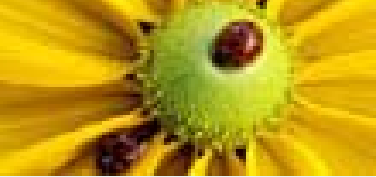


UvA 2005
Automated Reasoning

XPath Engine Performance under Equivalent Expressions

XHöskuldur Hlynsson
XChristian Kissig
XLaura Rimell



Outline of presentation

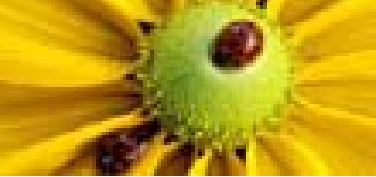
Introduction and Methodology

- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design: Rewrite rules
- Design: Hand crafted rewrites
- Design: Scheme for applying rewrite rules
- Design: Document set
- Design: Engines to evaluate
- Design: Framework for evaluation

Implementation

Results

- The Task
- Methodology
- Running the Tests
- Conclusions



The Task

Introduction and Methodology

- Outline of presentation

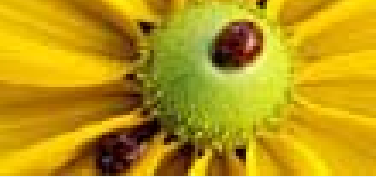
● The Task

- Design considerations
- Design: Basic Queries
- Design: Rewrite rules
- Design: Hand crafted rewrites
- Design: Scheme for applying rewrite rules
- Design: Document set
- Design: Engines to evaluate
- Design: Framework for evaluation

Implementation

Results

- XPath contains redundancies
- design Benchmark for XPath 1.0
- point out the engines tolerance towards those redundancies
- test the engines optimization techniques
- assist the developer in finding out weaknesses



Design considerations

Introduction and Methodology

- Outline of presentation
- The Task

● Design considerations

- Design: Basic Queries
- Design: Rewrite rules
- Design: Hand crafted rewrites
- Design: Scheme for applying rewrite rules
- Design: Document set
- Design: Engines to evaluate
- Design: Framework for evaluation

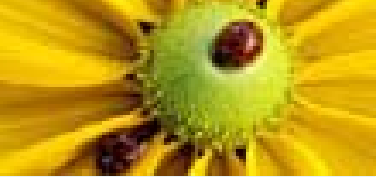
Implementation

Results

In order to carry out the benchmark we need:

- Set of basic queries
- Rewrite rules
- Hand crafted rewrites
- Scheme for applying rewrite rules
- Document set to apply the queries to
- XPath implementations to evaluate
- Framework to
 - ◆ feed initial and rewritten queries to a set of engines
 - ◆ collect results and statistics of runs

We now take a further look at those items and our ideas on implementing them.



Design: Basic Queries

Introduction and Methodology

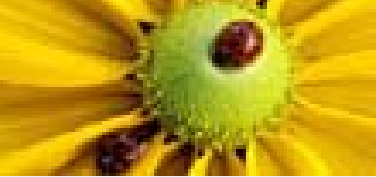
- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design: Rewrite rules
- Design: Hand crafted rewrites
- Design: Scheme for applying rewrite rules
- Design: Document set
- Design: Engines to evaluate
- Design: Framework for evaluation

Implementation

Results

Criteria for a *good* set of initial queries:

- *Naturalness*: An initial term should be natural in the sense that it could be written by a human or generated by some transformation process
- *Satisfiability*: A human would write more satisfiable terms than unsatisfiable
- *Unsatisfiability*: Unsatisfiable evaluations should be short circuited
- *Size*: Initial terms should be of non-trivial size.
- *Diversity*: Initial terms should cover most constructions from the specification
- *Applicability*: Queries should be rewriteable by at least one rule.



Design:Rewrite rules

Introduction and Methodology

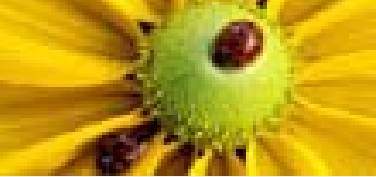
- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design:Rewrite rules
- Design:Hand crafted rewrites
- Design:Scheme for applying rewrite rules
- Design:Document set
- Design:Engines to evaluate
- Design:Framework for evaluation

Implementation

Results

Broadly speaking, the following categories of rewrite rules were identified:

- *Navigational Equivalences*: based on navigational steps
- *Predicate Equivalences*: inside predicates
- *Boolean Equivalences*: Tautologies and contradictions
- *DTD/XSchema-aware queries*: a priori knowledge in the form of DTDs/XSchemas



Design:Hand crafted rewrites

Introduction and Methodology

- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design:Rewrite rules
- **Design:Hand crafted rewrites**
- Design:Scheme for applying rewrite rules
- Design:Document set
- Design:Engines to evaluate
- Design:Framework for evaluation

Implementation

Results

There is a need for hand crafted rewrites of the queries because rewrite rules are basically an inverse of a normalization procedure done by a query engine.

Practical implementations should consider hiding the actual rules and queries used and only give results as a list of areas of potential improvement in engine capabilities and a list of failures to comply to specs.

Design:Scheme for applying rewrite rules

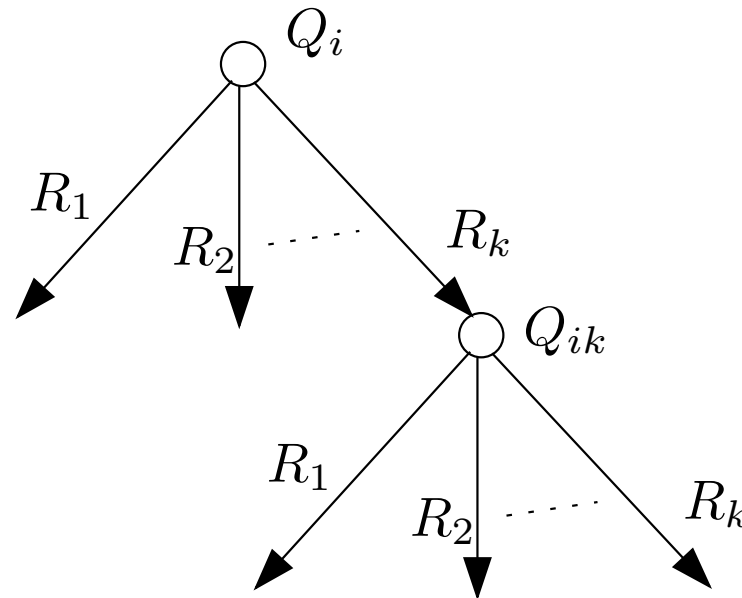
Introduction and Methodology

- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design: Rewrite rules
- Design: Hand crafted rewrites
- Design: Scheme for applying rewrite rules
- Design: Document set
- Design: Engines to evaluate
- Design: Framework for evaluation

Implementation

Results

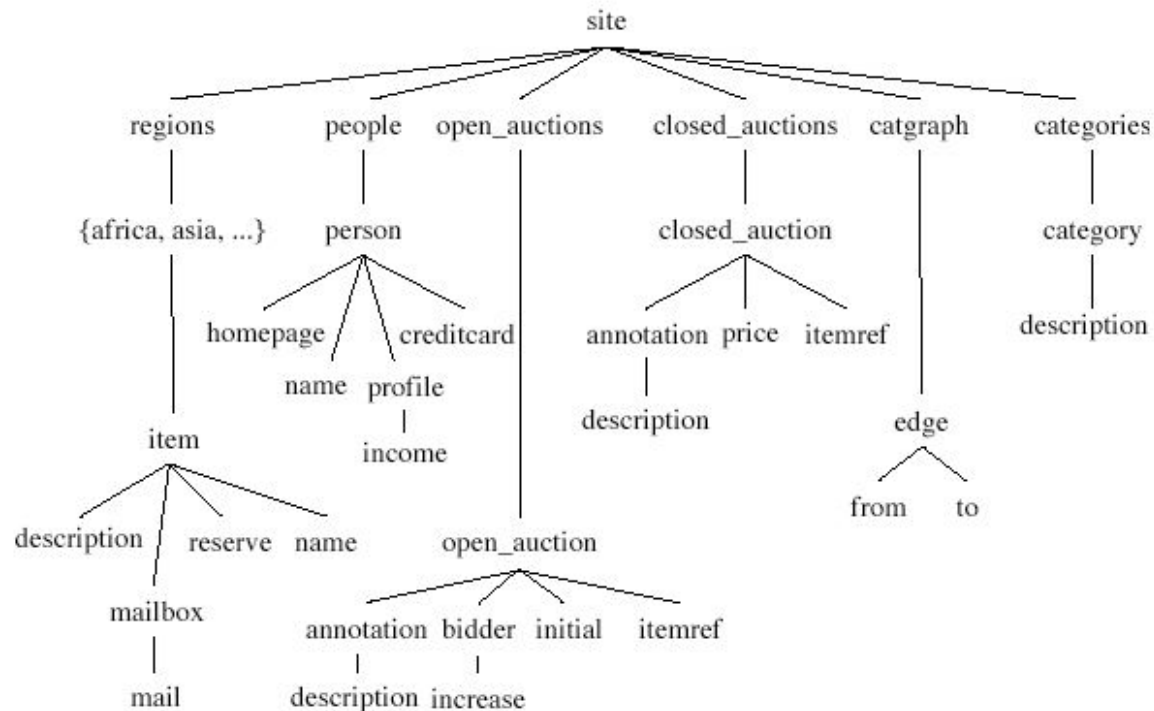
A rewrite scheme as the one depicted below has the property that no thought is needed to decide which rules to apply – even in the presence of idempotent combinations of rules. Simply apply all the rules to all queries iteratively.



For each run: Decide initial query set, rewrite rules and depth of rewrite tree. Press Run.

Design:Document set

Queries are based on XMark generated documents. XMark is in common use among XML benchmarking applications.



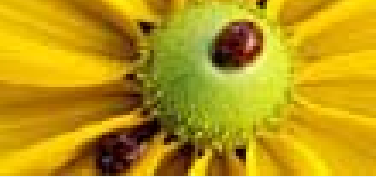
This figure show the major elements of the XMark structure.

Introduction and Methodology

- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design: Rewrite rules
- Design: Hand crafted rewrites
- Design: Scheme for applying rewrite rules
- Design: Document set
- Design: Engines to evaluate
- Design: Framework for evaluation

Implementation

Results



Design:Engines to evaluate

Introduction and Methodology

- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design:Rewrite rules
- Design:Hand crafted rewrites
- Design:Scheme for applying rewrite rules
- Design:Document set
- Design:Engines to evaluate
- Design:Framework for evaluation

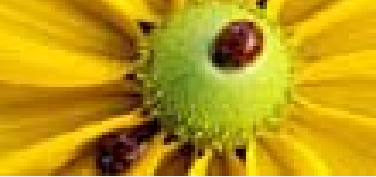
Implementation

Results

Two engines were handed to us for evaluation: Galax 0.4.0
Saxon 8.2

XPath queries processed in a two stage fashion by these engines:

1. Query parsed and translated into an internal representation whereby they are reduced to a small sublanguage,i.e. normalized.
2. Query is processed w.r.t. a loaded document.
Engine evaluation should respect this two stage procedure.



Design:Framework for evaluation

Introduction and Methodology

- Outline of presentation
- The Task
- Design considerations
- Design: Basic Queries
- Design:Rewrite rules
- Design:Hand crafted rewrites
- Design:Scheme for applying rewrite rules
- Design:Document set
- Design:Engines to evaluate

● Design:Framework for evaluation

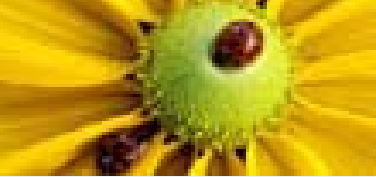
Implementation

Results

For running the queries a wrapper for feeding the queries to the engines and collecting statistics should be provided.

A program configurable in the depth of the tree, rewrite rules to be applied and initial set of queries should be provided also.

Ideally, a program to generate the initial queries themselves from the document structure should be provided too.



Overview

Introduction and Methodology

Implementation

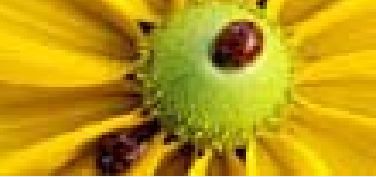
● Overview

● Technicalities

Results

An implementation should fulfill as many of the design criteria as possible.

- orientable towards different aspects of the XPath Spec
- scalable to applications of the XPath engine
- scalable towards future developments



Technicalities

Introduction and Methodology

Implementation

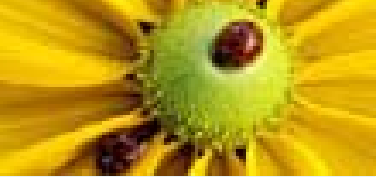
● Overview

● Technicalities

Results

The following seem natural choices for implementation:

- Wrapper: Perl
 - ◆ easy creation of XML documents
 - ◆ fast Pattern Matching
- Rewrites: OCaml
 - ◆ supports Term Rewriting by Pattern Matching
 - ◆ connects to the Theorem Prover COQ
 - ◆ includes Parser/Lexer structures
 - ◆ ships with advanced XML APIs



Results: Galax I

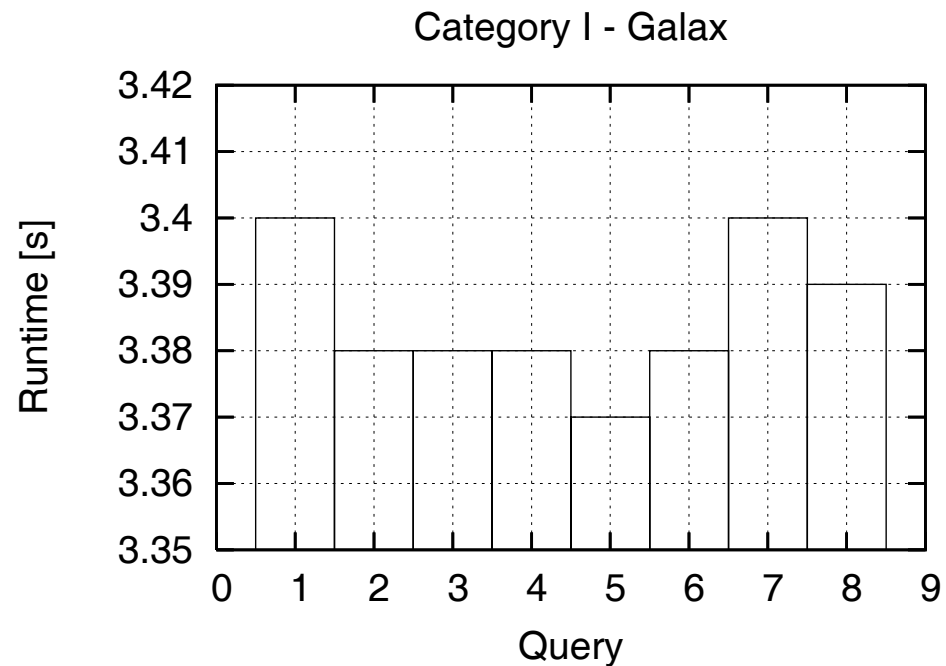
[Introduction and Methodology](#)

[Implementation](#)

Results

- **Results: Galax I**
- Results: Saxon I
- Results: Galax II
- Results: Saxon II
- Conclusions

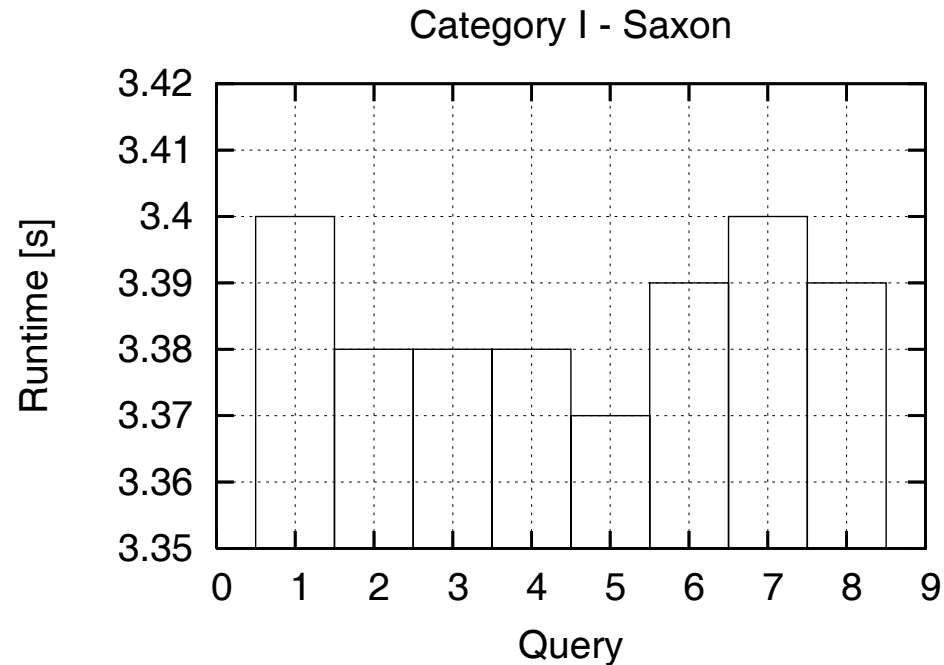
Results differ by 0.02 seconds such that no conclusions can be drawn for the measurement.



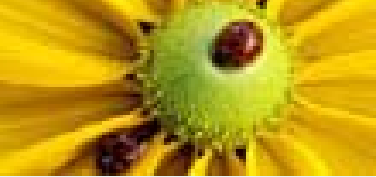
- Queries navigationally equivalent
- XMark document factor 0.3

Results: Saxon I

Again, the results do not differ enough to draw conclusions.



- Queries navigationally equivalent
- XMark document factor 0.3



Results: Galax II

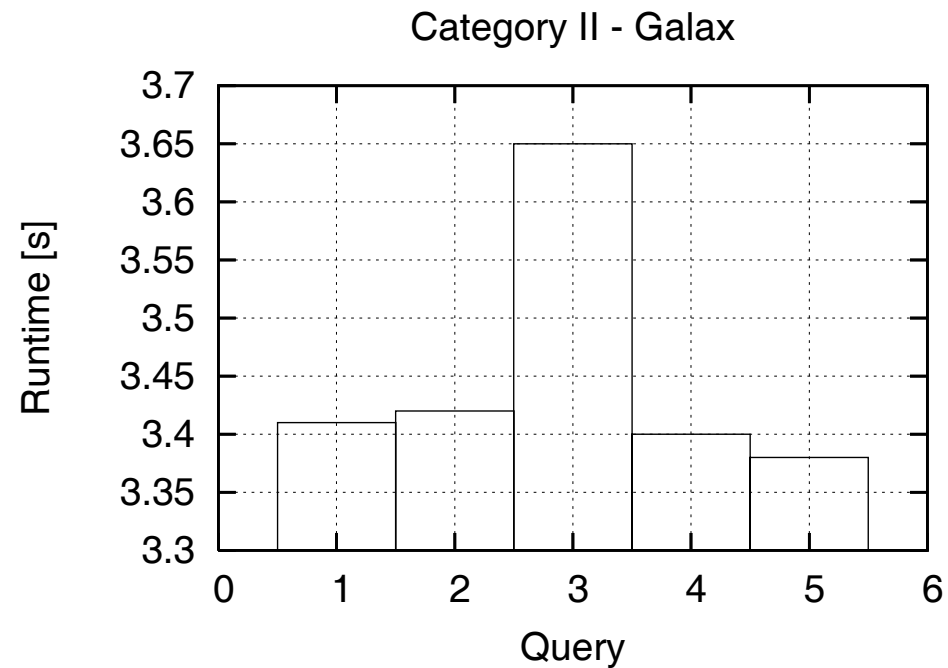
[Introduction and Methodology](#)

[Implementation](#)

Results

- Results: Galax I
- Results: Saxon I
- **Results: Galax II**
- Results: Saxon II
- Conclusions

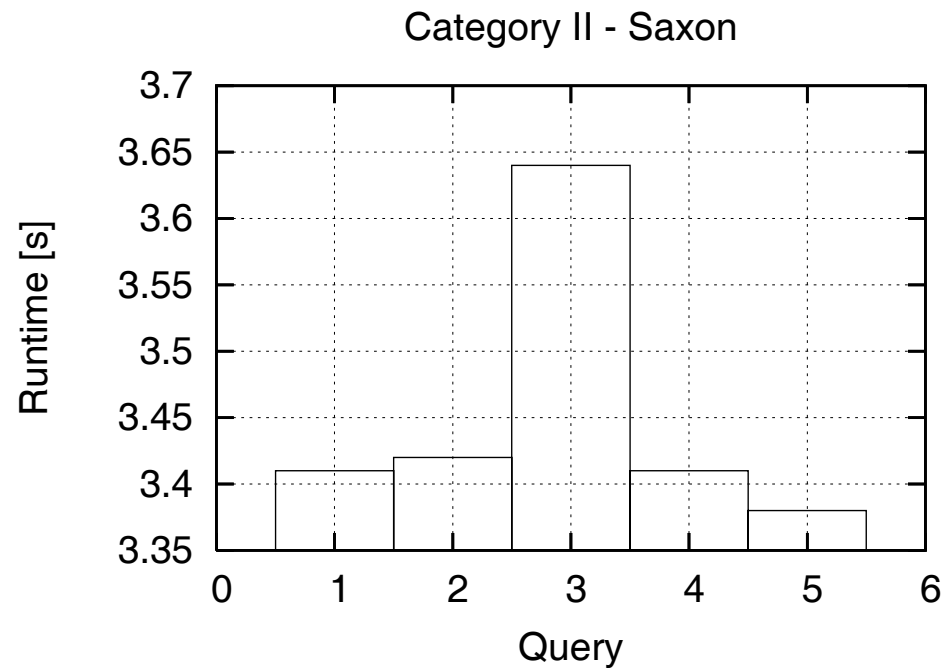
Third query takes longer to process. Points out weakness of the engine.



- Queries equivalent by Predicates
- XMark document factor 0.3

Results: Saxon II

Third query takes longer to process. Points out weakness of the engine.



- Queries equivalent by Predicates
- XMark document factor 0.3



Conclusions

Introduction and Methodology

Implementation

Results

- Results: Galax I
- Results: Saxon I
- Results: Galax II
- Results: Saxon II
- **Conclusions**

A few concluding remarks:

- Rewrite Rules are reverse to Normalization
- Equivalences w.r.t. a DTD/XSchema
- Manual Collection of Queries needs to be extended
- Adaption to other XPath engines, like XHive, Jaxen, ...
- Specialization of the Benchmark
 - ◆ Wireless Applications: memory usage
 - ◆ Stream Processing: attention to forward/backward Axes
- Implementation of the Query Generator