Pathfinder: XQuery — The Relational Way

Peter Boncz[1] Torsten Grust[4] Maurice van Keulen[2] Stefan Manegold[1] Jan Rittinger[1,3] Jens Teubner[4]

[1]CWI Amsterdam, The Netherlands [2]University of Twente, The Netherlands [3]University of Konstanz, Germany [4]Technische Universität München, Germany

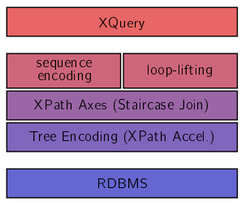
http://www.pathfinder-xquery.org/

Abstract

In previous work, we demonstrated that relational databases can serve as efficient back-ends for XML Query processing: The **XPath accelerator** ("*pre*/*post* order numbering") encodes XML documents in a schema-oblivious fashion. **Staircase join** encapsulates tree-specific knowledge for efficient XPath evaluation on *pre*/*post* encoded data. At VLDB 2004 we devised the **loop-lifting** compilation procedure that allows the **purely relational** evaluation of XQuery on off-the-shelf RDBMSs.

This demonstration assembles these techniques into a full-fledged XQuery implementation. The **Pathfinder compiler** translates incoming XQuery expressions into **purely relational** query plans. Though implementable on **any RDBMS**, these plans benefit from specific optimizations in our relational back-end **MonetDB**. Experiments confirm the **unsurpassed scalability** of our approach.

Relational XQuery Processing Stack



- Relational databases as back-ends for XML:
  - re-use existing technology: indexes, optimizers, ...
  - reach **unsurpassed scalability**,
  - ease **integration** with existing software.
- There is some tension between the relational processing model and XQuery:
  - **set-oriented** data model *vs.* **ordered sequences**,
  - **bulk processing** *vs.* explicit **iteration**.
- Pathfinder implements a fully relational XQuery **processing stack**:
  - a suitable **tree encoding** to store XML documents (*e.g.*, XPath accelerator),
  - efficient **XPath evaluation** by means of **staircase join**,
  - relational **encoding** for XQuery **sequences**,
  - bulk-oriented processing of iteration constructs in terms of **loop-lifting**.
- Pathfinder is designed as an XQuery-to-RDBMS **compiler**.

Source Language: XQuery Core

- Large subset of the W3C XQuery specification:
  - arbitrary expression nesting,
  - `id`/`idref` support.
- Pathfinder implements the **full axis feature**.
- **Static typing** at query compile time.
- Normalized XQuery Core representation for effective **query rewriting**.

atomic literals	document order (e_1 << e_2)
sequences (e_1, e_2)	node identity (e_1 is e_2)
variables ($\$v$)	arithmetics (+, −, ...)
`let` $\$v$:= e_1 `return` e_2	comparisons (`eq`, `lt`, ...)
`for` $\$v$ `at` $\$p$ `in` e_1 `return` e_2	Boolean operators (`and`, `or`, ...)
`if` e_1 `then` e_2 `else` e_3	`fn:doc`(e), `fn:root`(e)
`typeswitch` clauses	`fn:id`(e), `fn:idref`(e)
`element { }` e_1 `{ }` e_2	`fn:data`(e)
`text { }` e	`fs:distinct-doc-order`(e)
e_1 `order by` e_2,...,e_n	`fn:count`(e), `fn:sum`(e)
XPath (e/α::ν)	`fn:empty`(e)
user defined functions	`fn:position()`, `fn:last()`

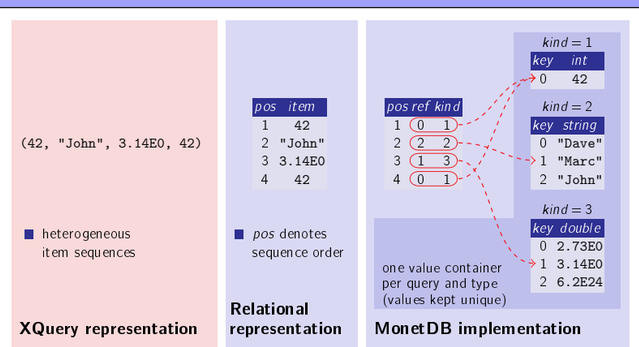
Target Language: Relational Algebra

- **Standard relational algebra**:
  - Row-numbering operator ϱ readily available in `SQL:1999`.
  - ◿, ε, τ just efficient short-hands for standard algebra expressions.
- Efficiently implementable on any relational back-end.
- Various nice properties:
  - All joins are **equi-joins**.
  - All unions are **disjoint**; duplicate elimination made explicit.

π	column projection, renaming
σ	row selection
$\dot\cup$	disjoint union
$\setminus$	difference
δ	duplicate elimination
$\times$	Cartesian product
$\bowtie$	equi-join
ϱ	row-numbering
◿	staircase join
ε, τ	element/text node construction
⊛	arithmetic/comparison/Boolean operator ∗

Relational Sequence Encoding

- XQuery: **ordered sequences** of **items**.
  - Sequences are always **flat**.
  - Possibly **heterogeneous** sequences.
- Encode sequence order in column *pos*.
- Columns *ref* and *kind* reference key *ref* in **value container** *kind*.
  - MonetDB's `multijoin` for efficient value lookup.

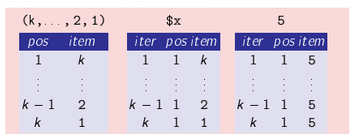


Loop-Lifting: Turn Iteration into Joins

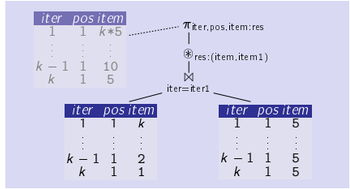
```
for $x in (k, ..., 2, 1)
return $x * 5
```

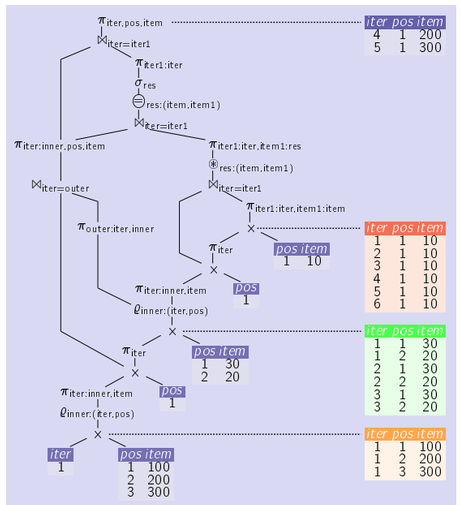
```
for $x in (100,200,300) return
 for $y in (30,20) return
  if ($x eq $y * 10) then $x else ()
```

- Encode iteration in column *iter*.



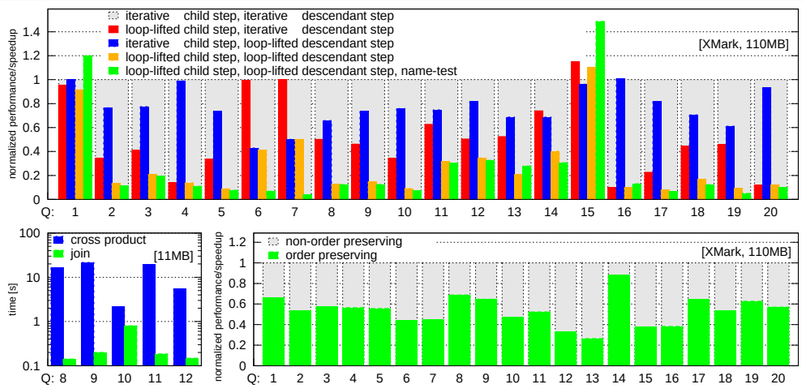
- **Bulk-oriented** processing of `for` clauses.



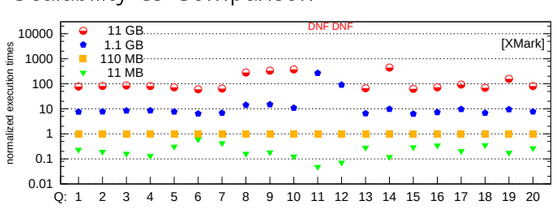


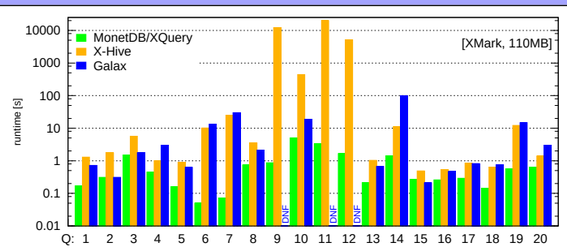
Optimizations

- Loop-lifted staircase join
  - perform embedded XPath location steps for multiple (nested) iterations in a single document-scan
- Name-test push-down
  - perform name- and kind-tests before/during loop-lifted staircase join instead of afterwards
- Join recognition
- Order awareness & order optimizations
  - order-preserving physical algebra
  - eliminate non-necessary sorting



Scalability & Comparison





Pathfinder + MonetDB = MonetDB/XQuery

- Available as open-source software
  - Mozilla-like license
- Download & developers website
  - `http://sf.net/projects/monetdb/`
- MonetDB/XQuery homepage
  - `http://monetdb-xquery.org/`

MonetDB XQuery