SAXONICA.COM
XSLT AND XQUERY PROCESSING

XML Tree Models for Efficient Copy

Michael Kay, Saxonica
XML Prague 2018

1

A compiler should be written in its own language

SAXONICA.COM
XSLT AND XQUERY PROCESSING

2

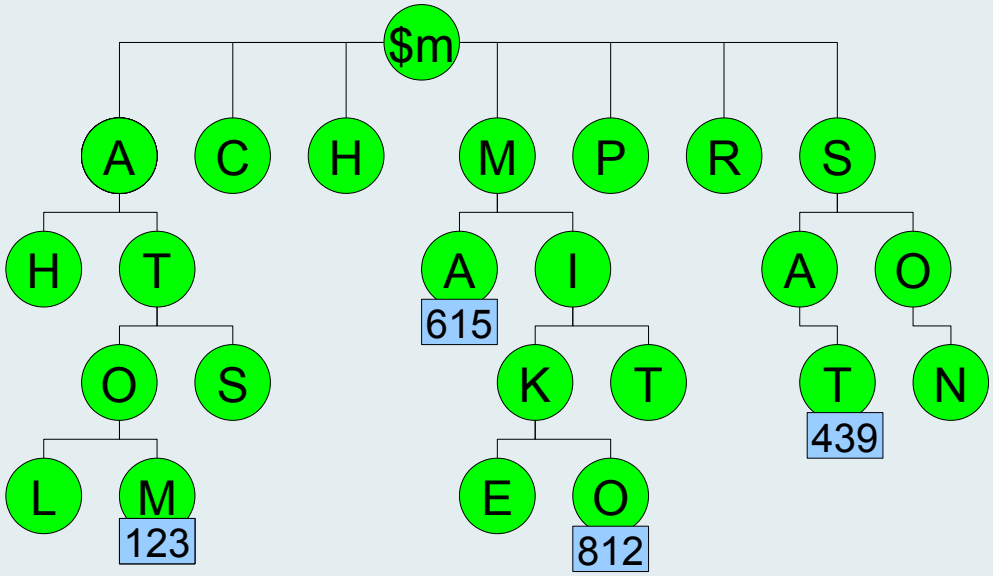
XSLT
Transformations
take
(at least)
linear time

3

JSON trees
can be updated
in
constant time

Digression: { persistent immutable versioned } tries

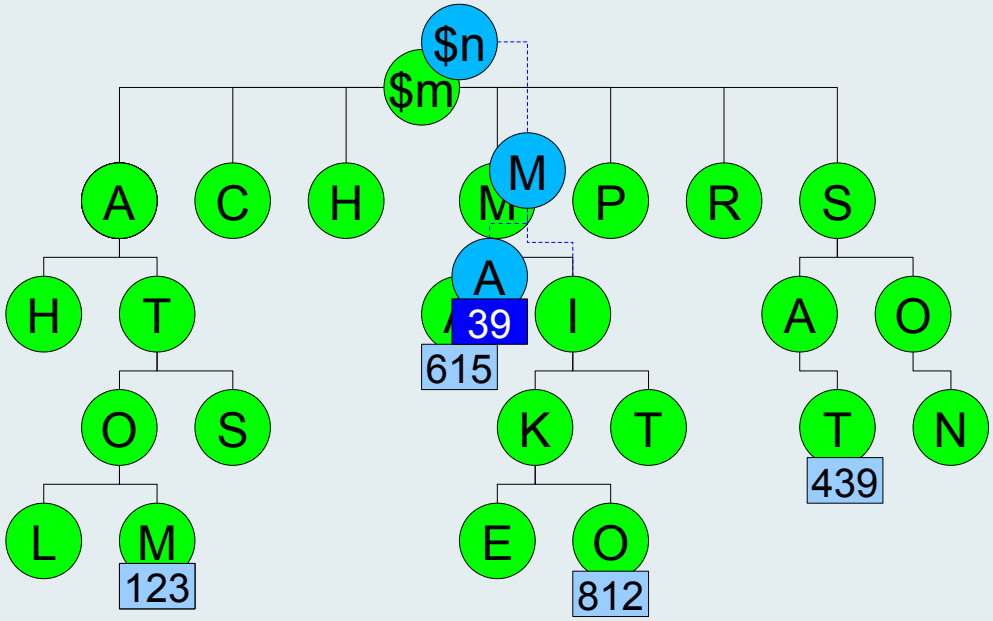
```
$m := map {
  "atom":123,
  "ma":615,
  "miko":812,
  "sat":439, }
```




```
$n := map:put($m, "ma", 39)
```

Digression: { persistent immutable versioned } tries

```
$n := map {
  "atom":123,
  "ma":39,
  "miko":812,
  "sat":439, }
```



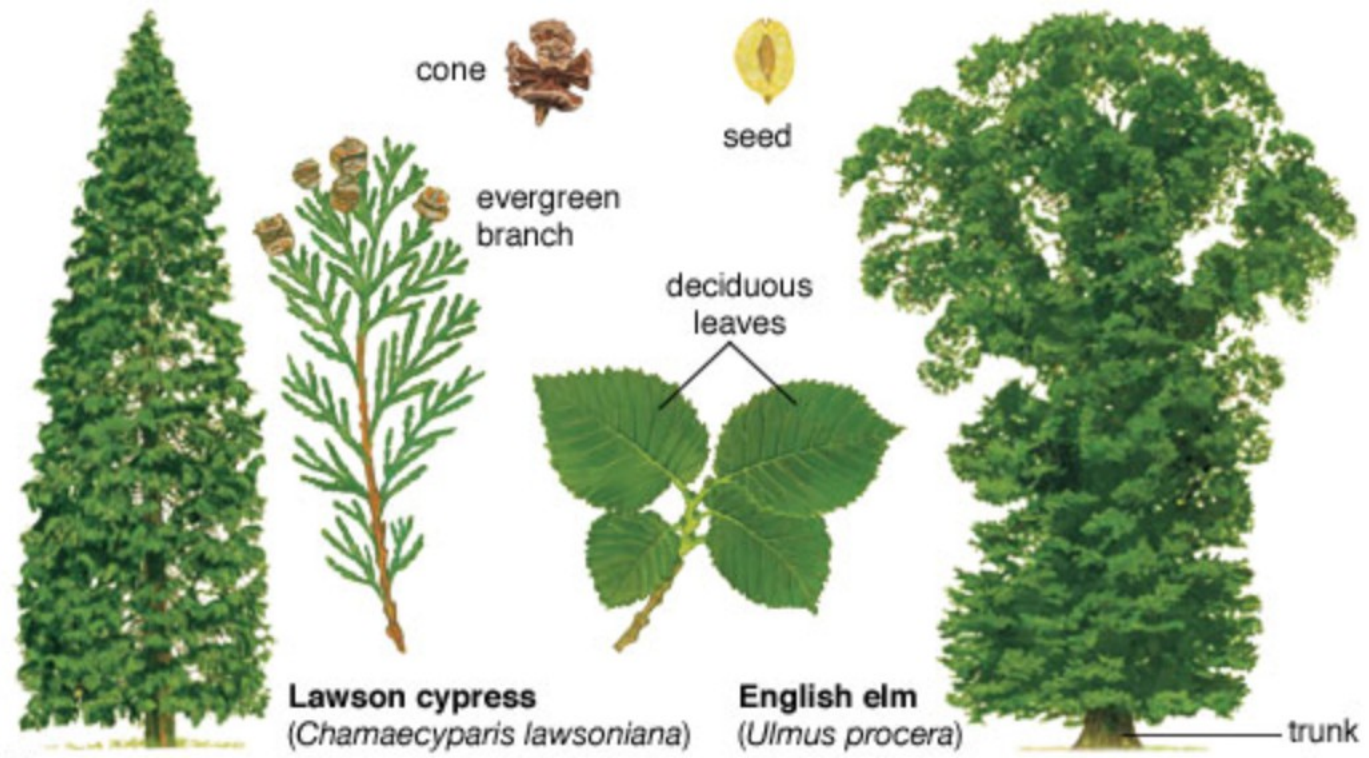
4

Trie updates
are cheap
because there are
no parent pointers

KONICA.COM
XQUERY PROCESSING

5

There is no intrinsic
reason why XML trees have
parent pointers
and JSON trees do not



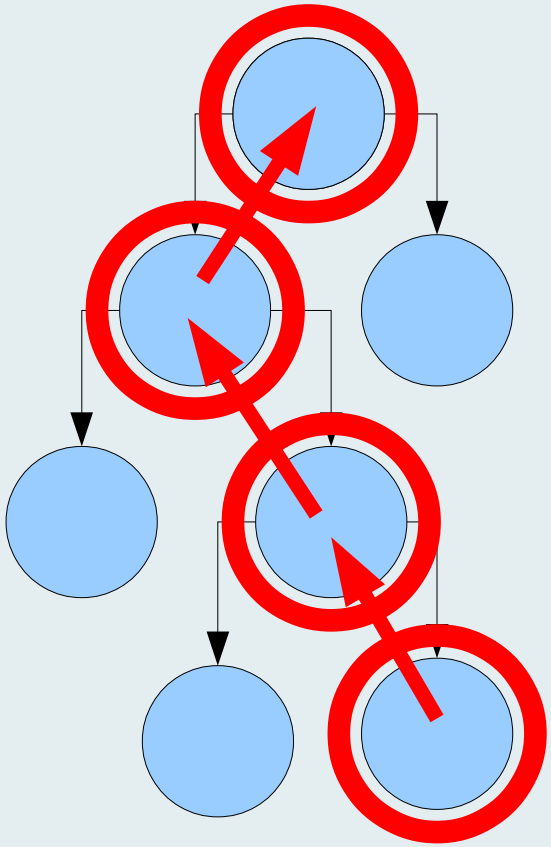
6

Access to parents and ancestors is useful

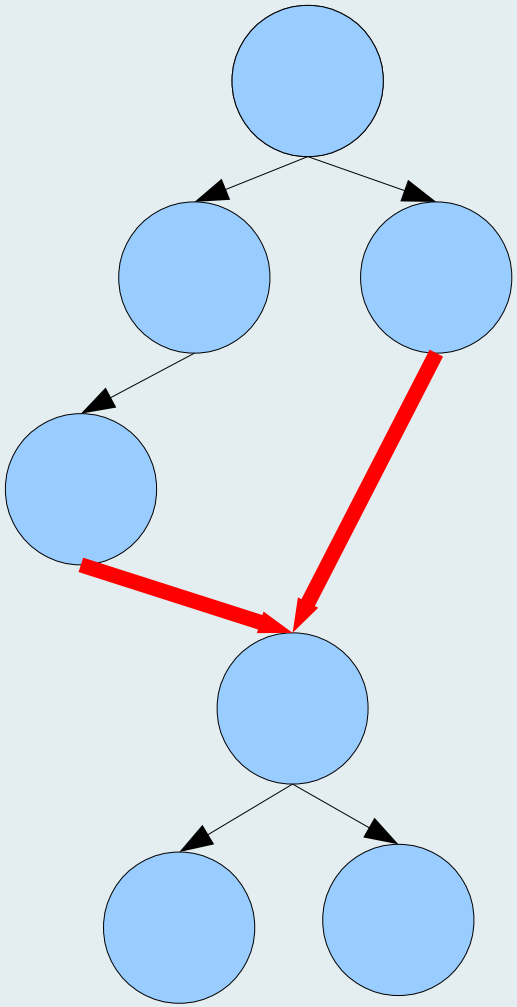
```
if (ancestor::*/@xml:lang = 'de')
```

7

But you can find your way back to ancestor nodes by retracing your steps



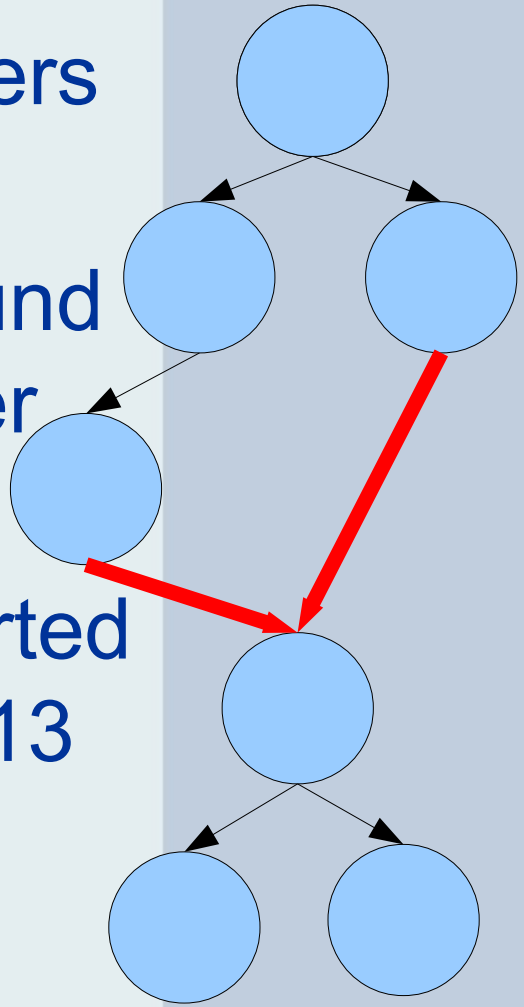
8



Without parent pointers, copying subtrees becomes trivial

KL-tree

- K-nodes: Java objects with pointers to children
- L-nodes: transient wrappers around K-nodes containing parent pointer and sibling position
- L-nodes have identity, can be sorted into document order, support all 13 XPath axes
- Fast but not fast enough



9

With the Saxon TinyTree it's a bit more complicated

The TinyTree

Depth	Kind	Name	A	B	Next/Up
0	Doc	-	-	-	-
1	Elem	2345	➤ first att	➤ first ns	0
2	WS	-	xxx	xxx	3
2	Elem	5678	➤ first att	➤ first ns	4
3	Text	-	➤ content	length	3
2	Elem	5678	➤ first att	➤ first ns	6

Very fast searching for elements by name

10

Where can we use cheap subtree copying?

```
<xsl:copy-of select="*"/>
```

?

```
<xsl:mode
  on-no-match=
  "shallow-copy"/>
```

XQUERY

```
<x>{a/b/c}</x>
```

XSLT Update Extension?

```
<uxsl:update>
  <uxsl:delete select=".//note"/>
</uxsl:update>
```

XQUERY UPDATE

```
transform $doc {
  delete .//note
}
```

Summary

- You need access to parents but you don't need parent pointers
- *Without parent pointers, copying subtrees is cheap*
- If subtree copying is cheap, small changes can be made in constant time
  - (which enables us to write an XSLT compiler in XSLT)
- *Performance figures: see paper*