- Représentation équivalente pour un contexte donné ;
- Type de coût différents (temps d'exécution, communication, prix) ;

Besoin d'un optimiseur permettant :

- Règles de transformations sur le modèle ;
- Tenir compte du contexte de médiation.

Optimiseur extensible

- Un optimiseur *classique* utilise toujours les mêmes règles de transformations ;
- Un optimiseur *extensible* permet l'ajout de nouvelles règles de transformations :
  - Augmente l'espace de recherche ;
  - Amélioration de l'optimiseur ;
  - Adaptation du système ;
- Un optimiseur extensible a besoin de :
  - Règles de transformation (ajout de transformations) ;
  - Stratégie de recherche (orienter l'espace de recherche) ;
  - Modèle de coût (estimer le coût d'évaluation) ;

Optimiseur extensible

- Un optimiseur *classique* utilise toujours les mêmes règles de transformations ;
- Un optimiseur *extensible* permet l'ajout de nouvelles règles de transformations :
  - Augmente l'espace de recherche ;
  - Amélioration de l'optimiseur ;
  - Adaptation du système ;
- Un optimiseur extensible a besoin de :
  - Règles de transformation (ajout de transformations) ;
  - Stratégie de recherche (orienter l'espace de recherche) ;
  - Modèle de coût (estimer le coût d'évaluation) ;

Optimiseur extensible

- Un optimiseur *classique* utilise toujours les mêmes règles de transformations ;
- Un optimiseur *extensible* permet l'ajout de nouvelles règles de transformations :
  - Augmente l'espace de recherche ;
  - Amélioration de l'optimiseur ;
  - Adaptation du système ;
- Un optimiseur extensible a besoin de :
  - Règles de transformation (ajout de transformations) ;
  - Stratégie de recherche (orienter l'espace de recherche) ;
  - Modèle de coût (estimer le coût d'évaluation) ;

Optimiseur extensible

- Un optimiseur *classique* utilise toujours les mêmes règles de transformations ;
- Un optimiseur *extensible* permet l'ajout de nouvelles règles de transformations :
  - Augmente l'espace de recherche ;
  - Amélioration de l'optimiseur ;
  - Adaptation du système ;
- Un optimiseur extensible a besoin de :
  - Règles de transformation (ajout de transformations) ;
  - Stratégie de recherche (orienter l'espace de recherche) ;
  - Modèle de coût (estimer le coût d'évaluation) ;

Optimiseur extensible

- Un optimiseur *classique* utilise toujours les mêmes règles de transformations ;
- Un optimiseur *extensible* permet l'ajout de nouvelles règles de transformations :
  - Augmente l'espace de recherche ;
  - Amélioration de l'optimiseur ;
  - Adaptation du système ;
- Un optimiseur extensible a besoin de :
  - Règles de transformation (ajout de transformations) ;
  - Stratégie de recherche (orienter l'espace de recherche) ;
  - Modèle de coût (estimer le coût d'évaluation) ;

Optimiseur extensible

- Un optimiseur *classique* utilise toujours les mêmes règles de transformations ;
- Un optimiseur *extensible* permet l'ajout de nouvelles règles de transformations :
  - Augmente l'espace de recherche ;
  - Amélioration de l'optimiseur ;
  - Adaptation du système ;
- Un optimiseur extensible a besoin de :
  - Règles de transformation (ajout de transformations) ;
  - Stratégie de recherche (orienter l'espace de recherche) ;
  - Modèle de coût (estimer le coût d'évaluation) ;

Architectures d'optimisation extensible

- Exodus [Carey et al. 90] :
  - Stratégie de recherche (coefficient d'amélioration) ;
- Volcano [Graefe et McKenna 93] :
  - Définition de stratégies de recherche ;
- EPOQ [Mitchell et al. 93] :
  - Objets ;
  - Stratégie de recherche (région d'optimisation) ;
- OPT++ [Kabra et DeWitt 99] :
  - Objets ;
  - Définition de stratégies de recherche ;
  - Optimisation *logique* et *physique*.

Architectures d'optimisation extensible

- Exodus [Carey et al. 90] :
  - Stratégie de recherche (coefficient d'amélioration) ;
- Volcano [Graefe et McKenna 93] :
  - Définition de stratégies de recherche ;
- EPOQ [Mitchell et al. 93] :
  - Objets ;
  - Stratégie de recherche (région d'optimisation) ;
- OPT++ [Kabra et DeWitt 99] :
  - Objets ;
  - Définition de stratégies de recherche ;
  - Optimisation *logique* et *physique*.

Architectures d'optimisation extensible

- Exodus [Carey et al. 90] :
  - Stratégie de recherche (coefficient d'amélioration) ;
- Volcano [Graefe et McKenna 93] :
  - Définition de stratégies de recherche ;
- EPOQ [Mitchell et al. 93] :
  - Objets ;
  - Stratégie de recherche (région d'optimisation) ;
- OPT++ [Kabra et DeWitt 99] :
  - Objets ;
  - Définition de stratégies de recherche ;
  - Optimisation *logique* et *physique*.

Architectures d'optimisation extensible

- Exodus [Carey et al. 90] :
  - Stratégie de recherche (coefficient d'amélioration) ;
- Volcano [Graefe et McKenna 93] :
  - Définition de stratégies de recherche ;
- EPOQ [Mitchell et al. 93] :
  - Objets ;
  - Stratégie de recherche (région d'optimisation) ;
- OPT++ [Kabra et DeWitt 99] :
  - Objets ;
  - Définition de stratégies de recherche ;
  - Optimisation *logique* et *physique*.

Architectures d'optimisation extensible

- Exodus [Carey et al. 90] :
  - Stratégie de recherche (coefficient d'amélioration) ;
- Volcano [Graefe et McKenna 93] :
  - Définition de stratégies de recherche ;
- EPOQ [Mitchell et al. 93] :
  - Objets ;
  - Stratégie de recherche (région d'optimisation) ;
- OPT++ [Kabra et DeWitt 99] :
  - Objets ;
  - Définition de stratégies de recherche ;
  - Optimisation *logique* et *physique*.

Motifs de règles (Rule Patterns)

Nous proposons un cadre de modélisation de règles :

Motif de règle condition => Motif de règle conclusion

Règles de transformation :

- Logiques ;
- Physiques ;

Règles de transformation

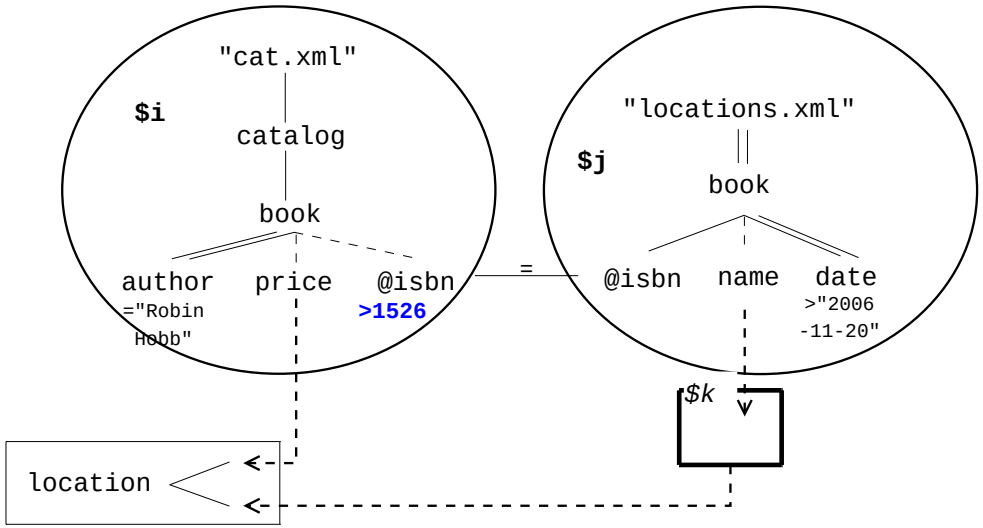
Optimisation logique : duplication de prédicat

R1 : $n_{\$c^1} \xrightarrow[\$H_1]{=} \$n^2$ => $n_{\$c^1} \xrightarrow[\$H_1]{=} \$n_{\$c^2}$

Règles de transformation

Optimisation logique : duplication de prédicat

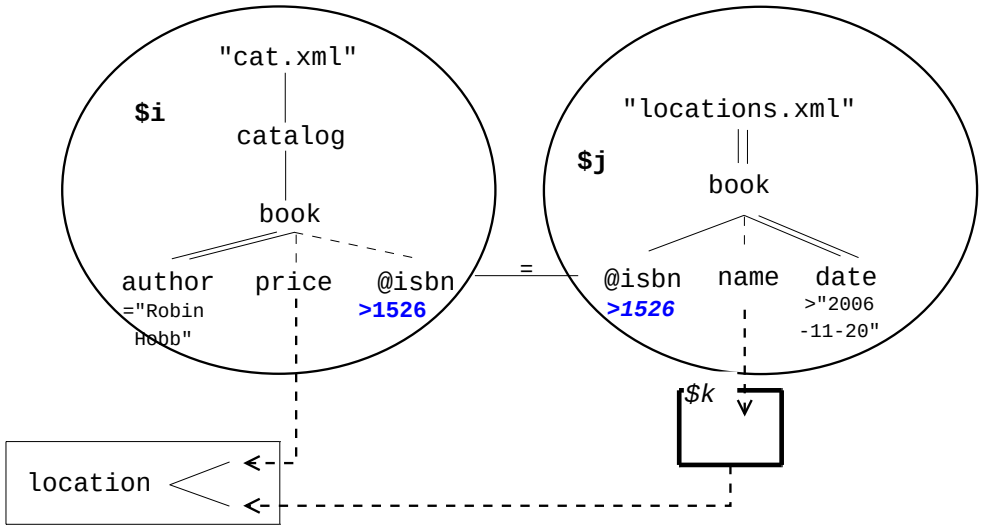
R1 : $n_1 \xrightarrow[\$H_1]{=} n_2$ $\Rightarrow$ $n_1 \xrightarrow[\$H_1]{=} n_2$ avec $\$c_1$ et $\$c_2$



Règles de transformation

Optimisation logique : duplication de prédicat

R1 : n_1 —(=, H_1)— n_2 c_1 c_2 => n_1 —(=, H_1)— n_2 c_1 c_2



Règles de transformation

Optimisation Physique : Modification Algorithme

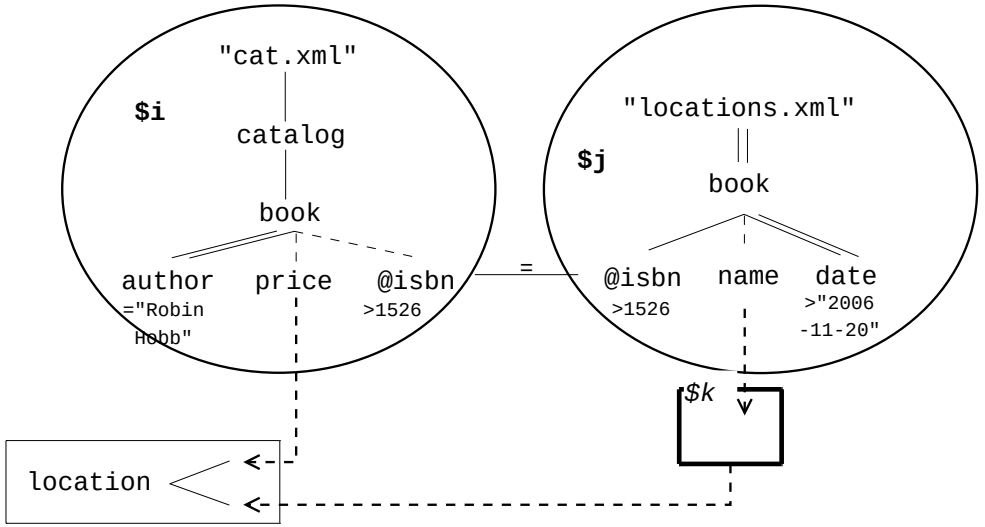
R2 : n_1 —(=, H_1)— n_2 => n_1 —(=, H_1)— n_2

(2) (1) (3) *(2) (1) (3)*

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

(2) << (3)

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



Règles de transformation

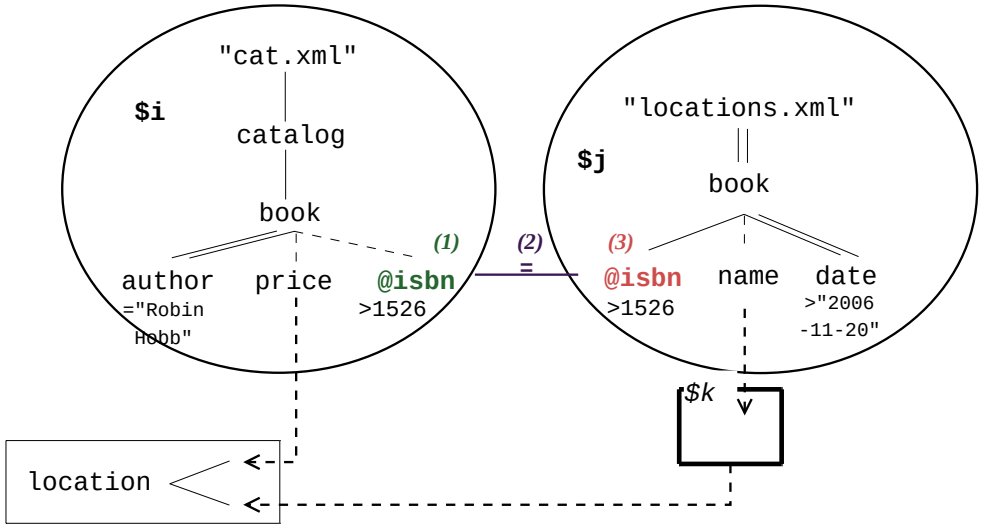
Optimisation Physique : Modification Algorithme

R2 : $n1 —(=, $H1)— $n2 (2) (1) (3) => $n1 —(=, $H1)— $n2 (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

Règles de transformation

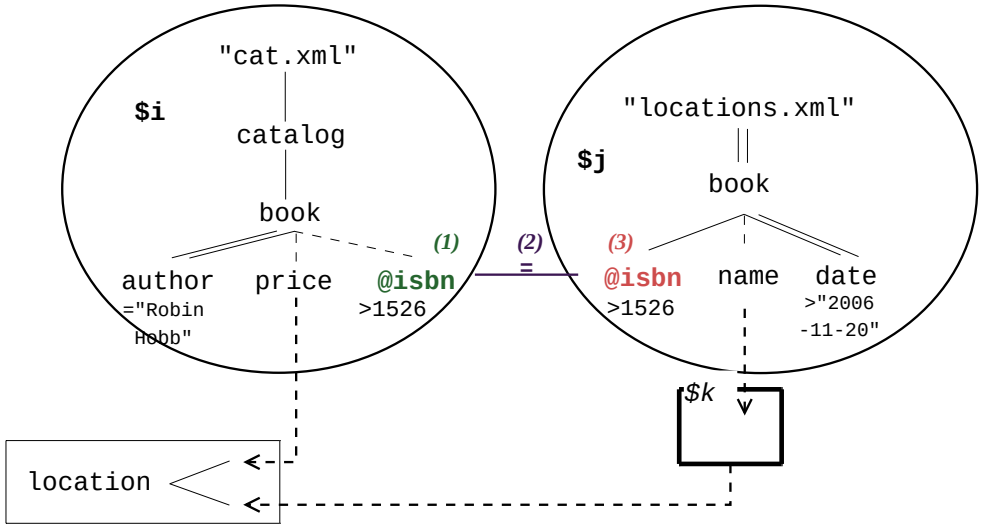
Optimisation Physique : Modification Algorithme

R2 : $n1 —(=, $H1)— $n2 (2) (1) (3) => $n1 —(=, $H1)— $n2 (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

Stratégie de recherche

Heuristique : coefficients d'amélioration sur chaque règle.

- Rapport de l'estimation avant/après transformation ;
- Attribué par calibrage des règles de transformations ;
- Utilisation d'un modèle de coût [Thèse Liu].

Stratégie de recherche pour orienter la génération de l'espace de recherche :

- Incrémentale ;
- Recuit simulé ;
- Génétique.

Stratégie de recherche

Heuristique : coefficients d'amélioration sur chaque règle.

- Rapport de l'estimation avant/après transformation ;
- Attribué par calibrage des règles de transformations ;
- Utilisation d'un modèle de coût [Thèse Liu].

Stratégie de recherche pour orienter la génération de l'espace de recherche :

- Incrémentale ;
- Recuit simulé ;
- Génétique.

Stratégie de recherche

Heuristique : coefficients d'amélioration sur chaque règle.

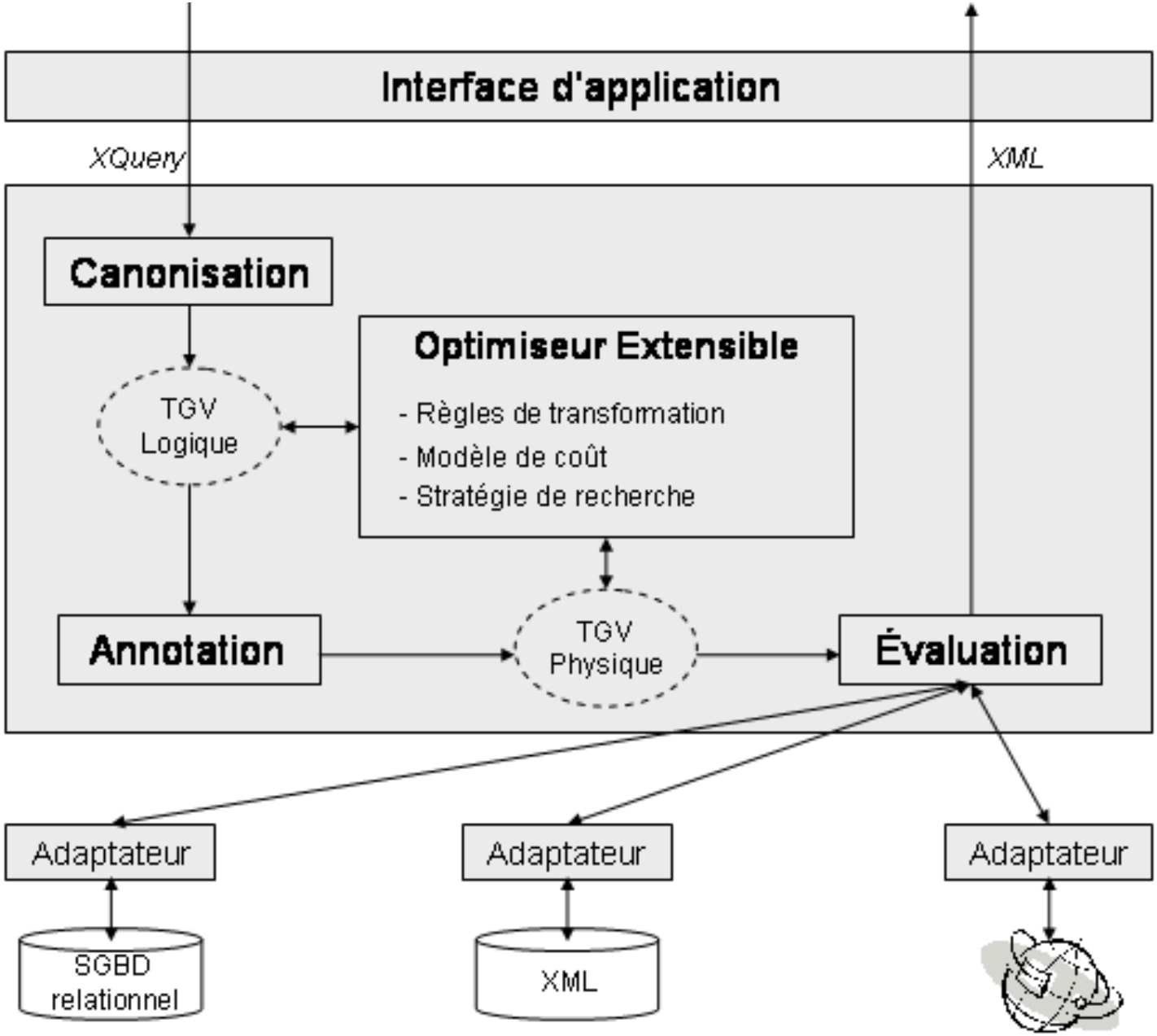
- Rapport de l'estimation avant/après transformation ;
- Attribué par calibrage des règles de transformations ;
- Utilisation d'un modèle de coût [Thèse Liu].

Stratégie de recherche pour orienter la génération de l'espace de recherche :

- Incrémentale ;
- Recuit simulé ;
- Génétique.

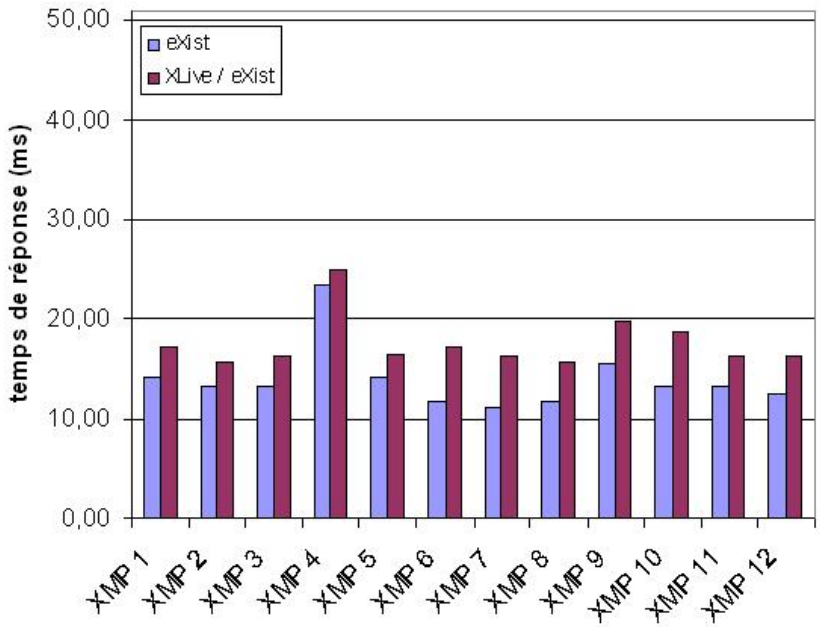
XLive

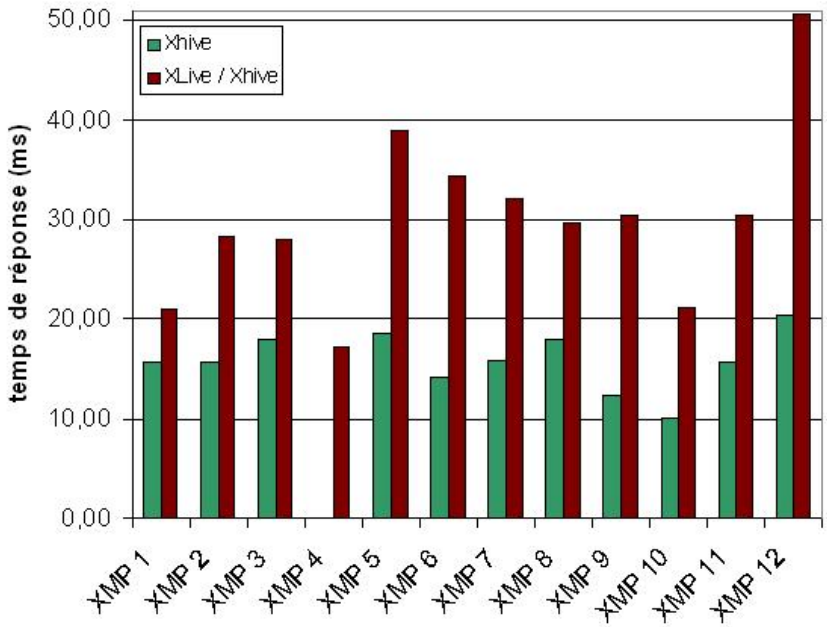
- Système de médiation XQuery/XML ;
- Tout XQuery non-typé ;
- Supporte les cas d'usage du W3C (non typés).


Interface d'application
XQuery
XML
Canonisation
Optimiseur Extensible
- Règles de transformation
- Modèle de coût
- Stratégie de recherche
TGV Logique
Annotation
TGV Physique
Évaluation
Adaptateur
Adaptateur
Adaptateur
SGBD relationnel
XML

Médiateur

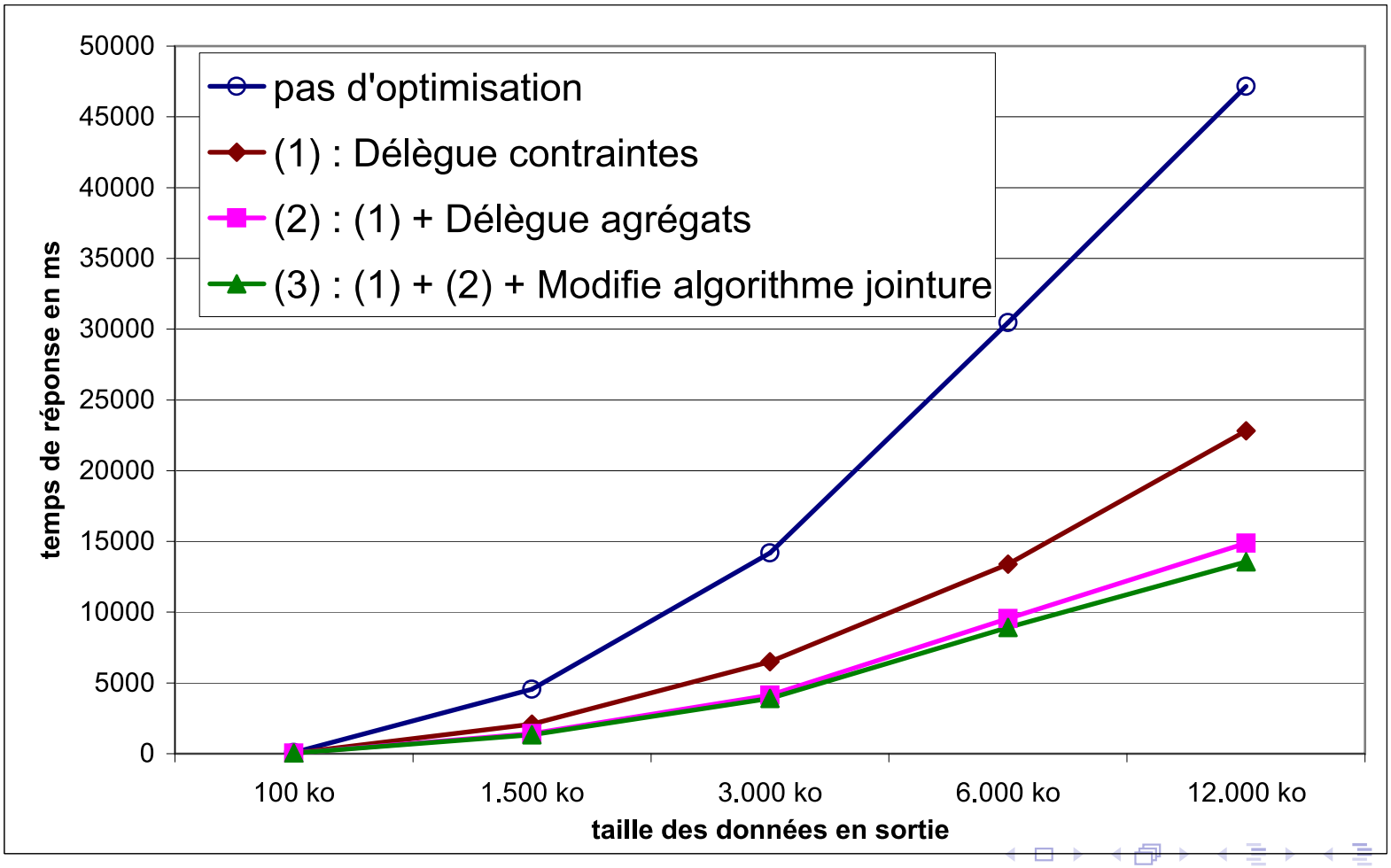
Requêtes des cas d'usage XMP sur *eXist* et *Xhive*.
Comparaison avec XLive.





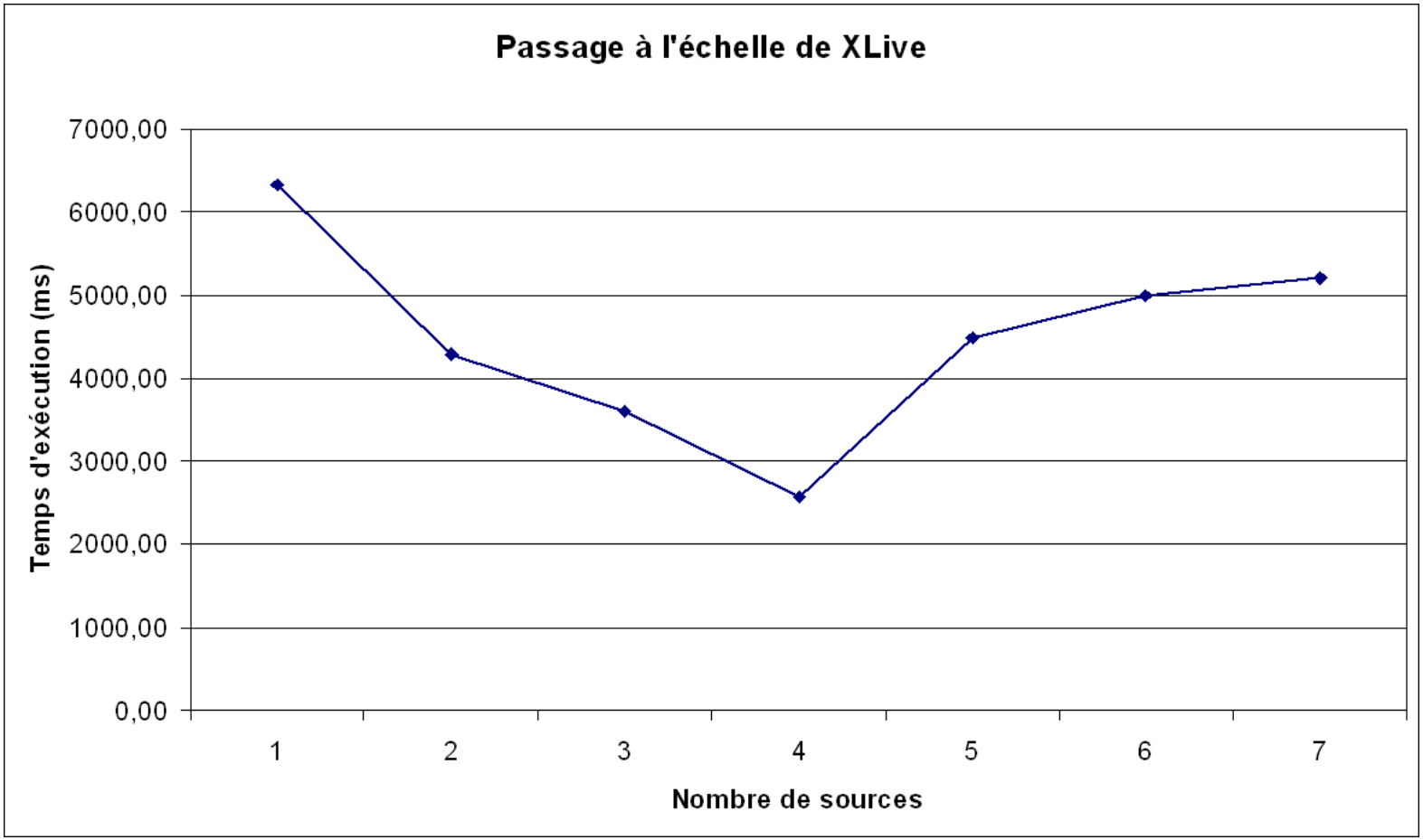
Optimiseur Extensible

Performances de l'optimiseur : Benchmark [Dragan et Gardarin 05].



Passage à l'échelle

Interrogation d'un jeu de données distribué sur plusieurs sources.



XLive & Projets

Utilisation dans des projets importants :

1. Projet Européen IST *WebSI* : Données hétérogènes ;
2. Projet Européen IST *Satine* : Services Web et P2P ;
3. Projet National ACI MD *SemWeb* : Données sémantiques et P2P ;
4. Projet National ANR *PADAWAN* : Capteurs.

Conclusion & Contributions

XLive : *Médiateur* tout *XQuery non typé* pour *XML* efficace grâce à son *optimiseur extensible*, validé par des projets de recherches.
Contributions :

- Règles de canonisation pour XQuery ;
- TGV, un modèle de représentation intuitif ;
- Motifs de règles, un cadre d'optimisation extensible ;
- Médiateur XLive, un système de médiation tout XML ;

[BDA 04]
[BDA 05]
[IBIS 06]
[DASFAA 07]
[TechReport 06a]
[TechReport 06b]

Perspectives

- Intégration des typages ;
- Améliorer la stratégie de recherche [Thèse Liu] ;
- Étendre aux vues matérialisées [Thèse Sans] ;
- Intégration dans les réseaux pair-à-pair ;
- Intégrer de nouvelles annotations aux TGV ;
- Intégration des mises à jour.

Merci !

Questions ?

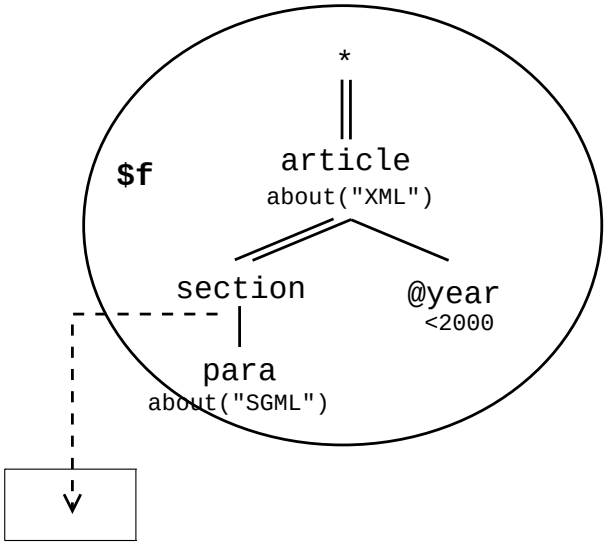
FAQ

- Références
- Algèbre abstraite
- La XAlgèbre
- Algèbre abstraite vs XAlgèbre
- Une évaluation à base d'annotation
- TGV et P2P
- Annotation de TGV
- Vues d'annotations
- TGV & sémantique
- TGV et requête floues
- équivalence XQuery / TGV / XAlgèbre
- Axes XPath
- XLive et les capteurs
- Et XMark ?
- Les résultats de Xhive
- Le passage à l'échelle
- Pourquoi XLive ?
- Le cycle des requêtes
- État de l'implémentation
- La stratégie de recherche
- Cas d'usage XMP
- TGV et indexation
- Les Types Abstraits de Données
- TGV et les Vues
- TGV et Vues matérialisées
- Les statements
- XQUpdate et TGV
- TGV et typage
- ℂ*Duce* et TGV
- Et NEXT ?
- Gestion des cycles
- Traitement des IDREF

Lier sémantique et TGV

- Fonction *about* (recherche sémantique) sur un document :

//article[about(.,"XML") and @year < 2000]
//section[about(para, "SGML")]



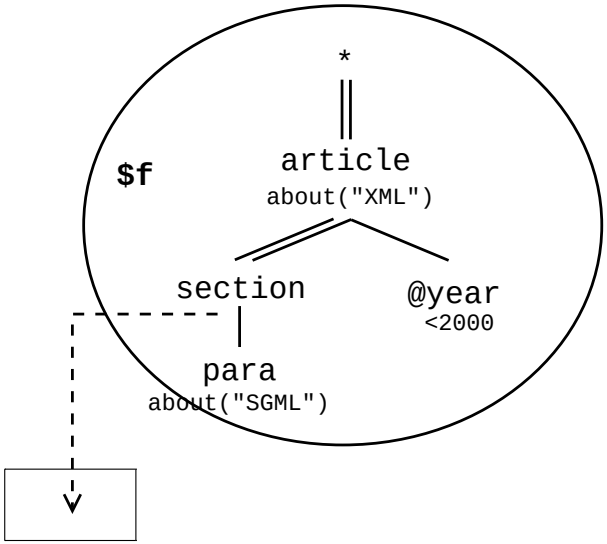
- Annotations sémantique (équivalence des éléments / métadonnées) ;
- Règles de transformation donnant un TGV sémantique ;

retour

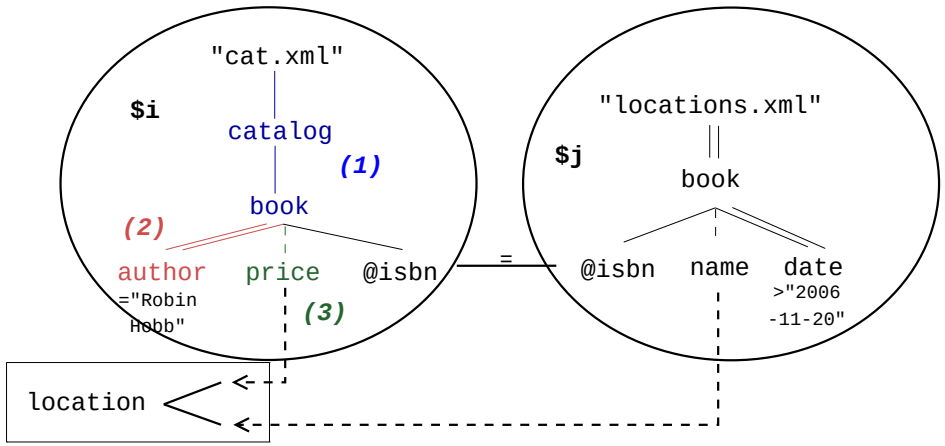
Lier sémantique et TGV

- Fonction *about* (recherche sémantique) sur un document :

//article[about(.,"XML") and @year < 2000]
//section[about(para, "SGML")]



- Annotations sémantique (équivalence des éléments / métadonnées) ;
- Règles de transformation donnant un TGV sémantique ;



(1) semantique : /catalogue/livre

(2) semantique : //auteur

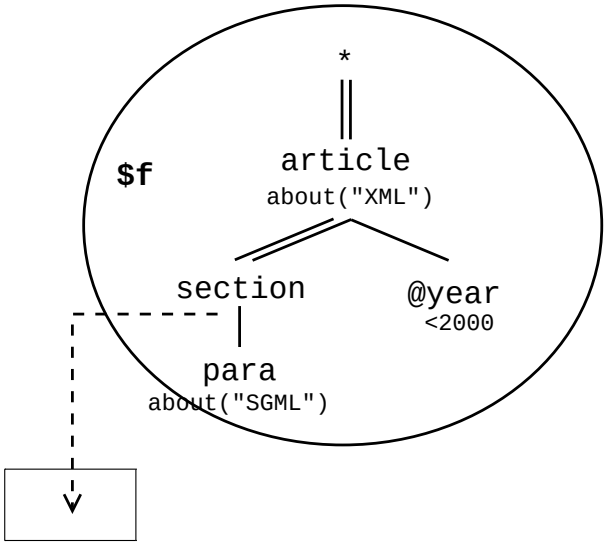
(3) semantique : /prix

retour

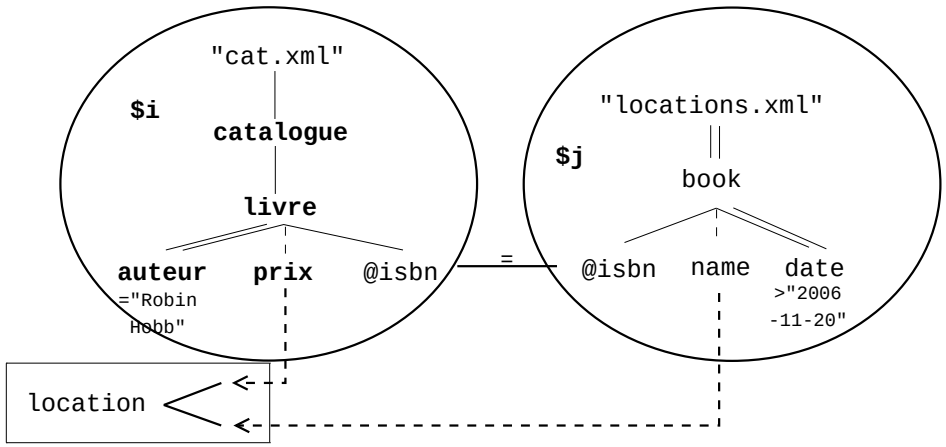
Lier sémantique et TGV

- Fonction *about* (recherche sémantique) sur un document :

//article[about(.,"XML") and @year < 2000]
//section[about(para, "SGML")]



- Annotations sémantique (équivalence des éléments / métadonnées) ;
- Règles de transformation donnant un TGV sémantique ;



retour

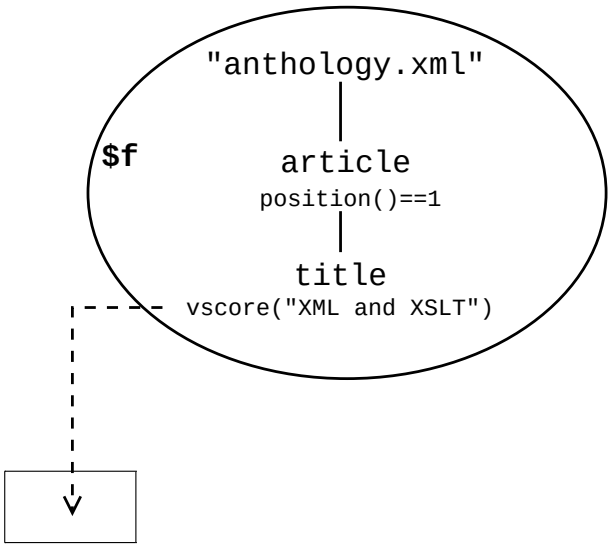
Requêtes floues et TGV

Ajouter des annotations de degré de vérité sur les éléments du TGV.

Mais aussi ajout de contraintes [Le Maitre 2005] : retour

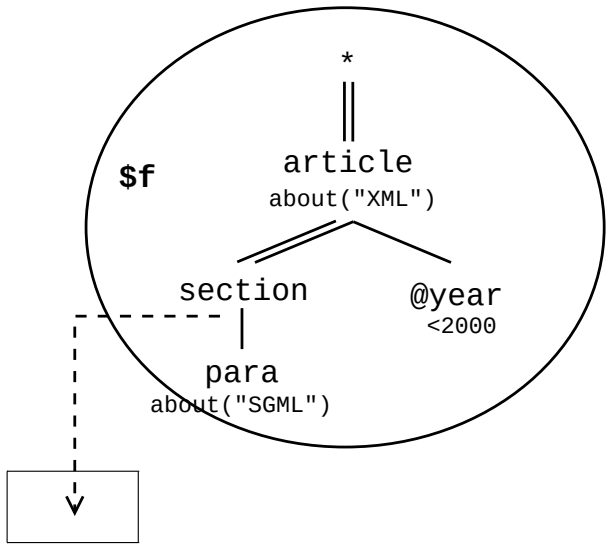
vscore (degré de similarité) :

vscore(doc("anthology.xml")/article[1]/title, "XML and XSLT")



about (recherche sémantique) :

//article[about(.,"XML") and @year < 2000]//section[about(para, "SGML")]



Annotation des TGV

Chaque type d'annotation défini dans le médiateur ce qu'il faut annoter :

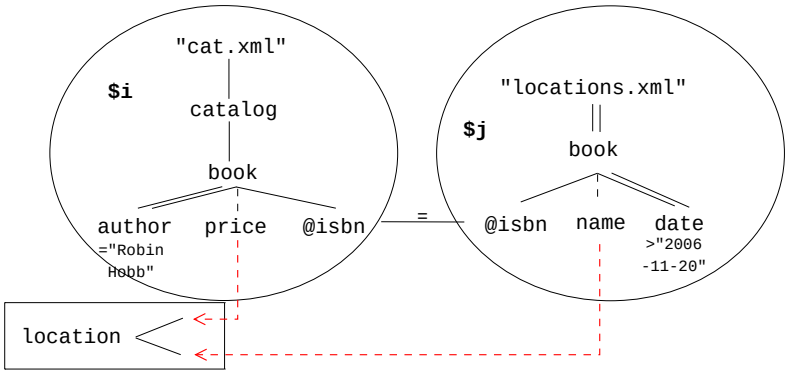
- Modèle de coût : formule en fonction des éléments présents ;
- Algorithme : opérateurs en fonction des motifs et hyperliens ;
- Localisation : en fonction des métadonnées et motifs.

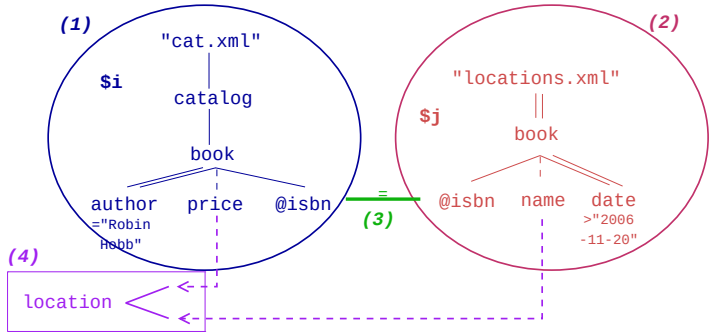
retour

Vues sur les annotations

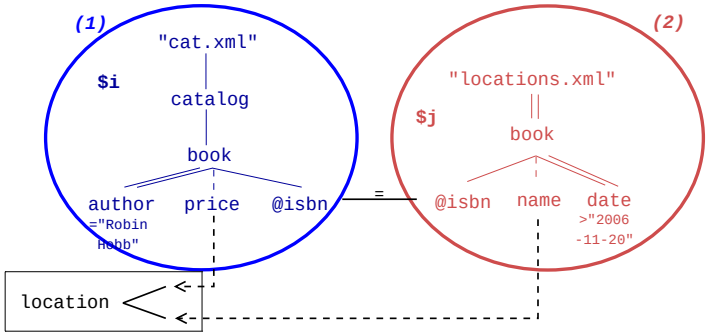
Pour chaque type d'annotation une vue est définie :

retour

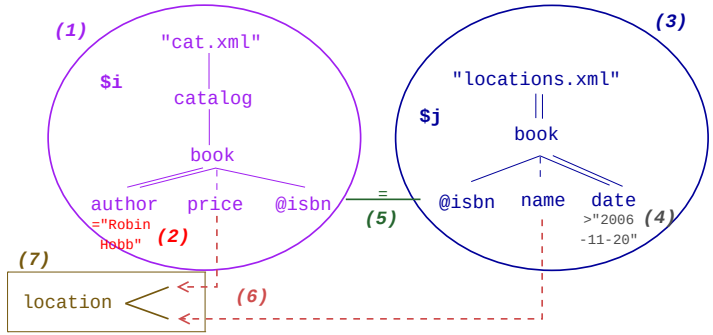




(1) XSource
(2) XSource
(3) XJoin Hash
(4) XConstruct

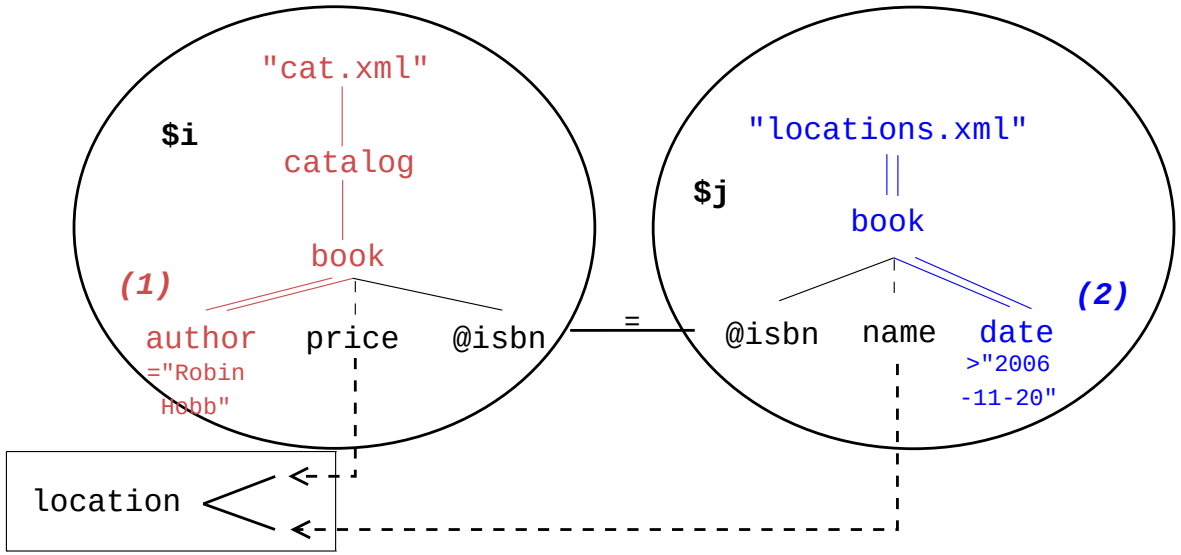


(1) source 1
(2) source 2



(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

TGV et indexation



(1) index :
s1 (id1)
s1 (id2)
s3 (id10)

(2) index :
s2 (id20)
s4 (id10)

retour

TGV et vues matérialisées

Besoin de remettre les données matérialisées à jour :

- La vue est une requête représentée par un *TGV* ;
- Ajout d'une annotation de type "*TimeStamp*" ;
- Mise à jour de la vue avec les sous parties nécessaires du TGV.

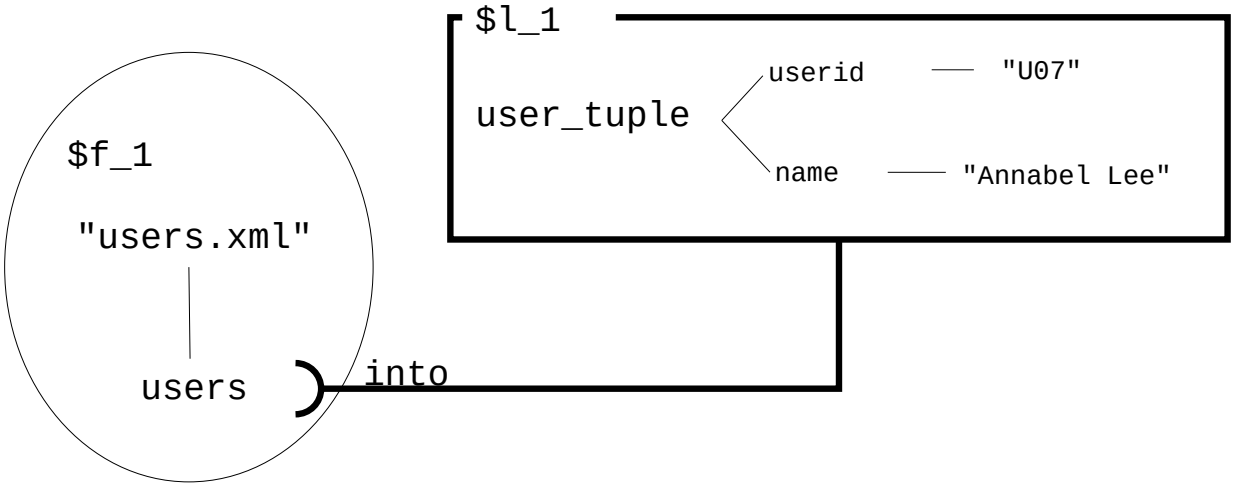
retour

XQUpdate et TGV

Catégorisation des opérations de mises à jour. Nouveaux hyperliens avec flèches particulières :

- INSERT INTO/AFTER/BEFORE ") " (flèches annotées) ;
- DELETE " × " ;
- REPLACE " ⊸ " ;
- RENAME " ⊢ " ;

Exemple :

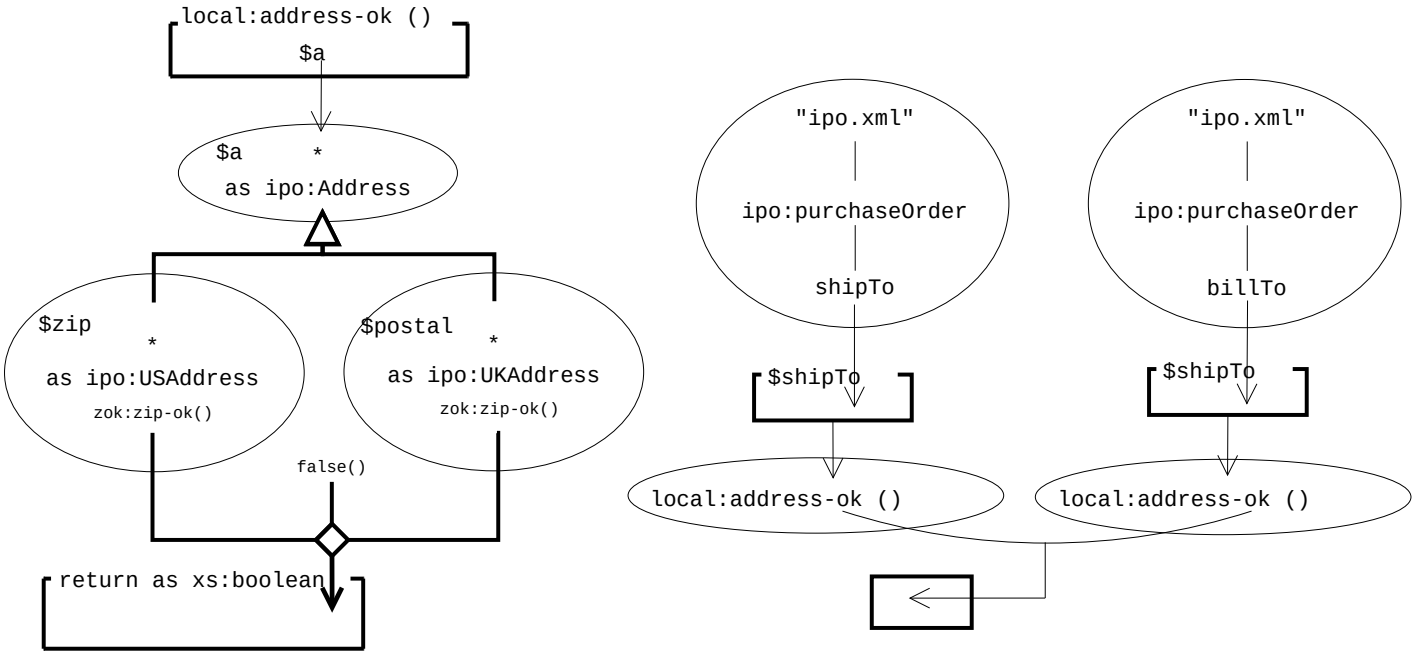


retour

Typage XQuery et TGV

Catégorisation des opérations de typage :

- Contraintes : *as*, *castable*, *Kind-Test* ;
- Hyperlien directionnel : *cast as*, *typeswitch*, *instanceof*, *treat as*.



retour

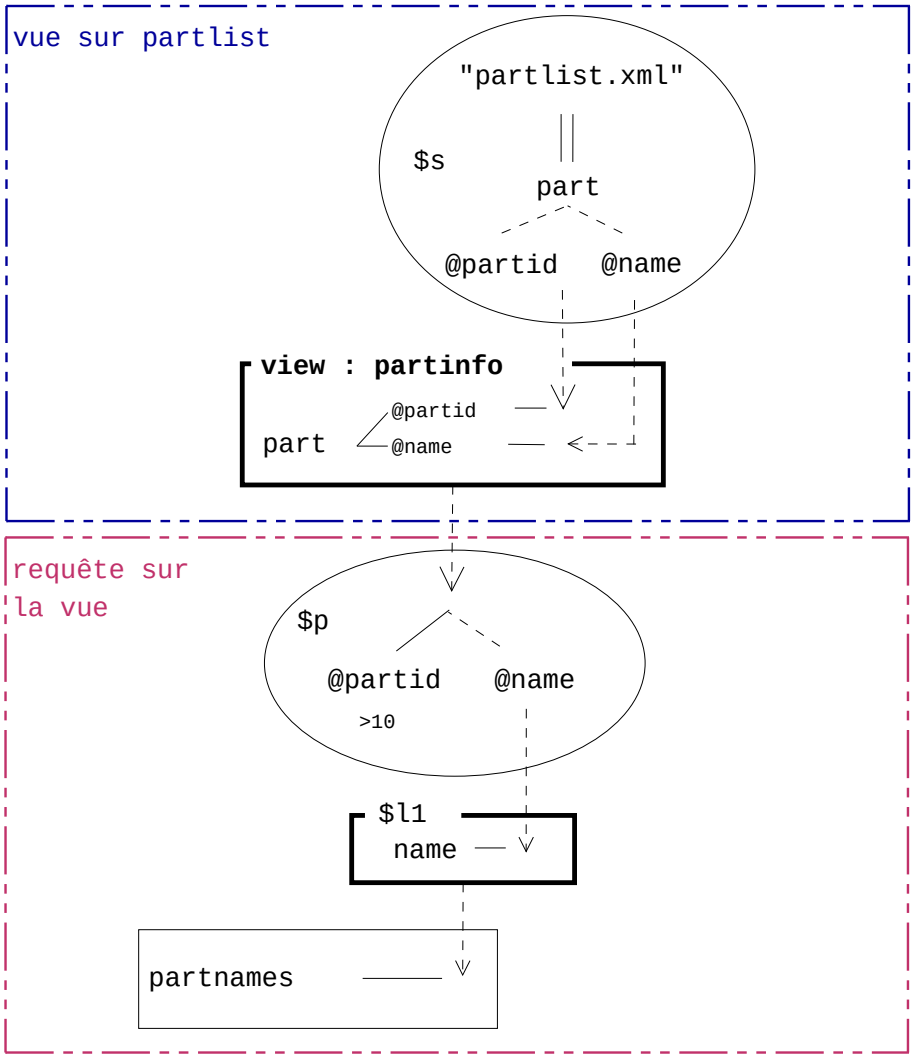
Requête cas d'usage de STRONG (4)

```
define function address-ok($a as element(*, ipo :Address))
as xs :boolean
{       typeswitch ($a)
            case $zip as element(*, ipo :USAddress)
                return zok :zip-ok($zip)
            case $postal as element(*, ipo :UKAddress )
                return pok :postal-ok($postal)
            default return false()
}
for $p in doc("ipo.xml")//element(ipo :purchaseOrder)
where not( address-ok($p/shipTo) and address-ok($p/billTo))
return $p
```

retour

Utilisation de vues avec les TGV

Définition d'un vue comme une fonction de TGV : retour



Et les statement ?

Le TGV définit alors :

- Fonction paramétrée ;
- Optimisé en tenant compte des paramètres ;
- Intégré dans le système de médiation ;

retour

Les Types Abstraits de Données

Définit de manière unique un type pour les TGV.
Permet l'implémentation dans tout système.

NodeLink NL
Utilise : N, Axis, int, boolean
Opérations

O1	createNodeLink	N × N × int × Axis × boolean	→ NL
O2	getPosition	NL	→ int
O3	getLinkAssociation	NL	→ Axis
O4	isMandatory	NL	→ boolean
O5	getPreviousNode	NL	→ N
O6	getNextNode	NL	→ N

Pré-conditions : $n \in N$; $i \in int$;

P1	define (getNodeLink (n))	$\iff \neg$ isRoot (n)
P2	define (getNextNodeLink (n, i))	$\iff$ i >= 0 && i < getSizeNodeLink (n)

Axiomes : $s \in string$; $n_1, n_2 \in N$; $pos \in int$; $a \in Axis$; $m \in boolean$; , $nl \in NL$;

A1	getPosition (createNodeLink (n_1, n_2, pos, a, m)) = pos
A2	getLinkAssociation (createNodeLink (n_1, n_2, pos, a, m)) = a
A3	isMandatory (createNodeLink (n_1, n_2, pos, a, m)) = m
A4	getPreviousNode (createNodeLink (n_1, n_2, pos, a, m)) = n_1
A5	getNextNode (createNodeLink (n_1, n_2, pos, a, m)) = n_2

▸ retour

ℂ*Duce* et TGV

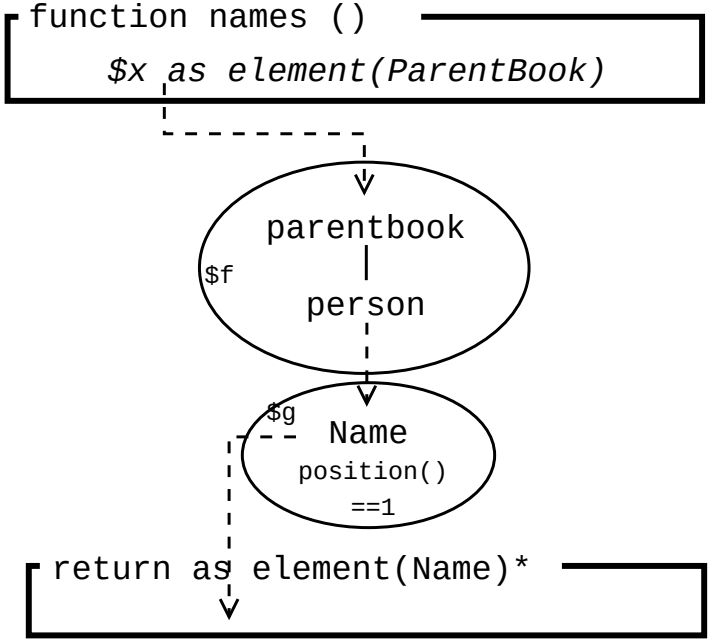
ℂ*Duce* :

- fortement typé ;
- Manipulation avec mappings ;
- Gestion de Séquences.

```
let names (ParentBook -> [Name*])
<parentbook>x -> (map x with <person ..>[ n _*] -> n)
```

TGV :

- Recherche arborescente et textuelle ;
- Manipulation d'arbres ;



retour

Les Cycles dans les TGV

Possible à cause des fonctions récursives.
Vérification à effectuer dans le médiateur grâce aux métadonnées.

retour

Traitement des IDREF avec TGV

XQuery défini la fonction :

`fn :idref ($arg as xs :string) as node()*.`

Nécessite un opérateur pour retrouver l'élément ciblé.

retour

Les cas d'usage XMP

Nom	Description
Q1	Domaine simple, double prédicat, projection d'un élément et d'un attribut
Q2	Triple définition de domaine, projection d'éléments
Q3	Définition de domaine simple
Q4	Let, fonctions, filtres, ordonnancement, quantificateur 'some', imbrication de requête
Q5	Double domaine de définition, jointure
Q6	Fonction d'agrégation, imbrication avec filtre, opération conditionnelle
Q7	Double prédicat, ordonnancement
Q8	let, redéfinition de domaine, filtre complexe, élément '*' dans un XPath, fonction
Q9	Opération ensembliste sur domaine, fonction "contains"
Q10	Let, redéfinition de domaine, filtre, fonction d'agrégat
Q11	Deux requêtes imbriquées
Q12	Double définition de domaine, let, ordonnancement, prédicat de jointure, composition de contraintes

Une algèbre abstraite

- Permet de définir un cadre d'évaluation à partir d'un TGV ;
- Facilite l'intégration dans des évaluateurs existants ;
- Permet la transformation en *XAlgèbre* présente dans XLive ;

$\Rightarrow$ A terme, définir une *évaluation* directe des TGV.

retour

Une évaluation des TGV

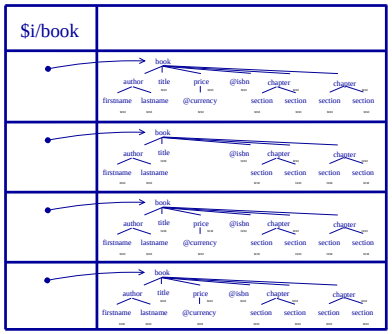
- Annotation de XTuples évalués ;
- Règles de transformation pour l'évaluation ;

⇒ Optimisation dynamique ;

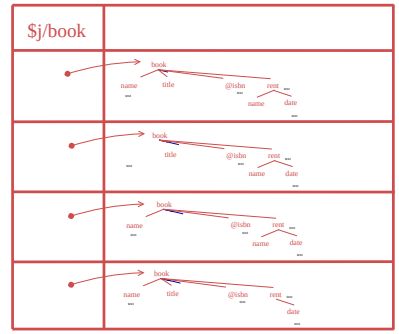
▸ Voir l'évaluation annotée

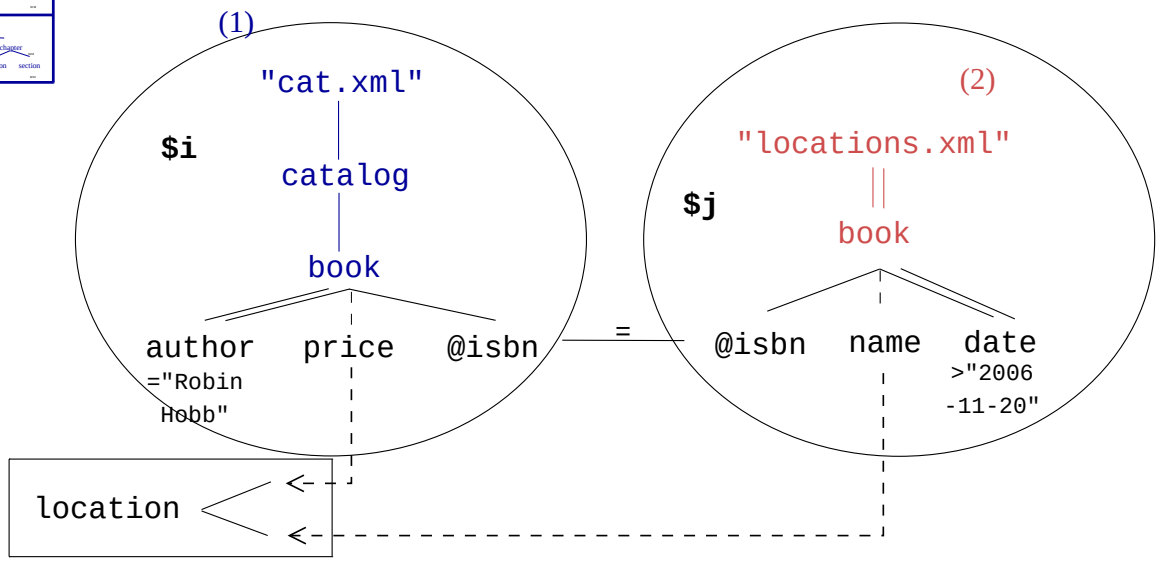
▸ retour

(1) evaluation

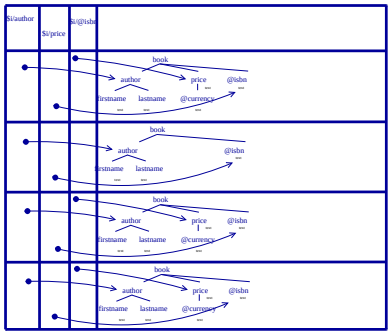


(2) evaluation

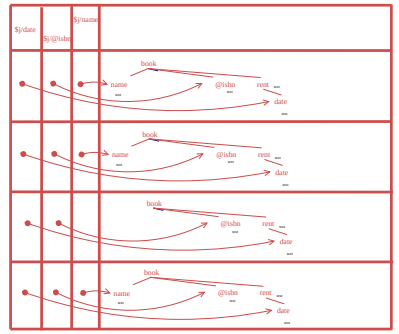


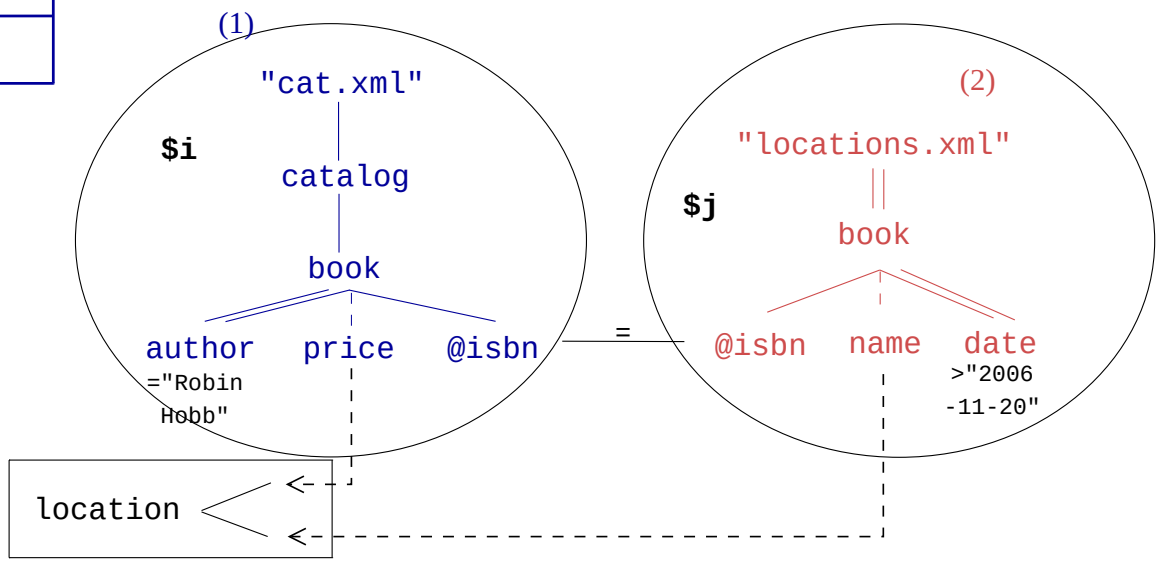


(1) evaluation



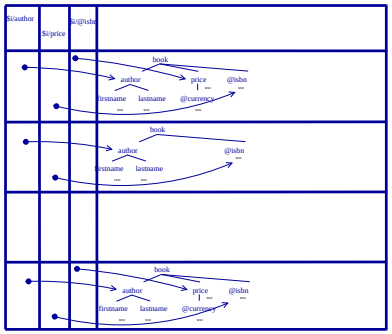
(2) evaluation



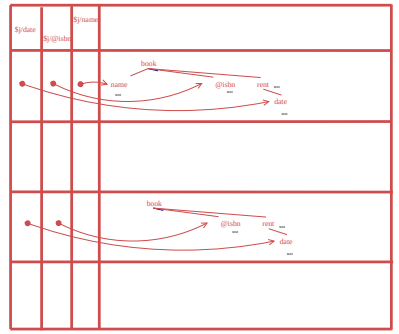


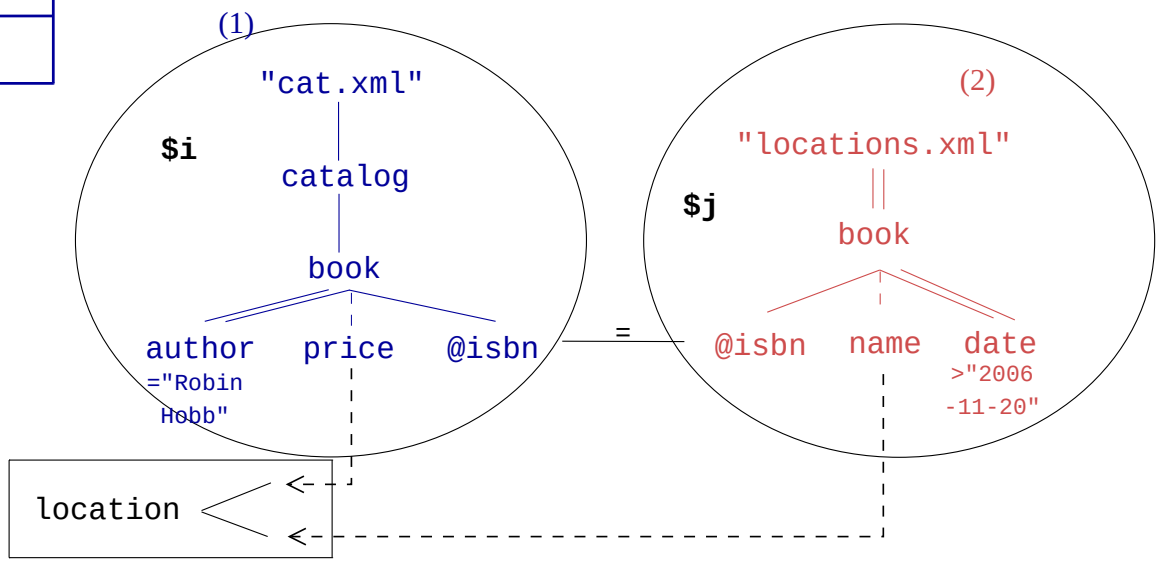
Règle de transformation de filtrage par motif d'arbre

(1) evaluation



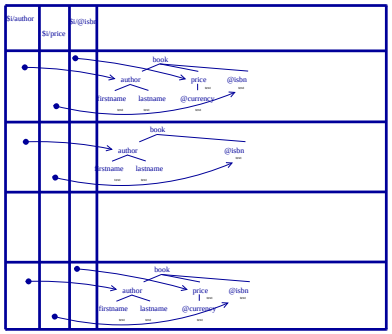
(2) evaluation



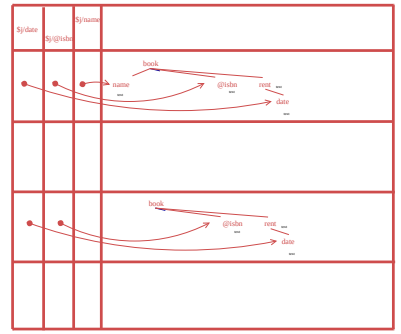


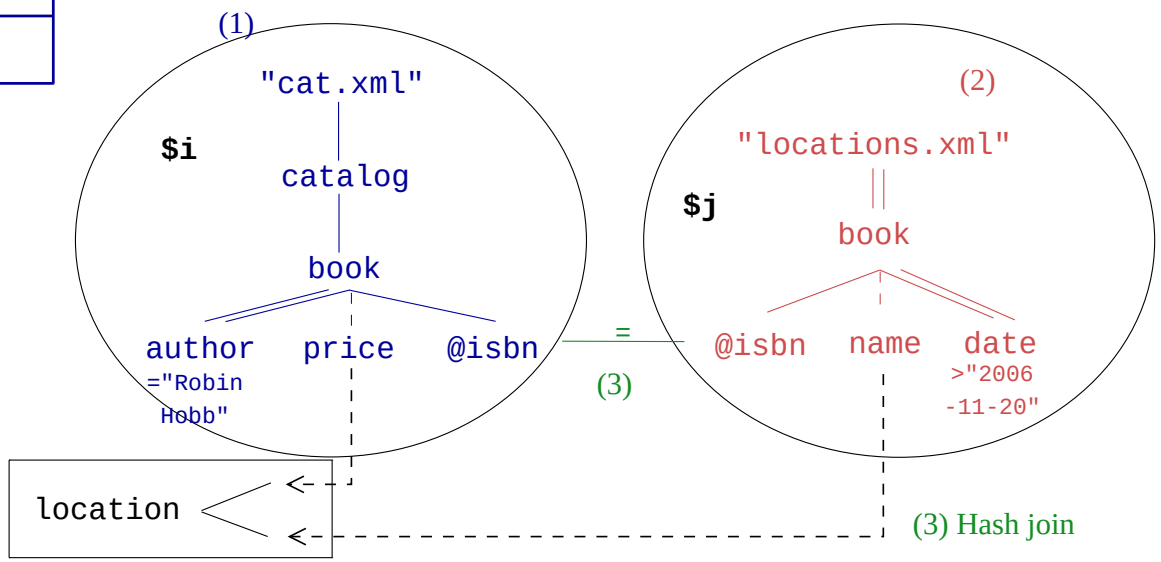
Règle de transformation de restriction d'arbres par contraintes

(1) evaluation

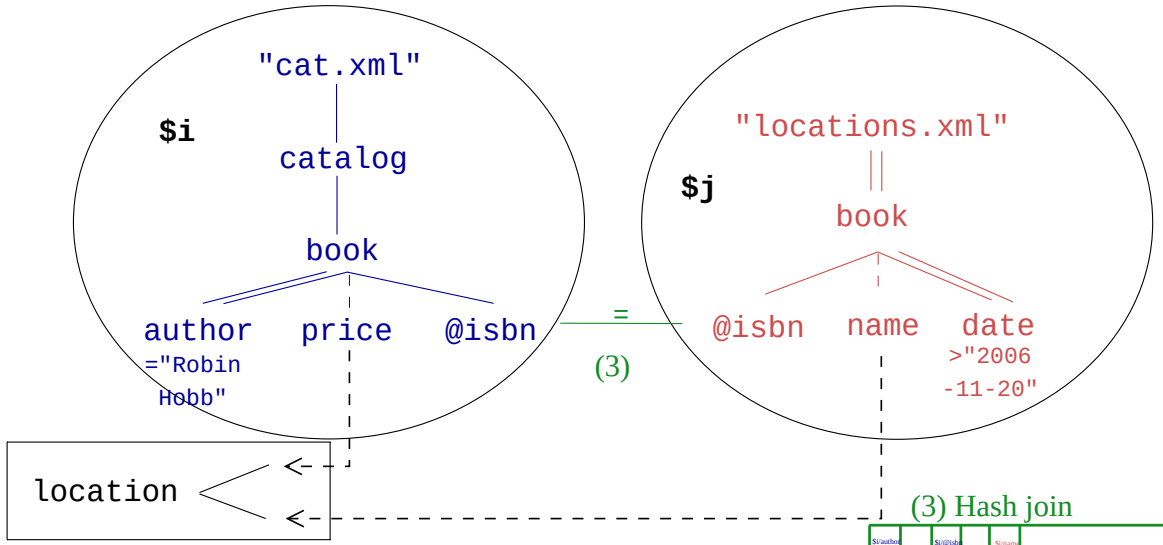


(2) evaluation

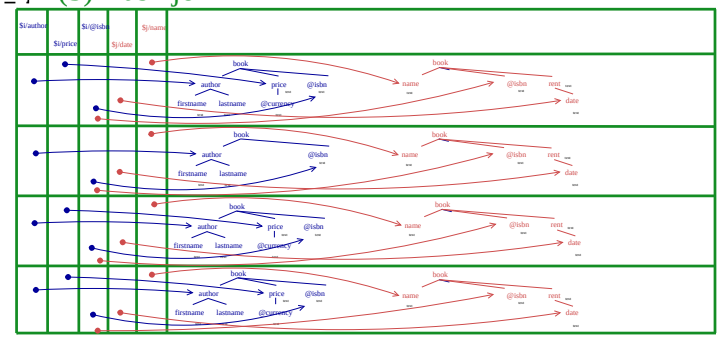




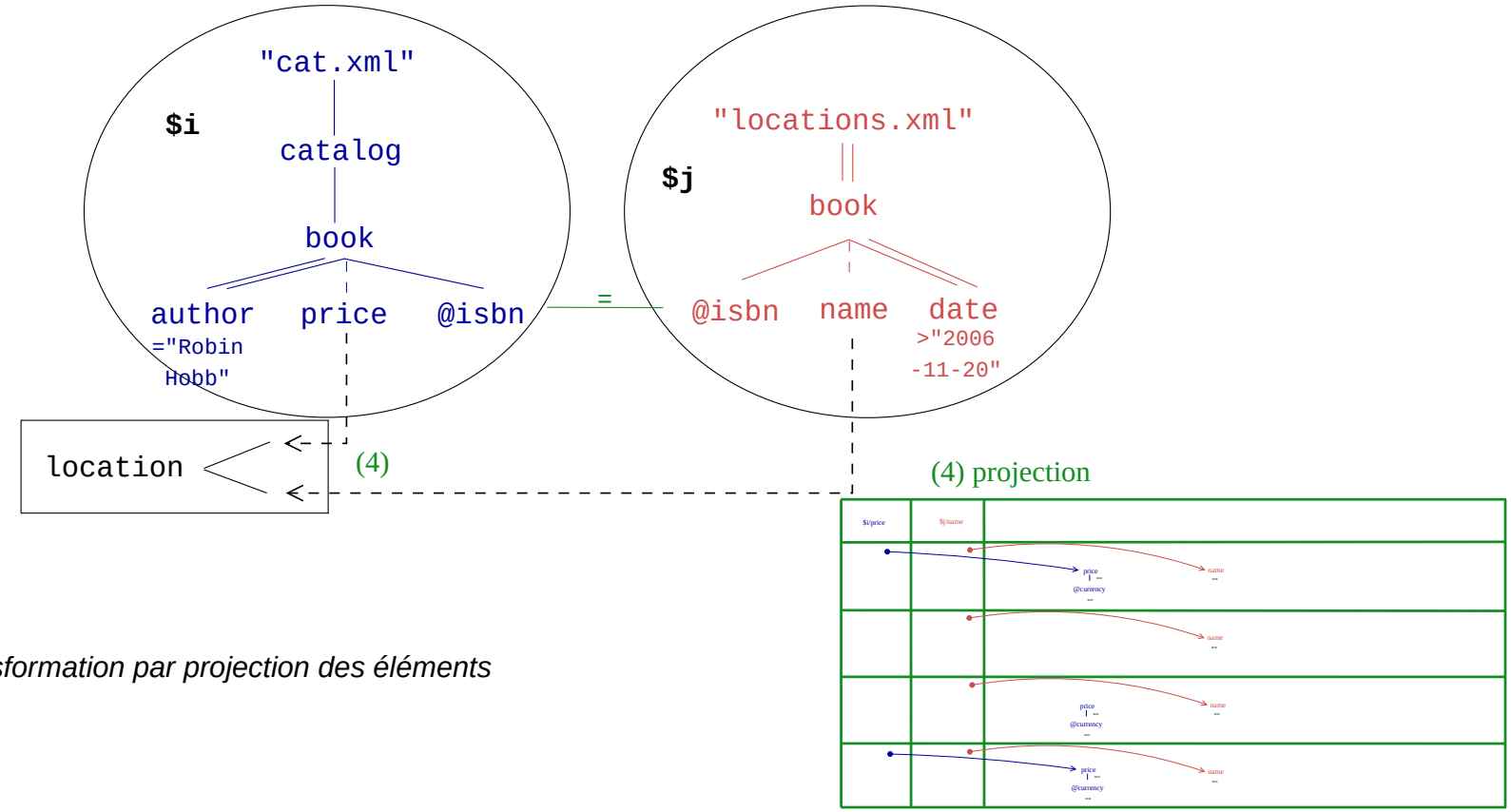
Règle de transformation de jointure d'arbre, avec algorithme



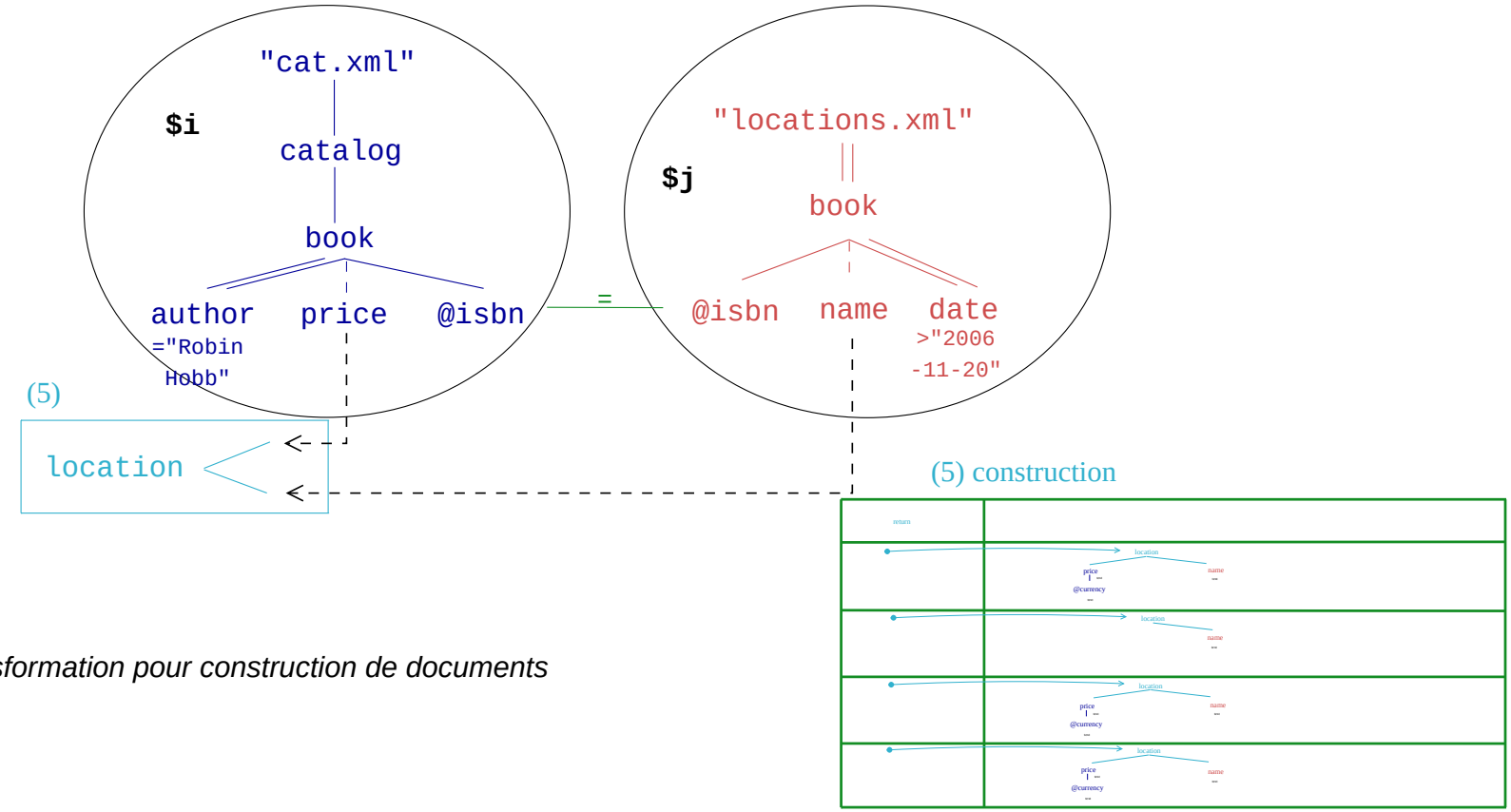
(3) Hash join



Règle de transformation de jointure d'arbre, avec algorithme



Règle de transformation par projection des éléments



Règle de transformation pour construction de documents

XQuery / TGV / XAlgèbre

- Correspondance directe XQuery / TGV ;
- Entre TGV et *XAlgèbre*, deux opérateurs manquants :
  - Opérations conditionnelles (If/Then/Else) ;
  - Utilisation de fonctions locales ;

retour

Algèbre abstraite & XAlgèbre

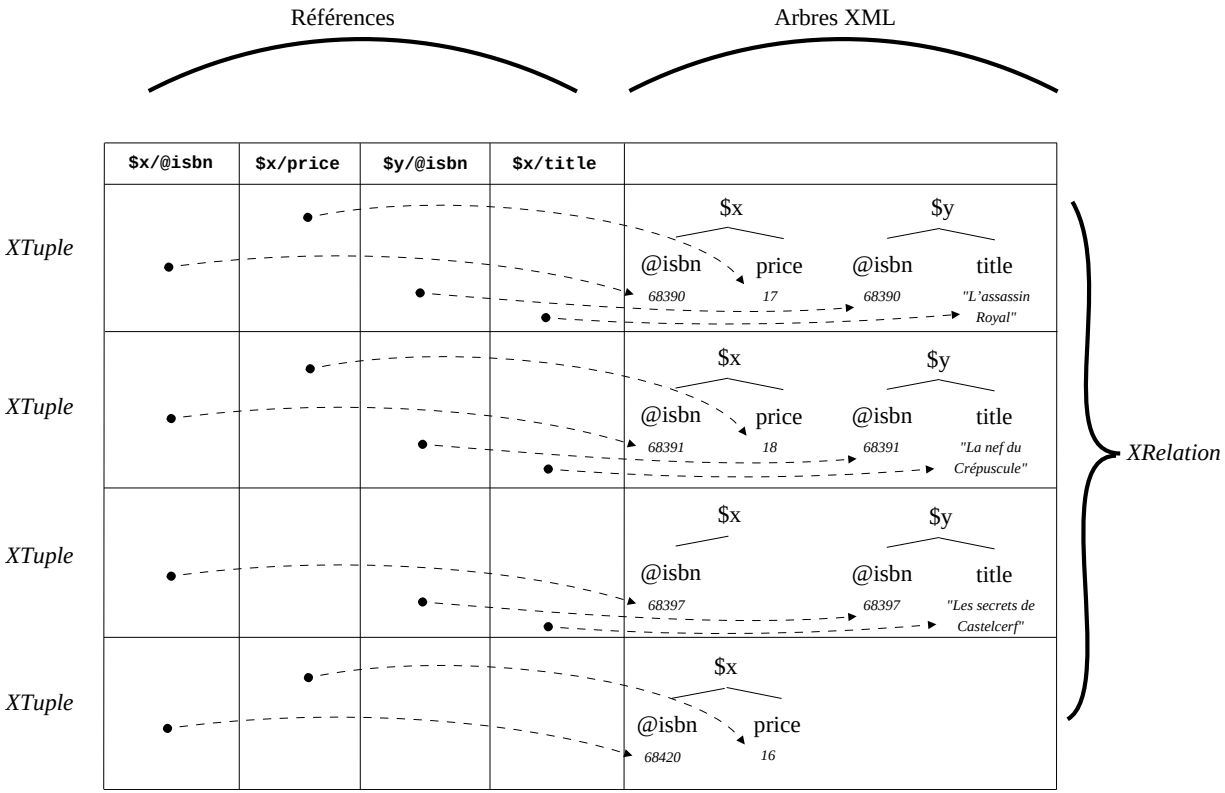
Règles de conversion TGV/Algèbre abstraite.
Algèbre abstraite = base d'évaluation.
⇒ Adaptabilité vers algèbre physique en fonction du système.
Exemple : *XAlgèbre*

TGV	**Algèbre Abstraite**	**XAlgèbre**
STP	$\phi(stp, \tau)$	XSource
ITP	$\phi(itp, \pi(itp, \tau))$	XSource
RTP + links	$\beta(rtp, \pi(rtp, \tau))$	XConstruct
ATP	$\alpha(atp, \tau)$ $\alpha(atp, \tau)$	XMin, ... XOrderBy
Constraint	$\sigma(rtp, \tau_r)$	XConstraint
JoinHyperlink	$\sigma(rtp, \tau_r)$	XJoin
SetHyperlink	$\sigma(rtp, \tau_r)$	XUnion, ...
IfThenElseHyperlink	$\sigma(rtp, \tau_r)$	XIf

retour

La XAlgèbre

XOpérateurs, manipulent des références des XTuples :



- XSource (filtre de document)
- XJoin (jointure d'XTuples)
- XConstraint (restriction)
- XOrderBy (ordonnancement)
- XUnion, XInter, XDiff (ensembliste)
- XMin, XMax, XCount, XAvg, XSum, etc... (agrégat)
- XIf (conditionnel)
- XConstruct (Construction de document)

retour

TGV et P2P

- Médiateur XLive est une source de données ;
- Modification de l'*évaluation* des TGV ;
- Règles de transformation sur l'évaluation en P2P.

retour

Les axes particuliers de XPath

Règles de canonisation de XPath [Olteanu et al. 2002]

- m/parent : :n ;
- m/ancestor : :n | m/ancestor-or-self : :n ;
- m/descendant-or-self : :n ;
- preceding | preceding-sibling ;
- following | following-sibling ;

Requête XQuery	Requête XQuery Canonique

retour

Les axes particuliers de XPath

Règles de canonisation de XPath [Olteanu et al. 2002]

- m/parent : :n ;
- m/ancestor : :n | m/ancestor-or-self : :n ;
- m/descendant-or-self : :n ;
- preceding | preceding-sibling ;
- following | following-sibling ;

Requête XQuery	Requête XQuery Canonique
for $i in doc("cat.xml")/catalog//title/ /**parent : :**book return {$i}	for $f in doc("cat.xml")/catalog//book where exists ($f/title) return {$i}

retour

Les axes particuliers de XPath

Règles de canonisation de XPath [Olteanu et al. 2002]

- m/parent : :n ;
- m/ancestor : :n | m/ancestor-or-self : :n ;
- m/descendant-or-self : :n ;
- preceding | preceding-sibling ;
- following | following-sibling ;

Requête XQuery	Requête XQuery Canonique
for $i in doc("cat.xml")/catalog//title/ /**ancestor : :**book return {$i}	for $i in doc("cat.xml")/catalog//book where exists ($i//title) return {$i}

retour

Les axes particuliers de XPath

Règles de canonisation de XPath [Olteanu et al. 2002]

- m/parent : :n ;
- m/ancestor : :n | m/ancestor-or-self : :n ;
- m/descendant-or-self : :n ;
- preceding | preceding-sibling ;
- following | following-sibling ;

Requête XQuery	**Requête XQuery Canonique**
for $i in doc("cat.xml")/catalog//title/ /**ancestor-or-self : :**book return {$i}	let l_1$:= for $f in doc("cat.xml")/catalog//book return ($f//title | $f) for $i in l_1$ where exists (l_1$) return {$i}

retour

Les axes particuliers de XPath

Règles de canonisation de XPath [Olteanu et al. 2002]

- m/parent : :n ;
- m/ancestor : :n | m/ancestor-or-self : :n ;
- m/descendant-or-self : :n ;
- preceding | preceding-sibling ;
- following | following-sibling ;

Requête XQuery	**Requête XQuery Canonique**
for $i in doc("cat.xml")/catalog/book /**descendant-or-self : :**title return {$i}	let $$l_1$$:= for $f in doc("cat.xml")/catalog/book return **($f//title | $f)** for $i in $$l_1$$ return {$i}

retour

Et NEXT ?

- Règles de canonisation pour forte désimbrication ;
- Génération de clause "`Group By`" ;
- Ne produit pas de forme canonique adéquate.

retour

XLive et les capteurs (PADAWAN)

Définition d'un adaptateur spécifique, pour :

- Des petites données (structuration nécessaire) ;
- Des mises à jour fréquentes et variées (stratégie de mise à jour) ;
- Un réseau Ad-hoc (besoin de cache).

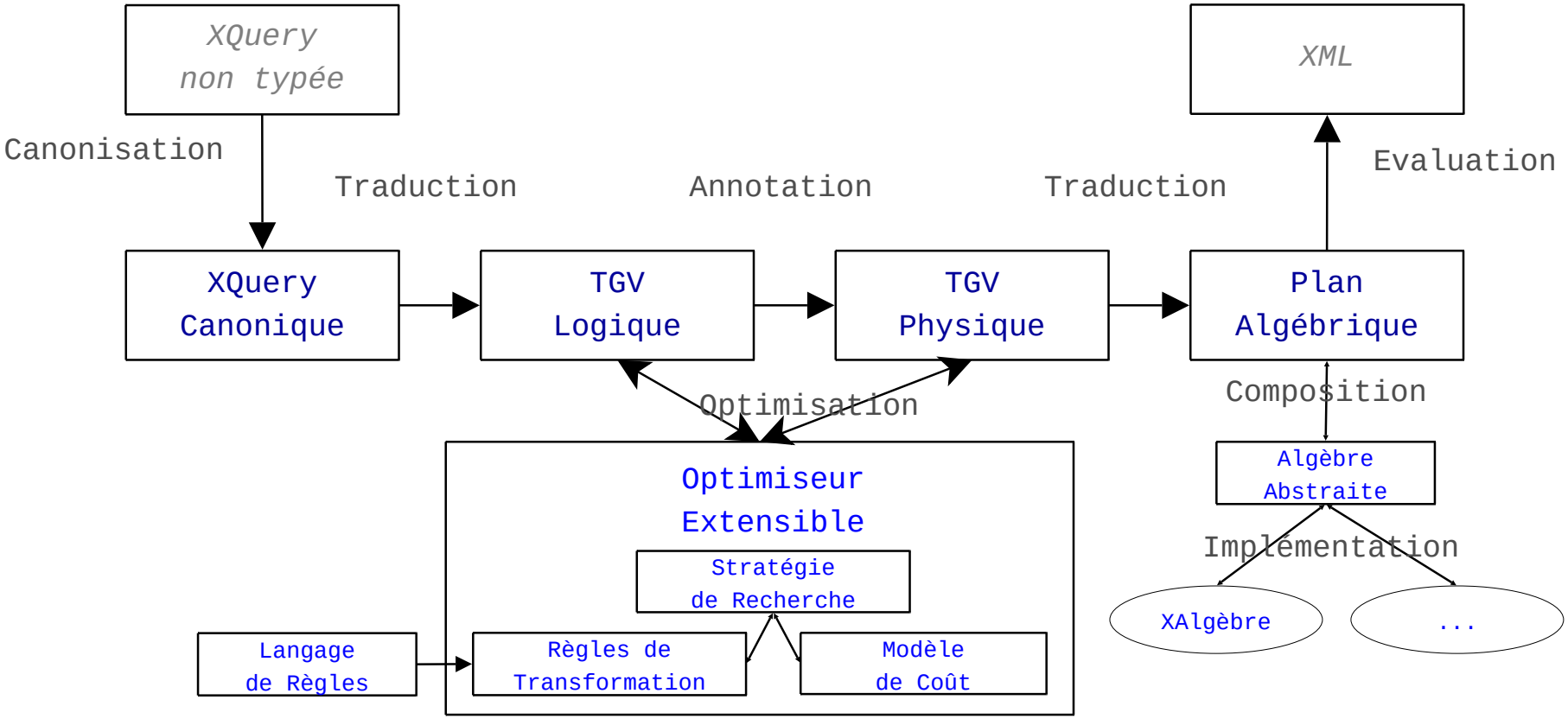
retour

XLive : An XML **Light** Integration Virtual Engine

- Système générique de médiation ;
- Micro-noyau ;
- plugins :
  - TGV (Modèle générique) ;
  - Cadre d'optimisation (Stratégie générique) ;
  - Modèle de coût (MathML : générique).

retour

Le cycle de traitement d'une requête



retour

Qu'est-ce qui est implémenté ?

- Canonisation ;
- Traduction XQuery canonique vers TGV ;
- Représentation des TGV ;
- Évaluation des TGV avec la *XAlgèbre* ;
- Annotations des TGV ;
- Cadre d'optimisation ;

retour

Quelle est la stratégie de recherche ?

Heuristique :

- Paire : *Règle de transformation / Coefficient*

Modèle de coût [Thèse Liu] :

- Sources connues ou autonomes ;
- Formule de coût I/O ou pages ;
- Calibration des sources ;
- Historique ;

retour

Pourquoi ne pas utiliser XMark ?

- Données non caractéristique de la médiation ;
- Manque de données orientées textes ;

$\Rightarrow$ Benchmark de [Florin et Gardarin 2005] facilite la médiation.

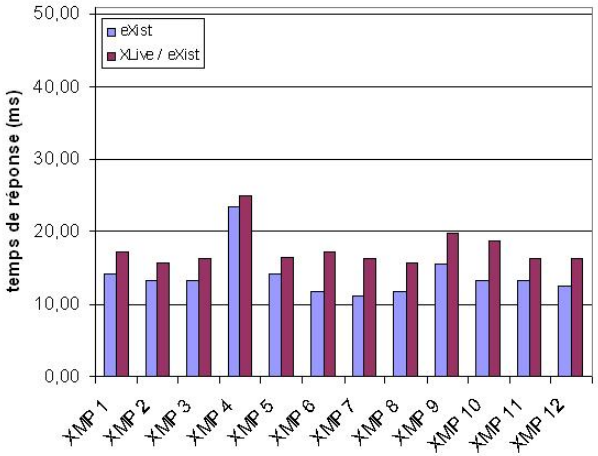
retour

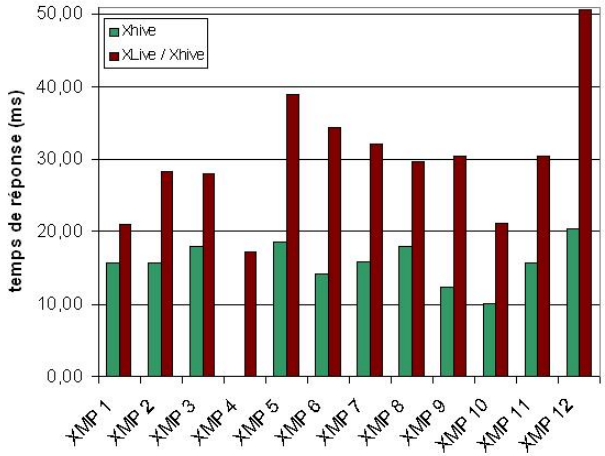
Pourquoi Xhive donne de tels résultats ?

Surcharge de traitement pour XLive :

- Données XML structurées différemment ;
- Restructuration en flues SAX dans XLive ;

$\Rightarrow$ Surcoût de traitement.



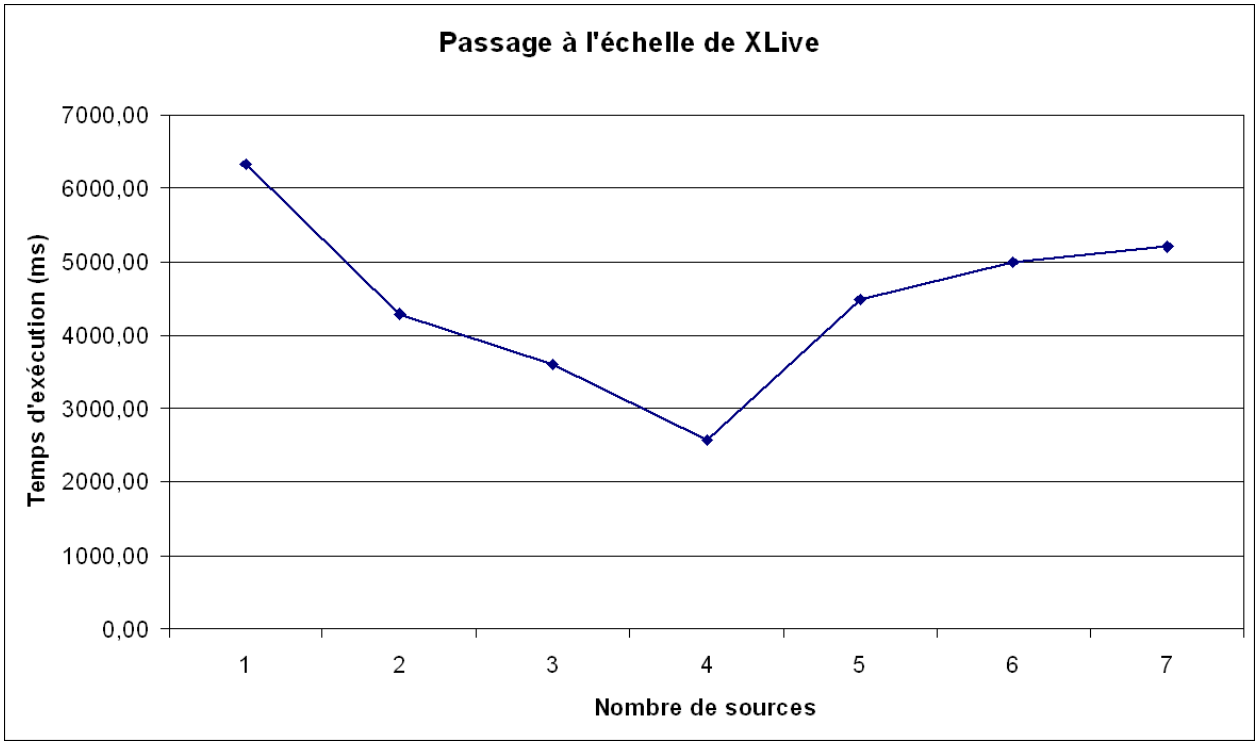


retour

Est-ce que le médiateur XLive passe à l'échelle ?

- Répartition des traitements (1 à 4 sources) ;
- Coût de communication avec de nombreuses sources (5 à 7 sources) ;

$\Rightarrow$ Augmentation tout au plus linéaire des temps de réponse ;



retour

Carey et al. 90 Carey M.J., DeWitt D.J., Graefe G., Haight D.M., Richardson J.E., Schuh D.T., Shekita E.J., Vandenberg S.L. "*The EXODUS Extensible DBMS Project : An Overview*", book : Readings on Object-Oriented Database System, Publisher : Morgan Kaufmann, San Mateo, CA, editeur : D. Maier and S. Zdonik, 1990

Graefe et McKenna 93 Graefe G., McKenna W.J. "*The Volcano Optimizer Generator : Extensibility and Efficient Search*", ICDE, 1993

Mitchell et al. 93 Mitchell G., Dayal U., Zdonik S.B. "*Control of an Extensible Query Optimizer : A Planning-Based Approach*", VLDB, 1993

Kabra et DeWitt 99 Kabra N., DeWitt D.J. "*OPT++ : an object-oriented implementation for extensible database query optimization*", VLDB, 1999

Amer-Yahia 01 Amer-Yahia S., Cho S., Lakshmanan L.V.S., Srivastava D., "*Minimization of Tree Pattern Queries*", SIGMOD Conference, 2001

Chen et al. 03 Chen Z., Jagadish H.V., Laksmanan L.V.S., Paparizos S. "*From Tree Patterns to Generalized Tree Patterns : On Efficient Evaluation of XQuery*", VLDB, 2003

Chen 04 Chen Z., "*From Tree Patterns to Generalized Tree Patterns : On Efficient Evaluation of XQuery*", University of British Columbia, Thesis, 2004

Dragan et Gardarin 05 Dragan F., Gardarin G. "*Benchmarking an XML Mediator*", ICEIS (1), 2005

BDA 04 Dang-Ngoc T.T., Gardarin G., Travers N. "*Tree Graph View : On Efficient Evaluation of XQuery in an XML Mediator*". Bases de Données Avancées (BDA national conference), 2004

BDA 05 T.-T. Dang-Ngoc, C. Jamard, and N. Travers. "*XLive : An XML Light Integration Virtual Engine*". Bases de Données avancées (BDA national conference), 2005.

IBIS 06 Travers N., Dang-Ngoc T.T., Liu T. "*TGV : an Efficient Model for XQuery Evaluation within an Interoperable System*", Interoperability in Business Information Systems (IBIS international journal), 2006

TechReport 06a Travers N., Dang-Ngoc T.T. "*Untyped-XQuery canonization*", Technical Report, PRiSM Laboratory, 2006

TechReport 06b Travers N., Dang-Ngoc T.T. "*The Tree Graph View Model*", Technical Report, PRiSM Laboratory, 2006

DASFAA 07 Travers N., Dang-Ngoc T.T., LiuT. "*TGV : a Tree Graph View for Modelling untyped XQuery*". Database Systems for Advanced Applications (DASFAA international conference), 2007