Семинар «Алгоритмы для Интернета»

Поиск в полуструктурированных данных

Докладчики:
Искандер Абсалямов
Николай Вяххи

16 ноября 2006
ауд. 100, ИТМО

План

0. О чём доклад
1. Постановка задачи
2. ElemRank
3. Обработка запросов

0. О чём доклад

- Полуструктурированные данные
- XML
- Что хотим
- HTML vs XML
- XRANK

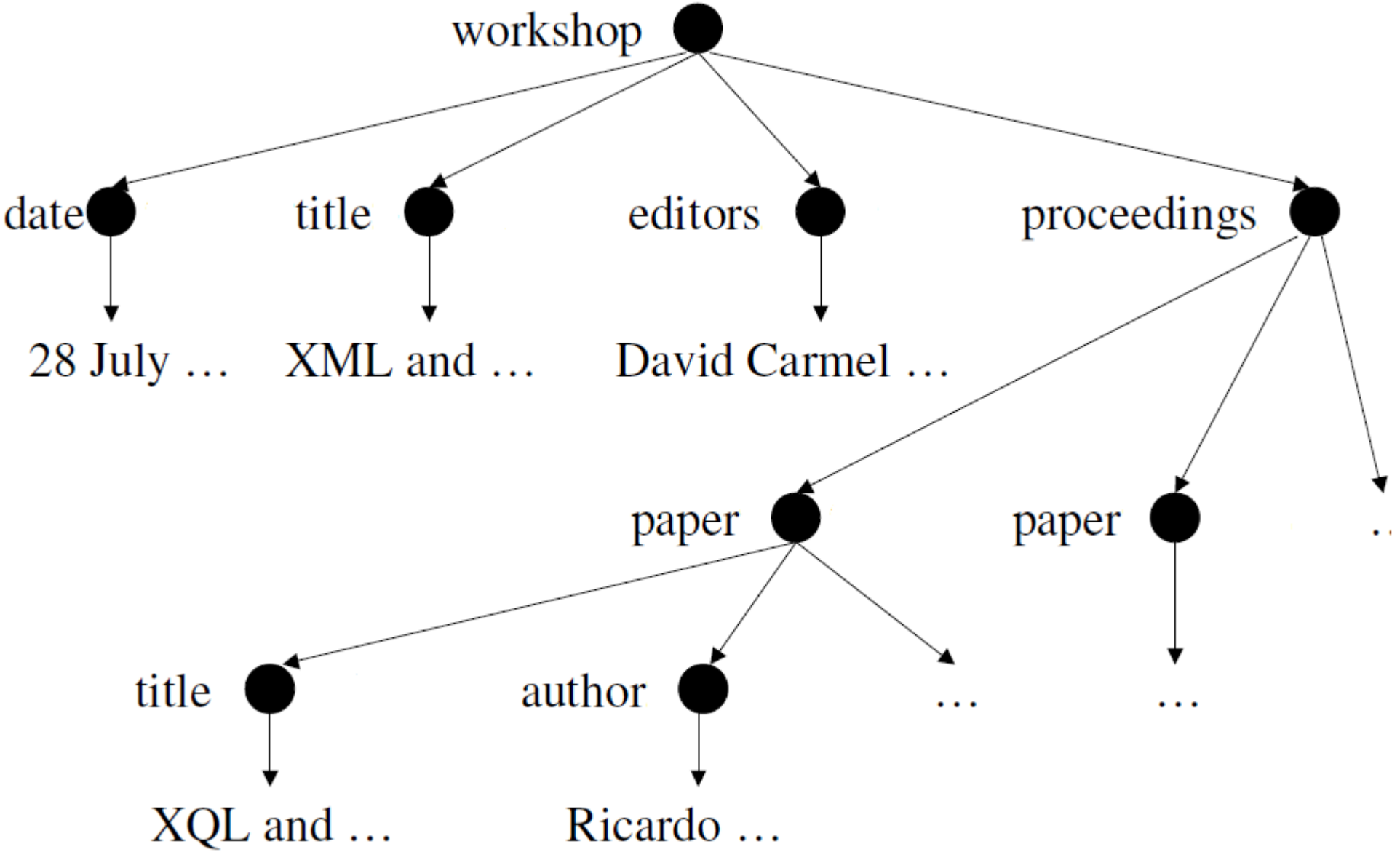
Полуструктурированные данные

данные без постоянной чётко определённой структуры

Полуструктурированные данные

данные со структурой, неизвестной пользователю

Полуструктурированные данные



XML (eXtensible Markup Language)

язык, удобный для представления полуструктурированных данных

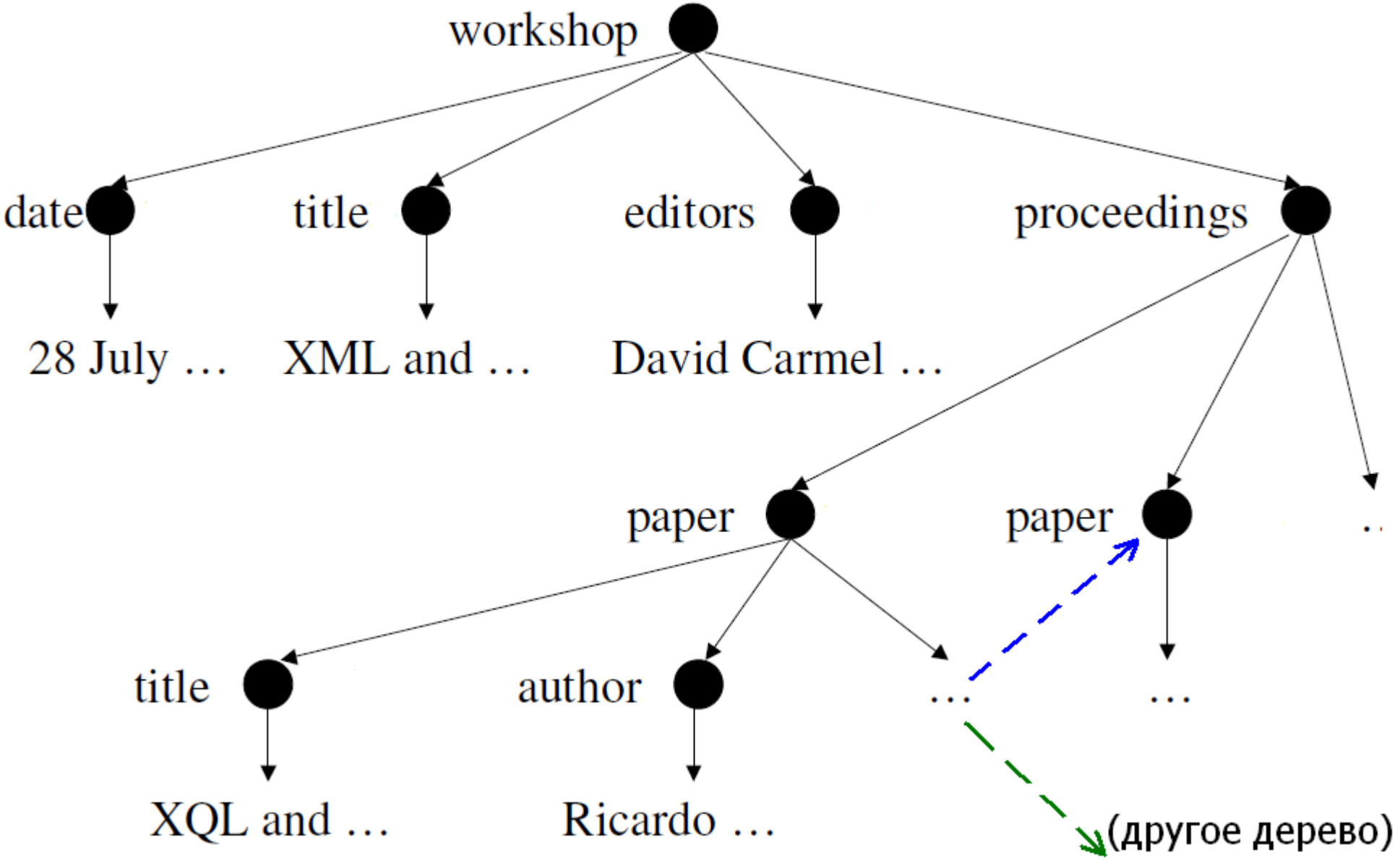
XML

```
<workshop date="28 July 2000">
    <title> XML and IR: A SIGIR 2000 Workshop </title>
    <editors> David Carmel, Yoelle Maarek, Aya Soffer </editors>
    <proceedings>
        <paper id="1">
            <title> XQL and Proximal Nodes </title>
            <author> Ricardo Baeza-Yates </author>
            <author> Gonzalo Navarro </author>
            <body>
                bla... bla... bla...
                bla... bla... bla...
                bla... bla... bla...
            </body>
        </paper>
        <paper id="2">
            <title> Querying XML in Xyleme </title>
            <body> bla... </body>
        </paper>
    </proceedings>
</workshop>
```

XML

```
<workshop date="28 July 2000">
    <title> XML and IR: A SIGIR 2000 Workshop </title>
    <editors> David Carmel, Yoelle Maarek, Aya Soffer </editors>
    <proceedings>
        <paper id="1">
            <title> XQL and Proximal Nodes </title>
            <author> Ricardo Baeza-Yates </author>
            <author> Gonzalo Navarro </author>
            <body>
                bla... bla... bla...
                <cite ref="2">Querying XML in Xyleme</cite>
                <cite xlink="../paper/xmlql/">A Query … </cite>
            </body>
        </paper>
        <paper id="2">
            <title> Querying XML in Xyleme </title>
            <body> bla... </body>
        </paper>
    </proceedings>
</workshop>
```

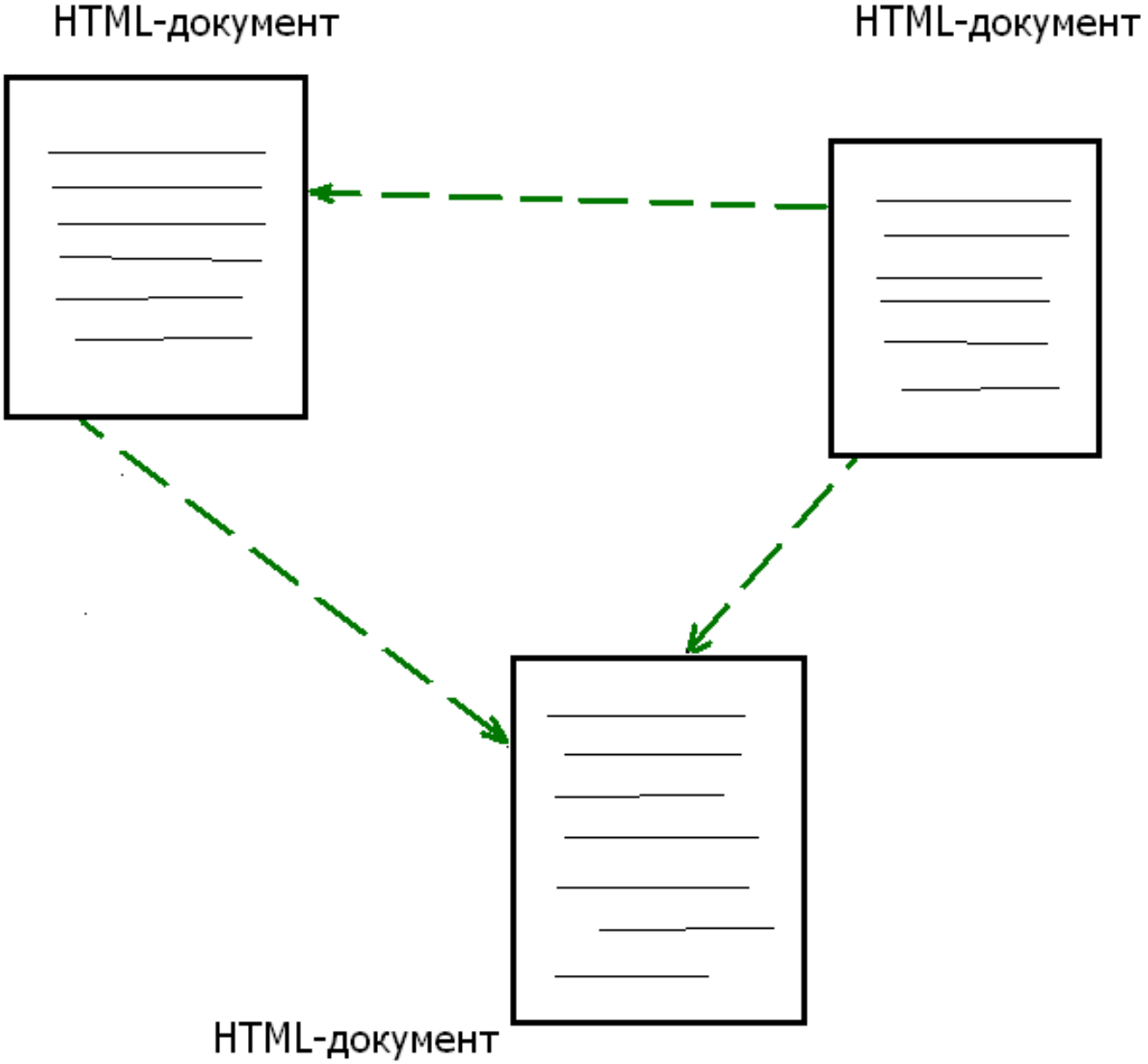
XML


workshop
date
title
editors
proceedings
28 July …
XML and …
David Carmel …
paper
paper
..
title
author
…
…
XQL and …
Ricardo …
(другое дерево)

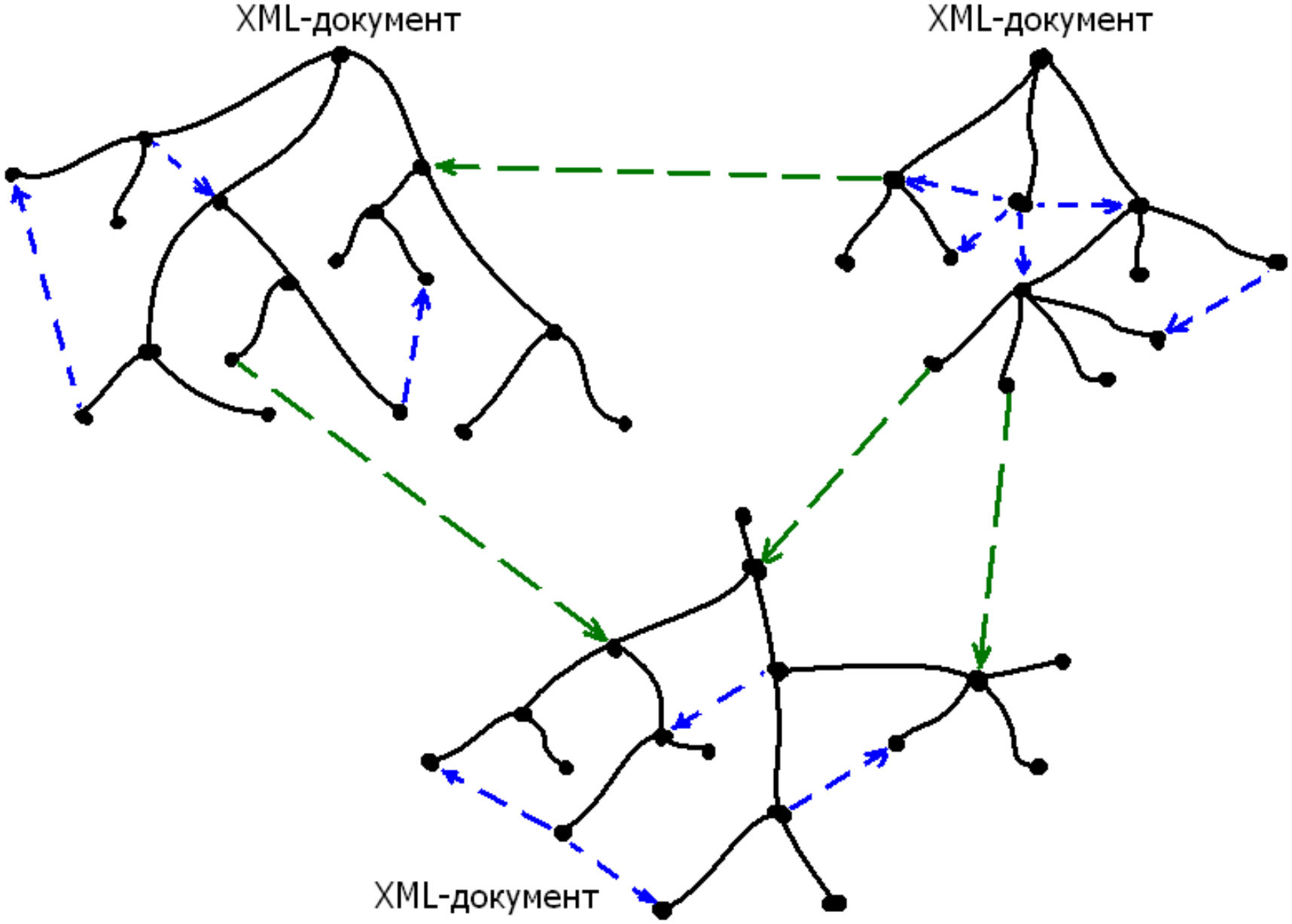
Что хотим

получить упорядоченные
по релевантности результаты
информационного поиска
в наборе XML-документов

Было с HTML:



Стало с XML:



HTML vs XML

- ищут целые документы
- ранжируют целые документы
- близость слов определяется положением в тексте

- ищут XML-элементы
- ранжируют XML-элементы
- близость слов определяется и положением в дереве

Трудности с XML

- Насколько глубокий XML-элемент возвращать?
- Как учесть структуру и ссылок, и самих XML-деревьев?
- Как считать близость слов запроса в разных XML-элементах?

Решение: XRANK

обобщение с HTML на XML

поиска на основе PageRank

План

0. О чём доклад
1. Постановка задачи
2. ElemRank
3. Обработка запросов

1. Постановка задачи

- Что имеем
- Что ищем
- Как ранжируем
- Архитектура XRANK

Что имеем

Набор XML-документов – граф

$$G = (V, CE, HE)$$

V – вершины: XML-элементы

CE – дуги вложенности

HE – дуги ссылок

Что ищем

R_0 – множество XML-элементов, содержащих все слова запроса

Что ищем

R_1 – множество XML-элементов, таких что ≥1 потомков содержат все слова запроса, ≥1 потомков содержат только часть слов

Как ранжируем

Желаемые свойства:

а) *специфичность результатов*

б) близость ключевых слов

в) учёт ссылок

Как ранжируем

Имеем:

$Q = (k_1, \ldots, k_n)$ – запрос

$R = R_1 \cup R_2$ – результаты поиска

v_1 in R, $ElemRank(v)$ (~PageRank)

Как ранжируем

k_i in Q, v_1.contains(k_i), тогда есть (v_1,v_2), (v_2,v_3), …, (v_{n-1}, v_n) из *CE*: v_n.contains(k_i)

$r(v_1, k_i) := ElemRank(v_n) * q^{(n-1)}$

Как ранжируем

k_i in Q, v_1.contains(k_i) m раз

тогда $\check{r}(v_1, k_i) := f(r_1, ..., r_m)$,

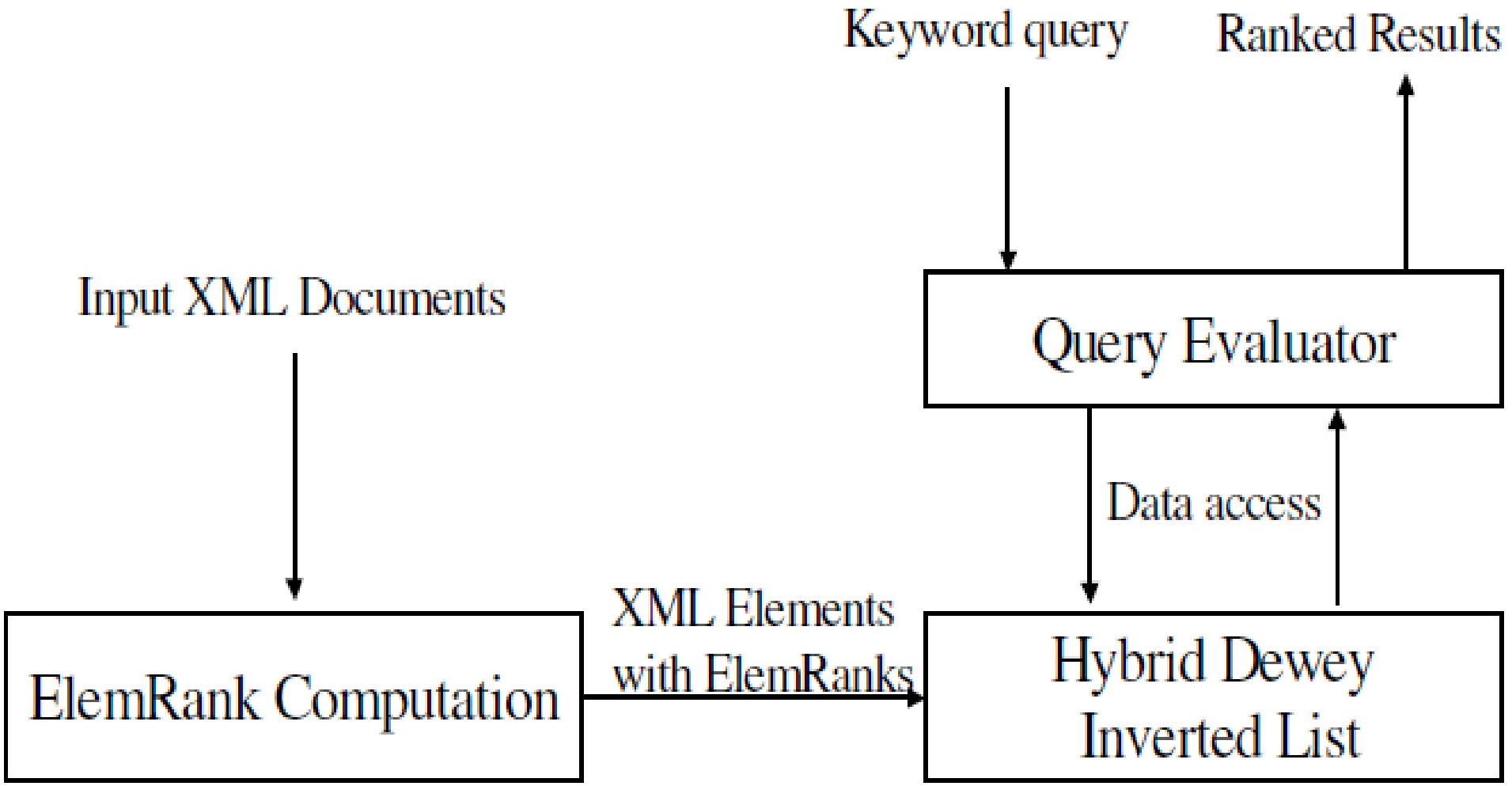
где $f = max$ или $\sum$

Как ранжируем

$$R(v_1, Q) = \left(\sum_{1 \le i \le n} \hat{r}(v_1, k_i) \right) \times p(v_1, k_1, k_2, \ldots, k_n)$$

p = 1 / размер_окна (мера близости слов)

Архитектура XRANK



План

0. О чём доклад
1. Постановка задачи
2. ElemRank
3. Обработка запросов

2. ElemRank

ElemRank(*v*) –
мера важности XML-элемента, вычисляемая на основе структуры гиперссылок

Вычисление Page*Rank*

Как оно для HTML (PageRank):

$$p(v) = \frac{1-d}{N_d} + d \times \sum_{(u,v) \in HE} \frac{p(u)}{N_h(u)}$$

$d = 0.85$
N_d – общее число документов
$N_h(u)$ – число ссылок из u

Вычисление ElemRank

Переход от HTML к XML:

a) дуги из *HE* и *CE* разных весов

b) дуги из *CE* двунаправлены

c) ранг элемента зависит
от рангов детей

Вычисление ElemRank

$$e(v) = \frac{1 - d_1 - d_2 - d_3}{N_d \times N_{de}(v)} + d_1 \sum_{(u,v) \in HE} \frac{e(u)}{N_h(u)}$$

$$+ d_2 \sum_{(u,v) \in CE} \frac{e(u)}{N_c(u)} + d_3 \sum_{(u,v) \in CE^{-1}} e(u)$$

d_1 – вероятность, что пришли в v по ссылке
$N_h(u)$ – число ссылок из элемента u

Вычисление ElemRank

$$e(v) = \frac{1 - d_1 - d_2 - d_3}{N_d \times N_{de}(v)} + d_1 \sum_{(u,v) \in HE} \frac{e(u)}{N_h(u)}$$

$$+ d_2 \sum_{(u,v) \in CE} \frac{e(u)}{N_c(u)} + d_3 \sum_{(u,v) \in CE^{-1}} e(u)$$

d_2 – вероятность, что пришли в v из родителя
$N_c(u)$ – число детей у элемента u

Вычисление ElemRank

$$e(v) = \frac{1 - d_1 - d_2 - d_3}{N_d \times N_{de}(v)} + d_1 \sum_{(u,v) \in HE} \frac{e(u)}{N_h(u)}$$

$$+ d_2 \sum_{(u,v) \in CE} \frac{e(u)}{N_c(u)} + d_3 \sum_{(u,v) \in CE^{-1}} e(u)$$

d_3 – вероятность, что пришли в v из потомка
N_{de} – число предков v

План

0. О чём доклад
1. Постановка задачи
2. ElemRank
3. Обработка запросов

3 Обработка запросов

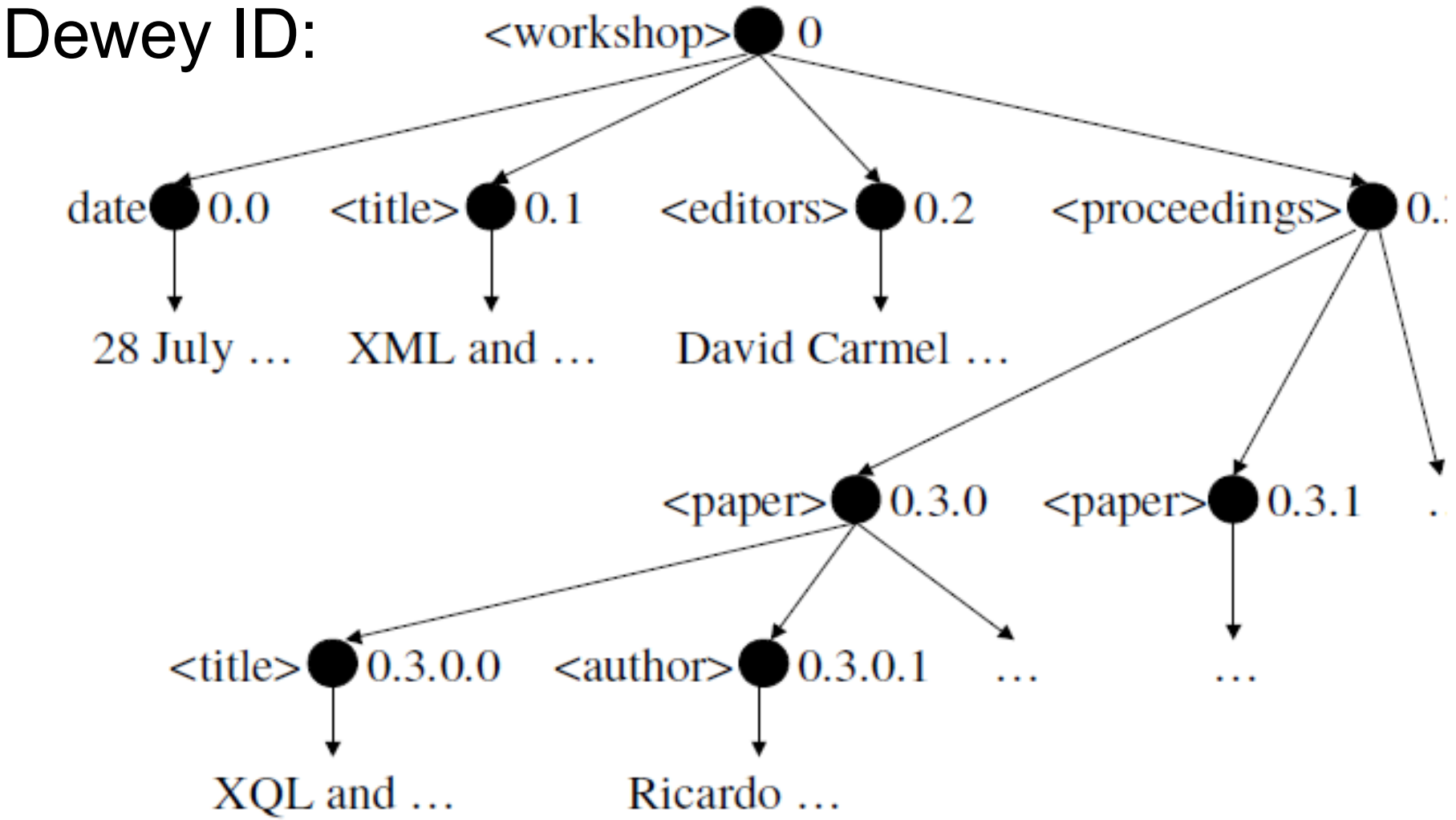
3.1 Наивный подход
3.2 Dewey Inverted List
3.3 Ranked Dewey Inverted List
3.4 Hybrid Dewey Inverted List

3.1 Наивная обработка запросов

Обращаться с каждым XML-элементом как с отдельным документом:

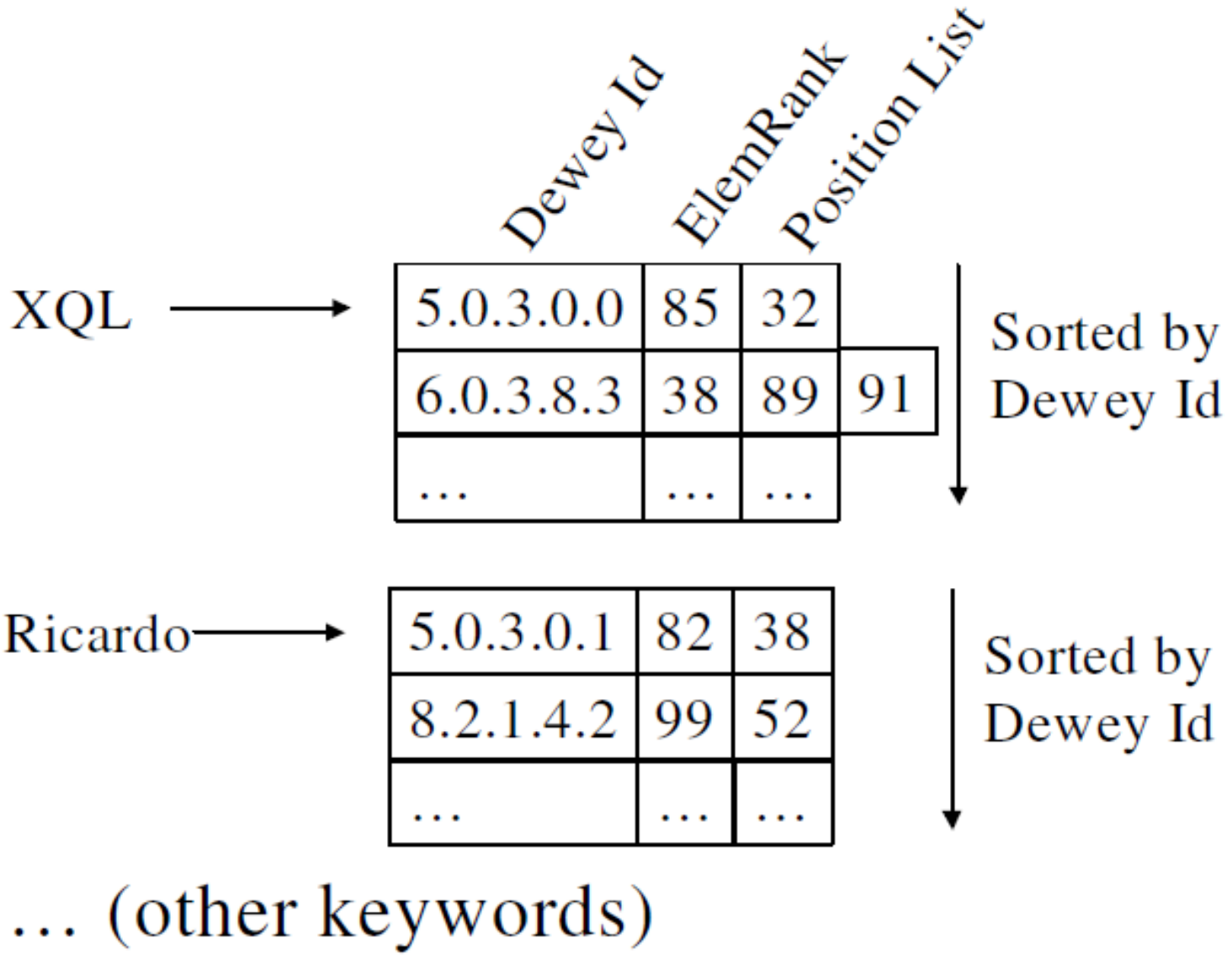
1) надо много памяти
2) результаты поиска будут повторяться
3) не учитывается специфичность

3.2 Dewey Inverted List (DIL)



3.2 Dewey Inverted List (DIL)

DIT:



3.2 Dewey Inverted List (DIL)

Основная идея:

сливать списки, одновременно вычисляя самый длинный общий префикс Dewey ID в разных списках

(смотреть на доску →)

3.2 Dewey Inverted List (DIT)

Недостатки:

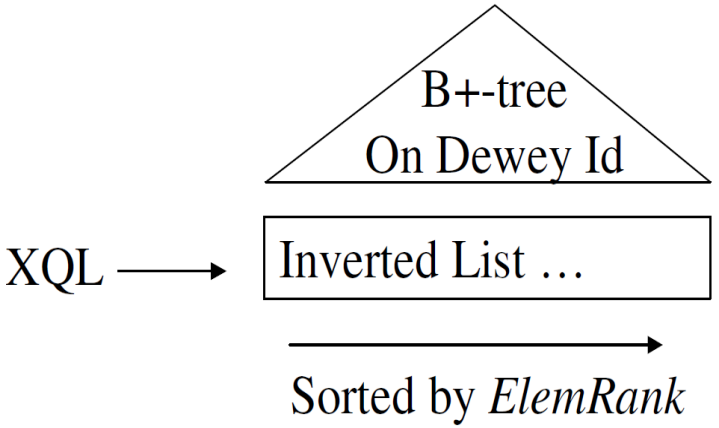
список ответов может оказаться *очень* длинным, а пользователю нужны только самые важные,

поэтому...

3.3 Ranked Dewey Inverted List

RDIL – DIL, отсортированный
не по Dewey ID, а по *ElemRank*

Взамен храним B+ дерево
по Dewey ID



(смотреть на доску →)

3.3 Ranked Dewey Inverted List

Недостатки:

если слова в запросе слабо коррелируют, возможно, придётся просмотреть все списки,

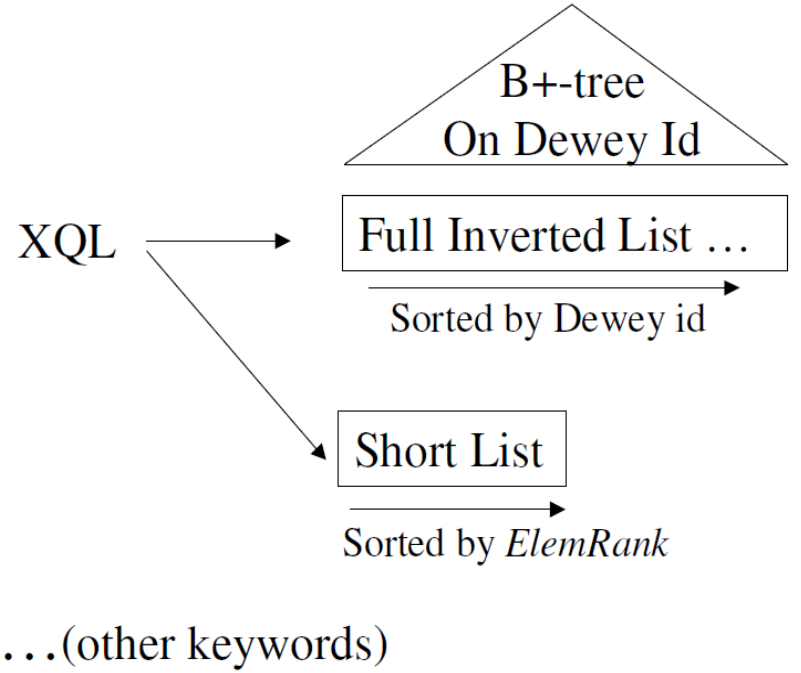
поэтому...

3.4 Hybrid Dewey Inverted List

Хранить:

+ полный DIL

+ начало RDIL



Начинать поиск с RDIL –
если медленно идёт,
переходить на DIL

4 Сопутствующая задача

Обновление
списков DIL и RDIL

Спасибо!

Источник: SIGMOD 2003

«XRANK: Ranked Keyword Search over XML Documents»

by Lin Guo, Feng Shao, Chavdar Botev, Jayavel Shanmugasundaram

Вопросы?