XML データベース XRel の実装とその評価

藤井眞吾[†], 天笠俊之[†], 吉川正俊[†,‡], 植村俊亮[†]

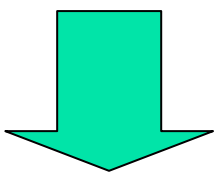
† 奈良先端科学技術大学院大学

‡ 国立情報学研究所ソフトウェア研究系

2002/03/06

研究の背景

- XML：データ・文書交換の標準形式として登場
- 今後 XML 文書が増大し、データベースに蓄積されることが予想
- 関係データベース (RDB) を用いての実現が有利



- RDB を用いた XML 文書の格納・検索手法 XRel の実装とその評価

XML およびその関連技術
XML 文書の例

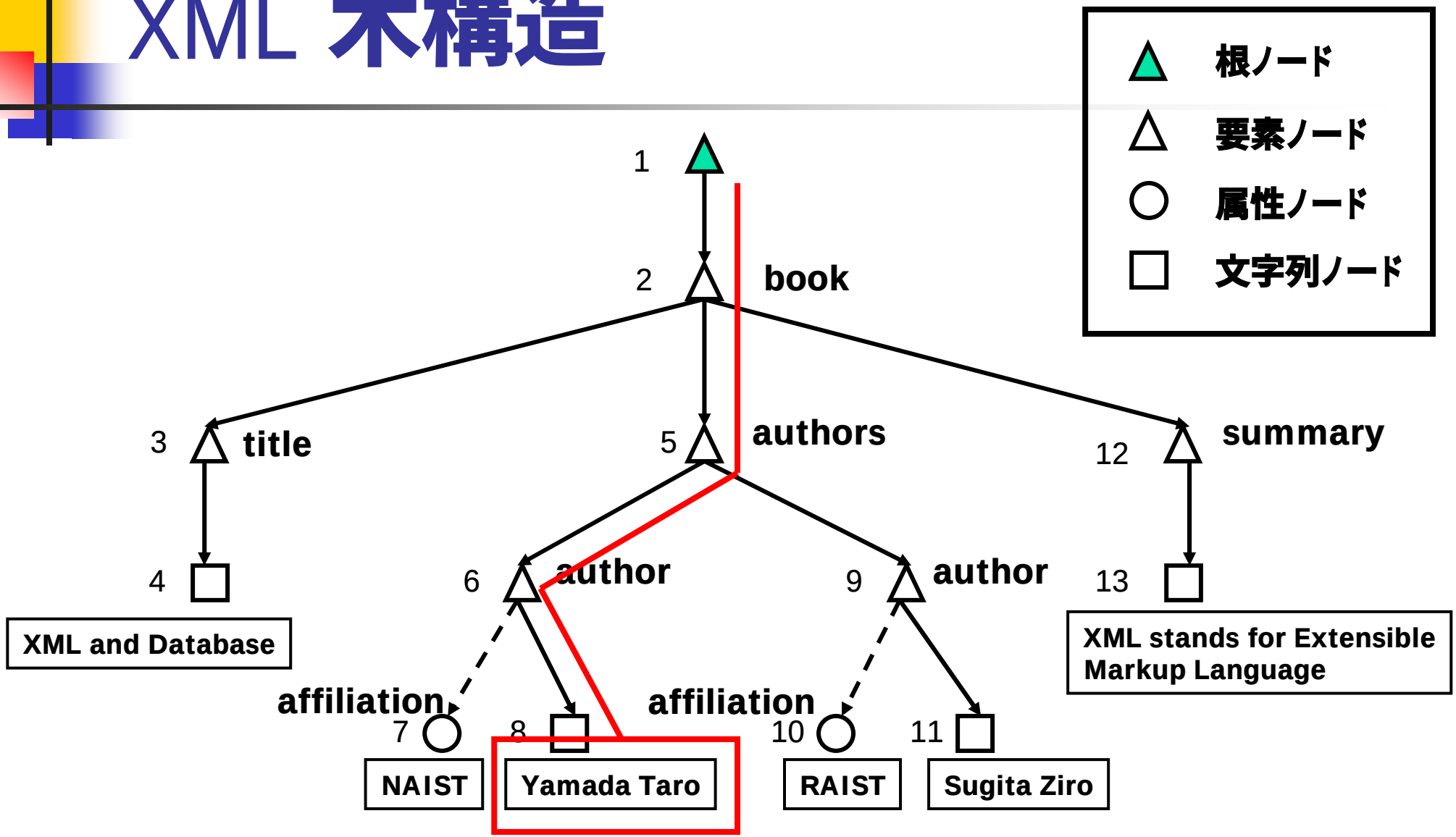
タイトル：XML and Database

著者：(NAIST) Yamada Taro, (RAIST) Sugita Ziro

要約：XML stands for Extensible Markup Language

```
<book>
    <title>XML and Database</title>
    <authors>
        <author affiliation="NAIST">Yamada Taro</author>
        <author affiliation="RAIST">Sugita Ziro</author>
    </authors>
    <summary>XML stands for Extensible Markup Language</summary>
</book>
```

XML およびその関連技術
XML 木構造



XML およびその関連技術
XPath 1.0

- **経路表現による** XML **文書の部分を特定**
  - /book/title
- **述語による文書部分の特定も可能**
  - /book[authors/author=‘Yamada Taro’]/title
- XQuery, XPointer **等の** XML **関連技術**

XML **データベースは** XPath **を効率よく扱えることが望ましい**

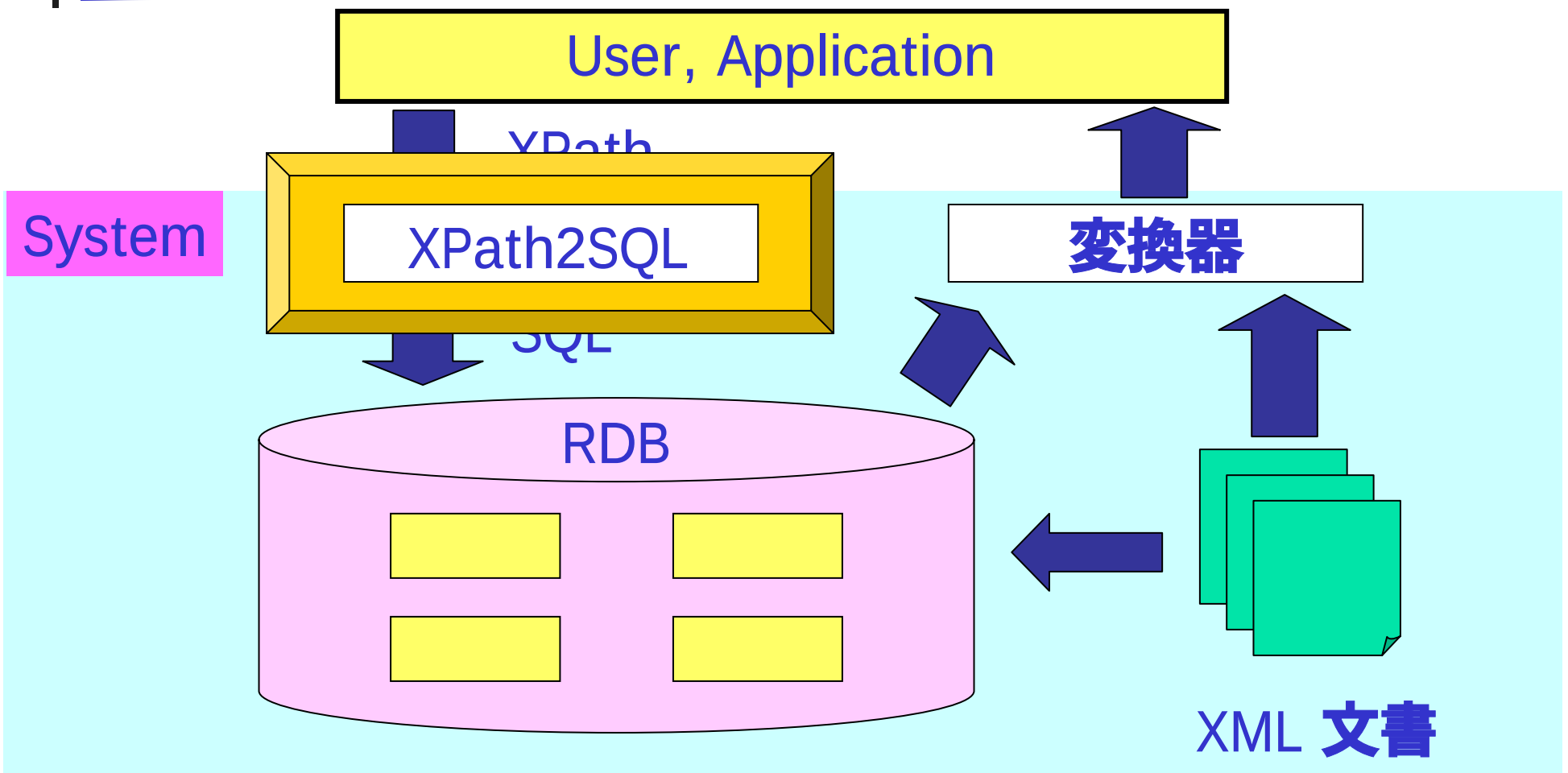
XRel (1/3)
XRel の概要

- XRel：関係データベースを用いた XML 文書の格納・検索手法
  - XML 文書の要素、属性、文字列に関する情報を「経路表現」と「先頭文字からのバイト数」の組で表現
  - XPath を効率よく扱うことが可能
  - 関係スキーマは文書型定義と独立している（汎用性）
  - 既存の RDBMS を拡張する必要がない

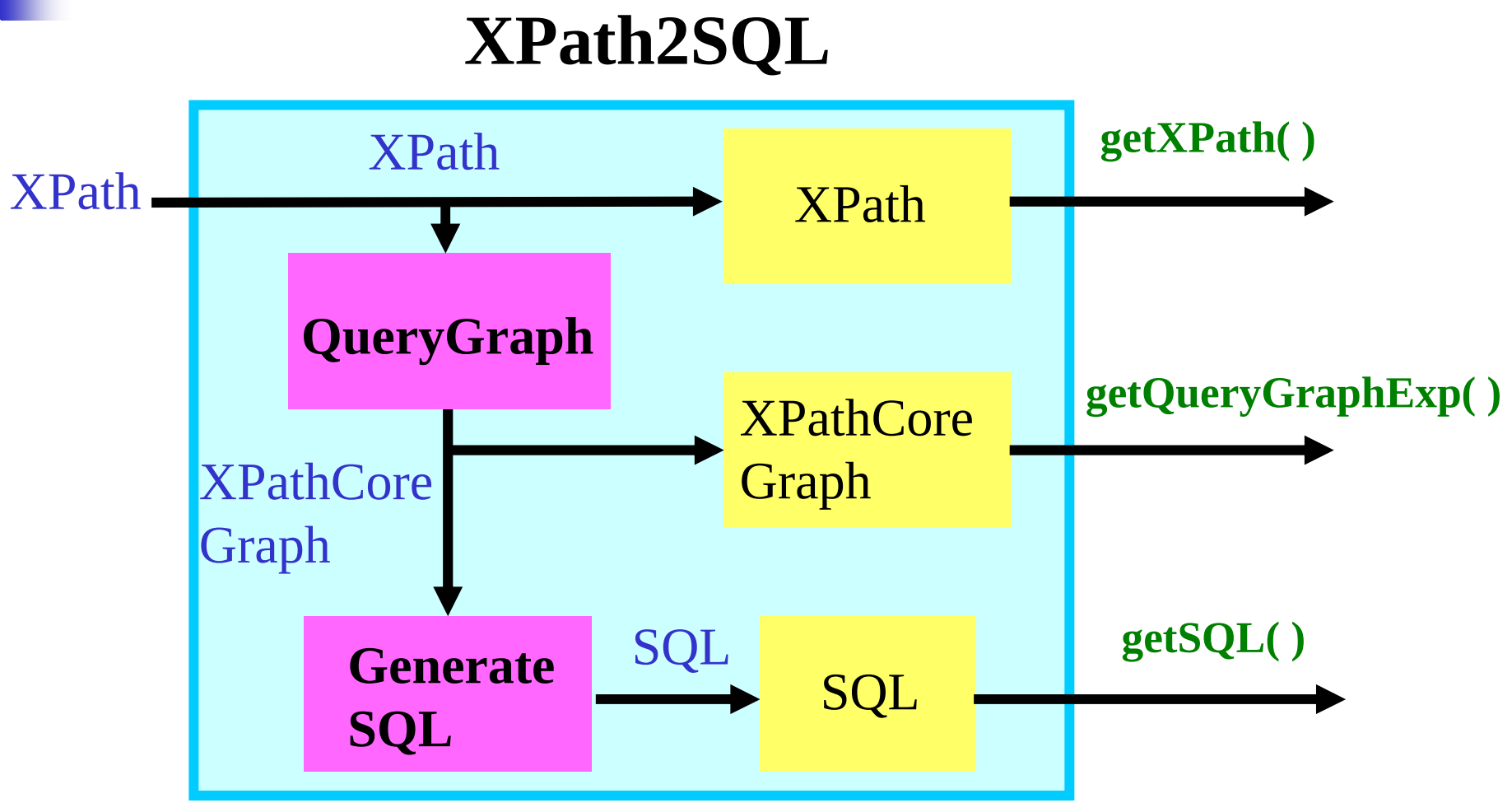
XRel (2/3)
格納

- Element
  (**文書**ID, **経路**ID, **始め**, **終わり**)
  (1, 1, 0, 240)
  (1, 2, 9, 39)
  (1, 3, 43, 168)
  (1, 4, 56, 103)
  (1, 4, 108, 155)
  (1, 6, 172, 231)
- Attribute
  (**文書**ID, **経路**ID, **始め**, **終わり**, **値**)
  (1, 5, 57, 57, NAIST)
  (1, 5, 109, 109, RAIST)
- Text
  (**文書**ID, **経路**ID, **始め**, **終わり**, **値**)
  (1, 2, 16, 31, XML and Database)
  (1, 4, 84, 94, Yamada Taro)
  (1, 4, 136, 146, Sugita Ziro)
  (1, 6, 181, 221, XML stands for …)
- Path
  (**経路**ID, **経路表現**)
  (1, #/book)
  (2, #/book#/title)
  (3, #/book#/authors)
  (4, #/book#/authors#/author)
  (5, #/book#/authors#/author#/@affiliation)
  (6, #/book#/summary)

XRel (3/3)
検索システム

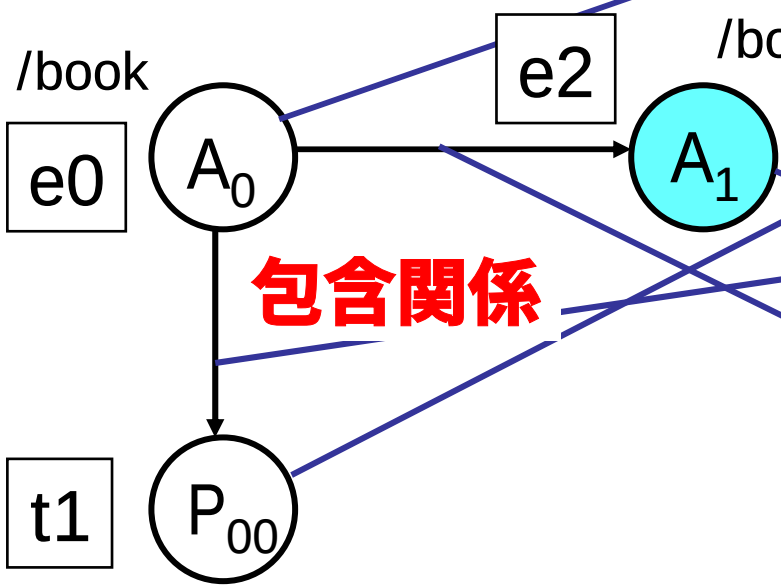


XRel の実装および応用例 (1/3)
XPath 式から SQL 問合せ式への変換モジュール(1/2)



XRel の実装および応用例 (2/3)
XPath 式から SQL 問合せ式への変換モジュール(2/2)

/book[authors/author=‘Yamada Taro’]/title




```
SELECT e2.docID, e2.start e2.end
FROM Element e0, Path p0, Text t1,
     Path p1, Element e2, Path p2
WHERE p0.pathexp LIKE ‘#/book’
AND e0.pathID = p0.pathID
AND P1.pathexp LIKE
     ‘#/book#/authors#/author’
AND t1.pathID = p1.pathID
AND t1.value LIKE ‘Yamada Taro’
AND e0.docID = t1.docID
AND e0.start < t1.start
AND e0.end > t1.end
AND p2.pathexp LIKE ‘#/book#/title’
AND e2.pathid = p2.pathid
AND e0.docID = e2.docID
AND e0.start < e2.start
AND e0.end > e2.end
ORDER BY e2.docID, e2.start, e2.end
```

文書 ID

開始位置

終了位置

XRel - Microsoft Internet Explorer

ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)

戻る 進む 中止 更新 ホーム 検索 お気に入り メディア 履歴 メール 印刷 編集

アドレス(D) http://db-www.aist-nara.ac.jp/~shingo-f/xrel/ 移動

XML 文書検索（XRel）

検索対象の XML 文書

- シェイクスピアの戯曲

問合せ例

TITLE 一覧を出力せよ

```
//TITLE
```

ADAM の台詞だけを出力せよ

```
/PLAY/ACT/SCENE/SPEECH
[SPEAKER = 'ADAM']
```

ADAM が現れる SCENE の TITLEを出力せよ

```
/PLAY/ACT/SCENE
[SPEECH/SPEAKER =
'ADAM']/TITLE
```

SQL に変換する XPath 式を入力してください。

XPath :

変換 クリア

システム管理用

掃除 : ゴミファイルの削除

XPath : /PLAY/ACT/SCENE[SPEECH/SPEAKER = 'ADAM']/TITLE
Result Number : 4 (Full Version (37 documents))

---

[1] : (3) xmlfiles/as_you.xml, 1927, 1978
<TITLE>SCENE I. Orchard of Oliver¥'s house.</TITLE>

[2] : (3) xmlfiles/as_you.xml, 52096, 52145
<TITLE>SCENE III. Before OLIVER¥'S house.</TITLE>

[3] : (3) xmlfiles/as_you.xml, 68347, 68383
<TITLE>SCENE VI. The forest.</TITLE>

[4] : (3) xmlfiles/as_you.xml, 69687, 69724
<TITLE>SCENE VII. The forest.</TITLE>

ページが表示されました イントラネット

Xmark
XML データベースの性能評価指標

- Xmark：XML Benchmark Project
  - **規模変更可能な** (scalable) **文書データの生成**
    - **ハードウェアや** OS **等の環境によらず、同一の** XML **文書が生成**
  - XQuery **による問合せ集合の提供**
    - XML **データベースとしての検索能力を評価するために** 20 **の問合せを用意**
    - **今回の実験では** 20 **のうち** 15 **の問合せを行った**

XRel の性能評価 (1/6)
実験環境

- **実験環境**
  - CPU：Pentium 4 (1.8 GHz)
  - **メインメモリ**：1 GB
  - OS：MIRACLE LINUX Version 2.0
  - **データベース**：Oracle 9i
  - **文書サイズ**：**約** 1.2 M**バイト** (1161652 **バイト**)
- **比較対象**：XML Query Engine (XQEngine)
  - Java **により記述**
  - XPath, XQL, XQuery **による問合せが可能**
  - Xmark **で用意された** 20 **の問合せのうち** 6 **つを実行可能**

XRel の性能評価 (2/6)
実験結果

問合せ番号	XRel (ms)	XQEngine (ms)	(XQEngine / XRel)
1	6.8	48.5	7.13
2	73.2	3509.3	47.9
3	599.7	-	-
4	56.0	-	-
6	8.8	2761.9	314
7	33.2	-	-
8	257.7	-	-
9	4383.0	-	-
13	18.2	129.9	7.14
14	245.8	-	-
15	14.6	214.6	14.7
16	95.0	226.8	2.39
17	256.1	-	-
18	13.9	-	-
19	2807.4	-	-

XRel の性能評価 (3/6)
XRel 単独の考察 (1/3)

- **得意な問合せ**
  - **指定する経路数が少ない**：Q2, 6, 15, 18
  - **条件に文字列照合を有する**：Q1, 4, (14)
  - **条件に順序節を有する**：Q2, (3)
  - **検索結果の表示コストが小さい**：Q6, 7
  - **結果の表示順序が拘束されていない**
- **不得意な問合せ**
  - **指定する経路数が多い**：Q3, (4), 8, 9, 14
  - **条件に文字列の比較を有する**：Q3, 8, 9
  - **指定する経路に「//」を有する**：Q(6, 7), 14, 19
  - **結果の表示順序が拘束される**

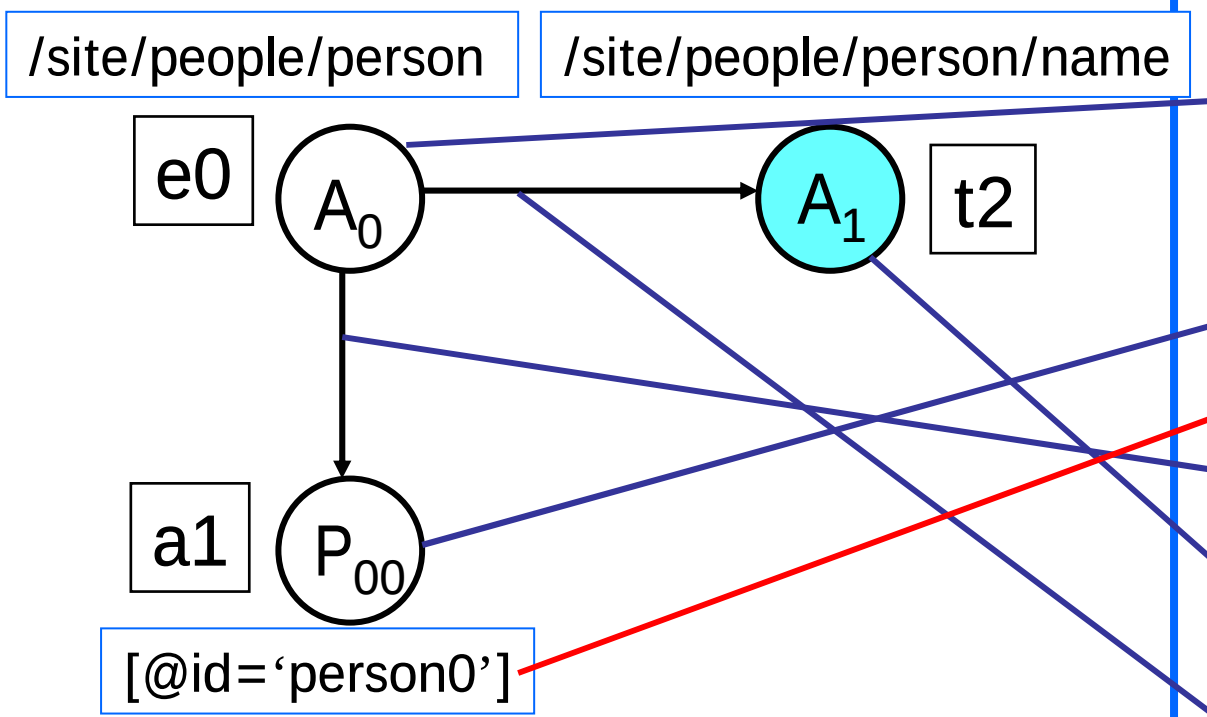
XRel の性能評価 (4/6)
XRel 単独の考察 (2/3)

- **得意な問合せ**：Q1
  - 「ID が ‘person0’ である人の名前を返す」

```
FOR $b IN document(“auction.xml”)/site/people/person/[@id=“person0”]
RETURN $b/name/text()
```

  - **条件に文字列照合を有する**

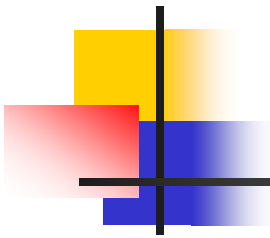
XRel の性能評価 (5/6)
XRel 単独の考察 (3/3)




```
SELECT t2.value
FROM Element e0, Path p0, Attribute a1,
     Path p1, Text t2, Path p2
WHERE p0.pathexp LIKE
     '#/site#/people#/person'
AND e0.pathID = p0.pathID
AND P1.pathexp LIKE
     '#/site#/people#/person#/@id'
AND a1.pathID = p1.pathID
AND a1.value LIKE 'person0'
AND e0.start < a1.start
AND e0.end > a1.end
AND p2.pathexp LIKE
     '#/site#/people#/person#/name'
AND t2.pathid = p2.pathid
AND e0.start < t2.start
AND e0.end > t2.end
ORDER BY t2.start;
```

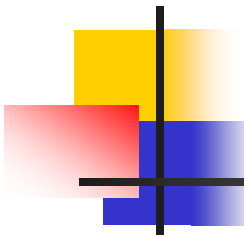
XRel の性能評価 (6/6)
XRel と XQEngine との比較

- Q2：47.9 倍
  - 順序節の検索に対する計算量の差 (XRel では順序に関する情報も格納)
- Q6：314 倍
  - 経路表現に「//」を有した式の検索能力の差 (XRel では文字列照合)
  - XRel 単独では不得意な問合せと思われたが XQEngine ではさらに不得意
- Q16：2.39 倍
  - XRel：結合される組を絞る条件がない



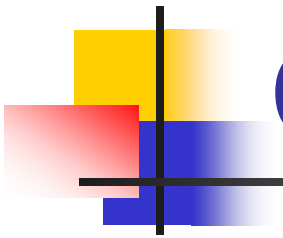
まとめ

- まとめ
  - XRel の移植性
    - 問合せ変換モジュールの実装
    - 変換モジュールを利用した応用システム
  - XRel の性能
    - 問合せには得意不得意がある
    - XML 検索システム XQEngine に対する高性能性



今後の課題

- 今後の課題
  - XRel では XML 文書の要素や属性に関する情報を「ファイルの先頭からのバイト数」で格納
    - 文書内容の更新により「先頭からのバイト数」がずれるため、データを関係データベースへ格納しなおす必要
    - 文書内容の更新があった場合でも、格納しなおす箇所を最小限にするようなシステムの実現
  - XQuery 式を SQL 式に機械的に変換する手法の考案



Q2

- **得意な問合せ**：Q2
  - **「すべての開催中の競売** (open_auction) **の初期の増加** (increase) **を返す」**

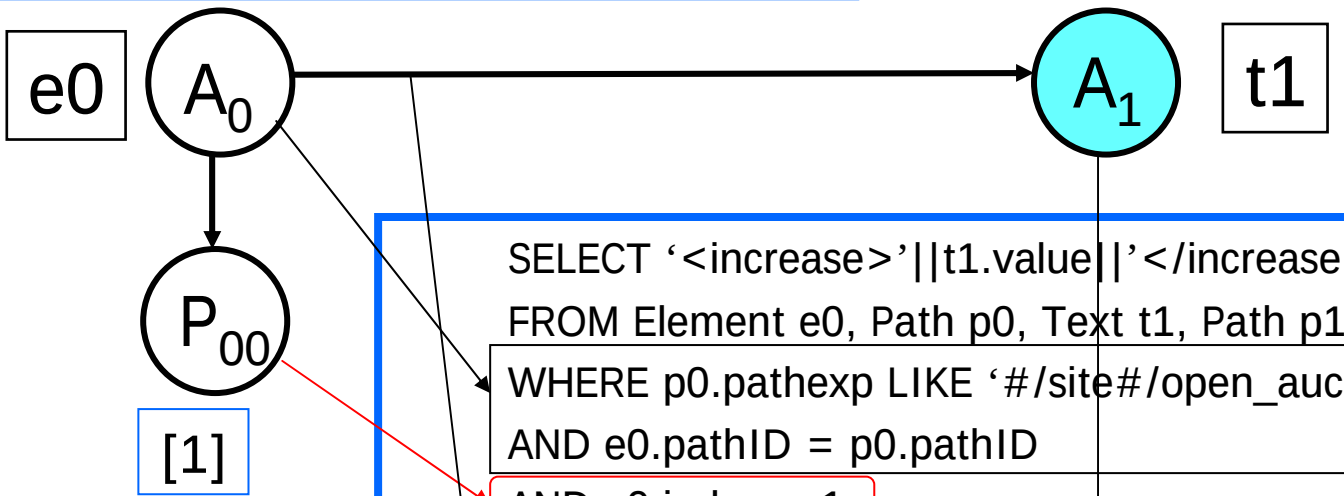
```
FOR $b IN document(“auction.xml”)/site/open_auctions/open_auction
RETURN <increase> $b/bidder[1]/increase/text() </increase>
```

  - **指定する経路数が少ない**
  - **順序節を有する**

Q2

/site/open_auctions/open_auction/bidder

/site/open_auctions/open_auction/bidder/increase



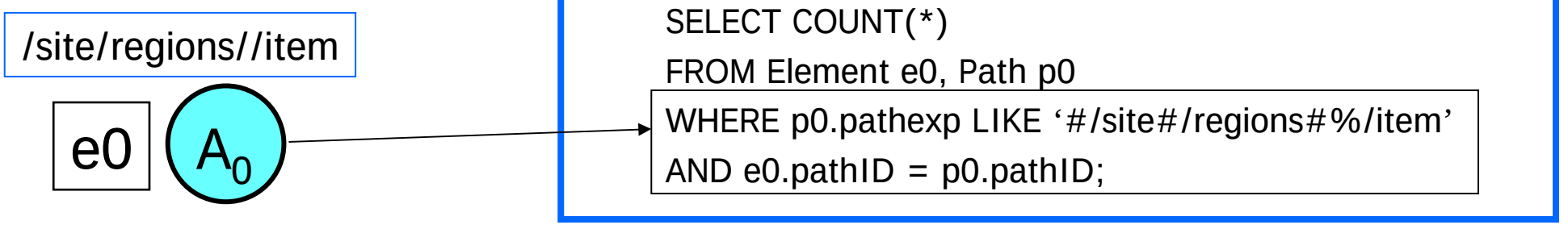

```
SELECT '<increase>'||t1.value||'</increase>
FROM Element e0, Path p0, Text t1, Path p1
WHERE p0.pathexp LIKE '#/site#/open_auctions#/open_auction/bidder'
AND e0.pathID = p0.pathID
AND e0.index = 1
AND P1.pathexp LIKE '#/site#/open_auctions#/open_auction/bidder#/increase'
AND t1.pathID = p1.pathID
AND e0.docID = t1.docID
AND e0.start < t1.start
AND e0.end > t1.end
ORDER BY t1.start;
```

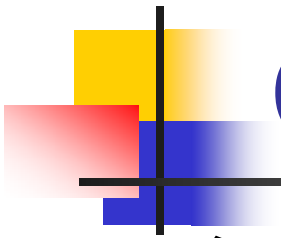
Q6

- **得意な問合せ**：Q6
  - **「すべての大陸どれだけの物品** (item) **があるか」**

```
FOR $b IN document("auction.xml")/site/regions
RETURN COUNT ($b//item)
```

  - **指定する経路数が少ない**
  - **検索結果の表示コストが小さい**





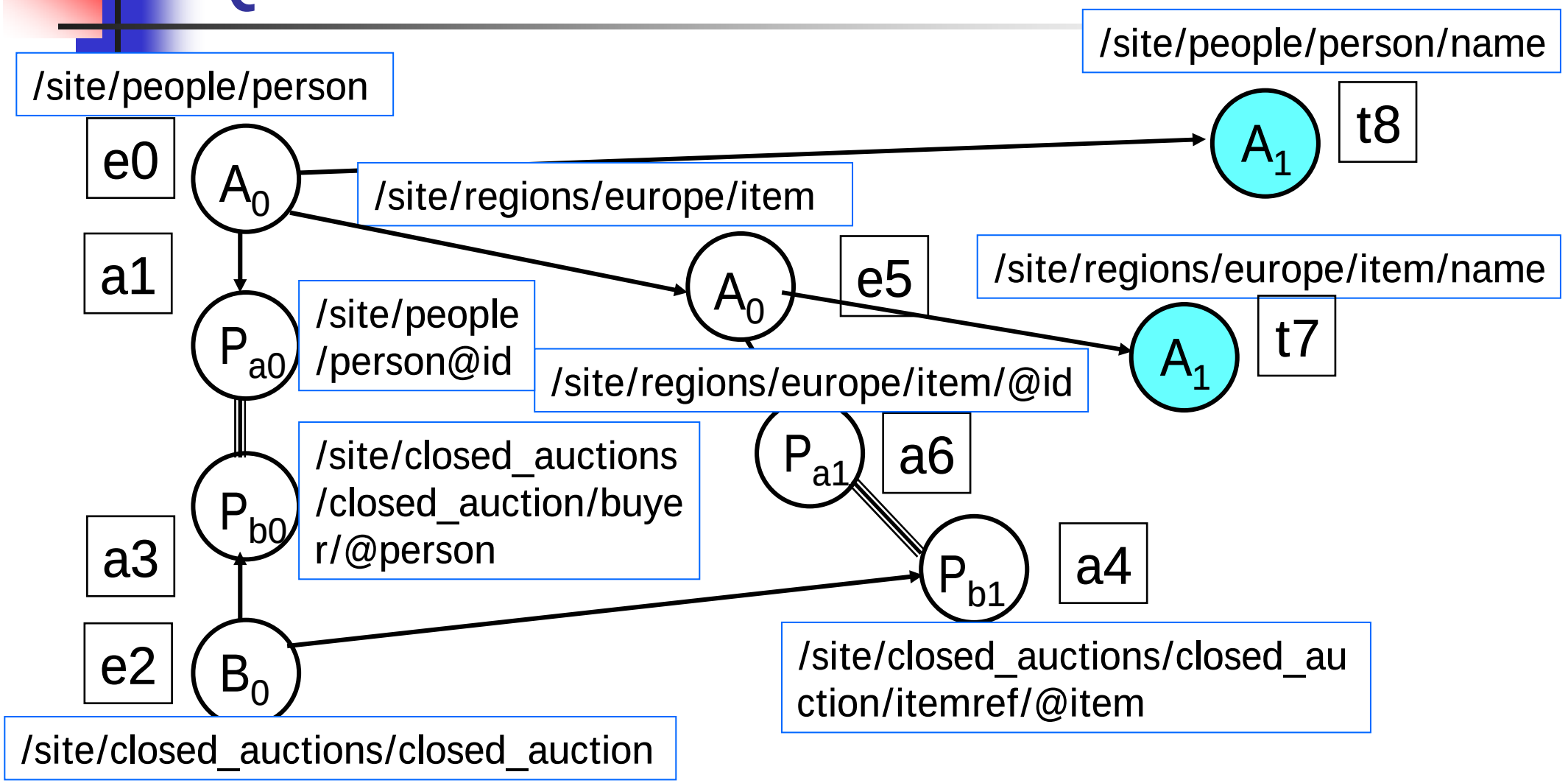
Q9

- **不得意な問合せ**：Q9
  - **「人の名前とヨーロッパで買った物品のリスト」**

```
FOR $p IN document(“auction.xml”)/site/people/person
    LET $a := FOR $t IN document(“auction.xml”)/site/closed_auctions/closed_auction
            LET $n := FOR $t IN document(“auction.xml”)/site/regions/europe/item
            WHERE $t/itemref/@item = $t2/@id
            RETURN $t2
    WHERE $p/@id = $t/buyer/@person
    RETURN <item> $n/name/text() </item>
RETURN <person name=$p/name/text()> $a </person>
```

  - **指定する経路数が多い**
  - **文字列の比較を有する**

Q9


/site/people/person
/site/people/person/name
e0
A0
A1
t8
/site/regions/europe/item
a1
A0
e5
/site/regions/europe/item/name
Pa0
/site/people
/person@id
A1
t7
/site/regions/europe/item/@id
Pa1
a6
/site/closed_auctions
/closed_auction/buye
r/@person
Pb0
a3
Pb1
a4
e2
B0
/site/closed_auctions/closed_au
ction/itemref/@item
/site/closed_auctions/closed_auction

XRel の性能評価
XRel 単独の考察

- 得意な問合せ
  - 指定する経路数が少ない
    -> 表の結合 (join) 回数が少ない
  - 条件に文字列照合を有する、条件に順序節を有する
    -> 結合される表の組が絞られる
  - 検索結果の表示コストが小さい
  - 結果の表示順序が拘束されていない
    -> 整列 (sort) が不要
- 不得意な問合せ
  - 指定する経路数が多い
    -> 表の結合回数が多い
  - 条件に文字列の比較を有する
    -> 文字列同士の比較による表の結合
  - 指定する経路に「//」を有する
    -> wildcard (%) を用いた文字列照合と表の結合の組合せ
  - 結果の表示順序が拘束される
    -> 整列が必要