University of Munich – Institute of Computer Science

http://www.pms.informatik.uni-muenchen.de

Rules for Efficient XPath Evaluation

Dan Olteanu, Holger Meuss, Tim Furche, François Bry

olteanu@pms.informatik.uni-muenchen.de

The issue of concern

Our goal

An efficient XPath evaluation in a stream-based context.

The problem

- XPath allows back and forth navigation in the XML tree using location paths.
- Backward navigation is not appropriate in a stream-based context.
- Can backward navigation be avoided in XPath ?

Our current contribution

Rules for rewriting arbitrary XPath 1.0 location paths into ones without backward navigation:

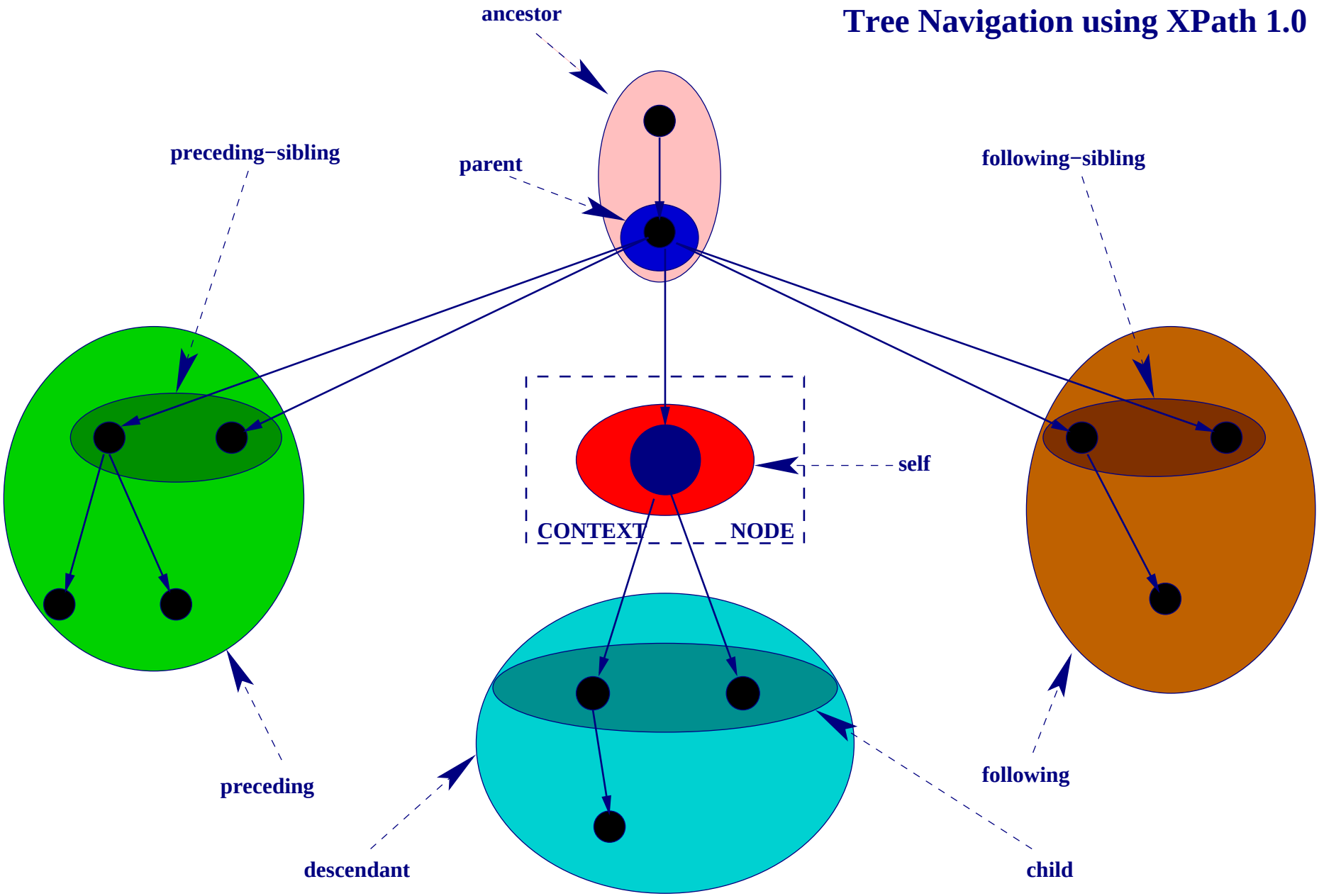
- two sets of equivalences between location paths with and without backward navigation.
- syntactical characterization of non-rewritable location path class.
- rewriting of the above mentioned class using XPath 2.0 / XQuery 1.0 variables.

Motivation

Stream-based query processing gained interest due to its application in

- handling large documents that can not be processed in memory, e.g. corpora, astronomical or biological data.
- data integration, e.g. Tukwila.
- publish-subscribe architectures, e.g. XTrie, XFilter.

The existing stream-based query systems consider only a subset of XPath capabilities for forward navigation. We aim at enabling the entire set of XPath navigation capabilities to be considered for stream-based processing.


Tree Navigation using XPath 1.0
ancestor
preceding−sibling
parent
following−sibling
self
CONTEXT NODE
preceding
following
descendant
child

The xPath location path language

Grammar

$$
\begin{aligned}
\mathit{path} ::= {} & \mathit{path}\ \texttt{|}\ \mathit{path} \mid \texttt{/}\ \mathit{path} \mid \mathit{path}\ \texttt{/}\ \mathit{path} \mid \mathit{step}\ . \\
\mathit{step} ::= {} & \mathit{axis}\ \texttt{::}\ \mathit{nodetest} \mid \mathit{axis}\ \texttt{::}\ \mathit{nodetest}\ \texttt{[}\ \mathit{qualifier}\ \texttt{]} \mid \bot\ . \\
\mathit{qualifier} ::= {} & \mathit{qualifier}\ \texttt{and}\ \mathit{qualifier} \mid \mathit{qualifier}\ \texttt{or}\ \mathit{qualifier} \mid \\
& \texttt{(}\ \mathit{qualifier}\ \texttt{)} \mid \mathit{path}\ \texttt{=}\ \mathit{path} \mid \mathit{path}\ \texttt{==}\ \mathit{path} \mid \mathit{path}\ . \\
\mathit{axis} ::= {} & \mathit{reverse\_axis} \mid \mathit{forward\_axis}\ . \\
\mathit{reverse\_axis} ::= {} & \texttt{parent} \mid \texttt{ancestor} \mid \texttt{ancestor-or-self} \mid \\
& \texttt{preceding} \mid \texttt{preceding-sibling}\ . \\
\mathit{forward\_axis} ::= {} & \texttt{self} \mid \texttt{child} \mid \texttt{descendant} \mid \texttt{descendant-or-self} \mid \\
& \texttt{following} \mid \texttt{following-sibling}\ . \\
\mathit{nodetest} ::= {} & \mathit{tagname} \mid \texttt{*} \mid \texttt{text()} \mid \texttt{node()}\ .
\end{aligned}
$$

Differences between xPath and XPath

What we leave out

- references
- functions and most operators
- attributes
- schema types and namespaces
- comments and processing instructions
- abbreviated syntax

What we add

- equality based on node identity (==), cf. XQuery, XPath 2.0 Datamodel.
  p == q computed as `count(`p | q`) < count(`p`) + count(`q`)` in XPath 1.0.
- unions at every location step, cf. XQuery 1.0.

Denotational Semantics for XPath

$\mathcal{Q}$: Qualifier $\rightarrow$ Node $\rightarrow$ Boolean

$$\mathcal{Q}[\![q_1 \texttt{ and } q_2]\!]x = \mathcal{Q}[\![q_1]\!]x \wedge \mathcal{Q}[\![q_2]\!]x$$

$$\mathcal{Q}[\![q_1 \texttt{ or } q_2]\!]x = \mathcal{Q}[\![q_1]\!]x \vee \mathcal{Q}[\![q_2]\!]x$$

$$\mathcal{Q}[\![\texttt{(}q\texttt{)}]\!]x = \mathcal{Q}[\![q]\!]x$$

$$\mathcal{Q}[\![p]\!]x = \mathcal{S}[\![p]\!]x \neq \emptyset$$

$$\mathcal{Q}[\![p_1 \texttt{ = } p_2]\!]x = \{x_1 | x_1 \in \mathcal{S}[\![p_1]\!]x \wedge x_2 \in \mathcal{S}[\![p_2]\!]x \wedge \mathit{value}(x_1) = \mathit{value}(x_2)\} \neq \emptyset$$

$$\mathcal{Q}[\![p_1 \texttt{ == } p_2]\!]x = \{x_1 | x_1 \in \mathcal{S}[\![p_1]\!]x \wedge x_1 \in \mathcal{S}[\![p_2]\!]x\} \neq \emptyset$$

$$\mathcal{S} : \text{Pattern} \to \text{Node} \to \text{Set(Node)}$$

$$\begin{aligned}
\mathcal{S}[\![p_1 \mid p_2]\!]x &= \mathcal{S}[\![p_1]\!]x \cup \mathcal{S}[\![p_2]\!]x \\
\mathcal{S}[\![/p]\!]x &= \mathcal{S}[\![p]\!](\mathit{root}(x)) \\
\mathcal{S}[\![p_1/p_2]\!]x &= \{x_2 | x_1 \in \mathcal{S}[\![p_1]\!]x \wedge x_2 \in \mathcal{S}[\![p_2]\!]x_1\} \\
\mathcal{S}[\![p\texttt{[}q\texttt{]}]\!]x &= \{x_1 | x_1 \in \mathcal{S}[\![p]\!]x \wedge \mathcal{Q}[\![q]\!]x_1\} \\
\mathcal{S}[\![\texttt{self::}n]\!]x &= \{x_1 | x_1 = x \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{child::}n]\!]x &= \{x_1 | x_1 \in \mathit{children}(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{parent::}n]\!]x &= \{x_1 | x_1 \in \mathit{parent}(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{descendant::}n]\!]x &= \{x_1 | x_1 \in \mathit{children}^+(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{descendant-or-self::}n]\!]x &= \{x_1 | x_1 \in \mathit{children}^*(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{ancestor::}n]\!]x &= \{x_1 | x_1 \in \mathit{parent}^+(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{ancestor-or-self::}n]\!]x &= \{x_1 | x_1 \in \mathit{parent}^*(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{preceding::}n]\!]x &= \{x_1 | x_1 \in \mathit{preceding}(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{preceding-sibling::}n]\!]x &= \{x_1 | x_1 \in \mathit{preceding\text{-}sibling}(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{following::}n]\!]x &= \{x_1 | x_1 \in \mathit{following}(x) \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\texttt{following-sibling::}n]\!]x &= \{x_1 | x_1 \in \mathit{following\text{-}sibling} \wedge \mathit{nodetest}(x_1, n)\} \\
\mathcal{S}[\![\bot]\!]x &= \emptyset
\end{aligned}$$

Location Path Equivalences

Path equivalence

Location paths p_1 and p_2 are *equivalent*, noted $p_1 \equiv p_2$, when, for every document and context node within, they select the same set of nodes.

$p_1 \equiv p_2$ if $\mathcal{S}[\![p_1]\!] = \mathcal{S}[\![p_2]\!]$, i.e. if $\mathcal{S}[\![p_1]\!]x = \mathcal{S}[\![p_2]\!]x$ for all nodes x (from any document).

1. Equivalences involving paths with reverse steps

General and Specific Equivalences

2. Basic equivalences

Adjunctions

- If $p_1 \equiv p_2$ and p relative, then $p_1/p \equiv p_2/p$.
- If $p_1 \equiv p_2$ and p_1, p_2 relative, then $p/p_1 \equiv p/p_2$.
- If $p_1 \equiv p_2$, then $p_1\,[q] \equiv p_2\,[q]$.

Qualifier flattening If p_2 relative, then $p\,[p_1/p_2] \equiv p\,[p_1\,[p_2]]$.

Qualifiers with joins If p_2 relative, then $p\,[p_1\ \theta\ /p_2] \equiv p\,[p_1\,[\texttt{self::node()}\ \theta\ /p_2]]$.

General Equivalences

- provide interaction between a location path p or $p/a_n::n$ and a reverse step $a_m::m$, outside or inside qualifiers:

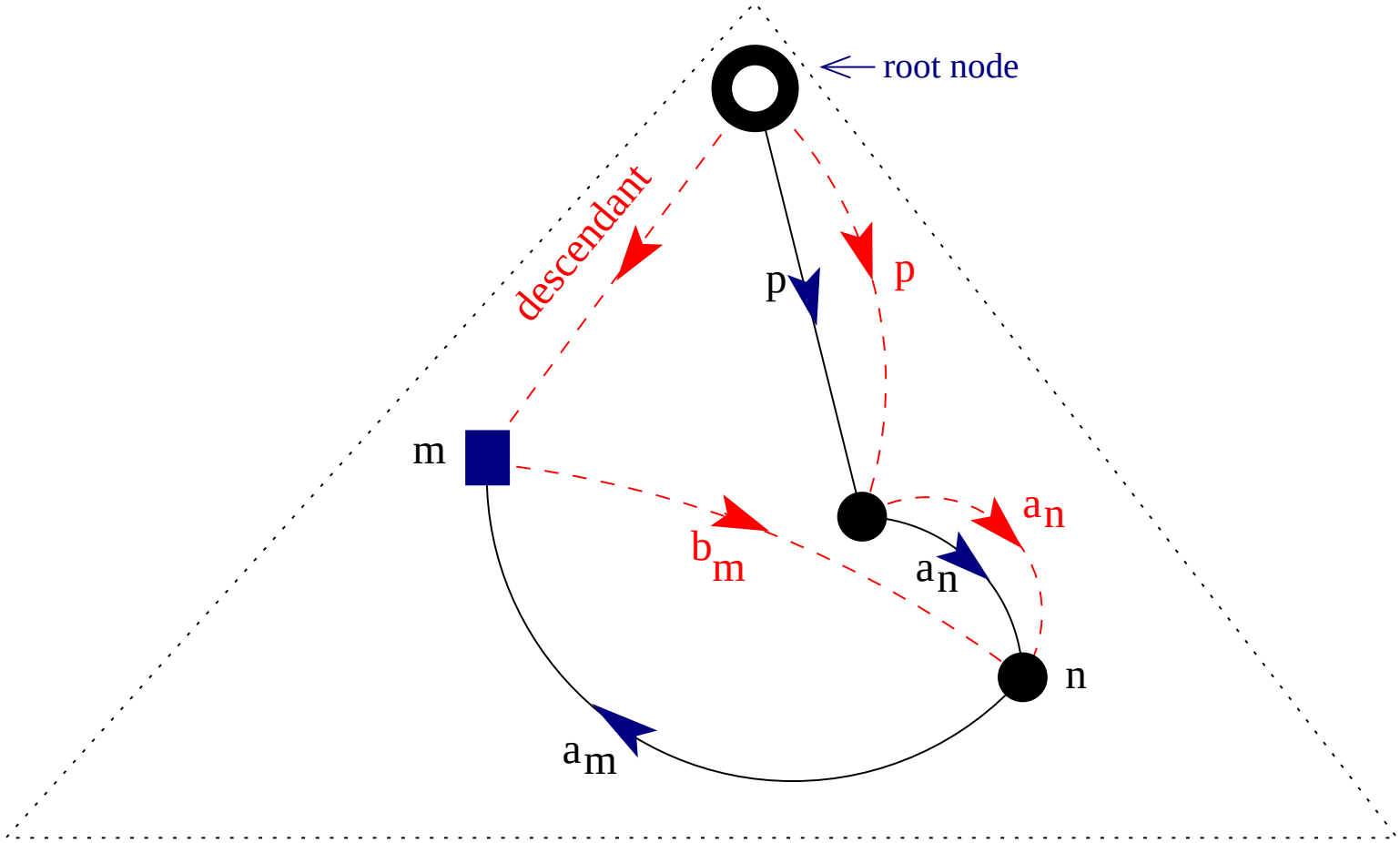
  $/p/a_n::n/a_m::m \equiv p'$ or $p[a_m::m] \equiv p'$

- p' does no longer contain the reverse axis a_m, but its *symmetrical* forward axis, i.e. if a_m is an `ancestor` axis, then the symmetrical forward axis is `descendant`.

- there are only two general equivalences, i.e. for reverse steps outside and inside qualifiers.

General Equivalences for Absolute Paths with Reverse Steps

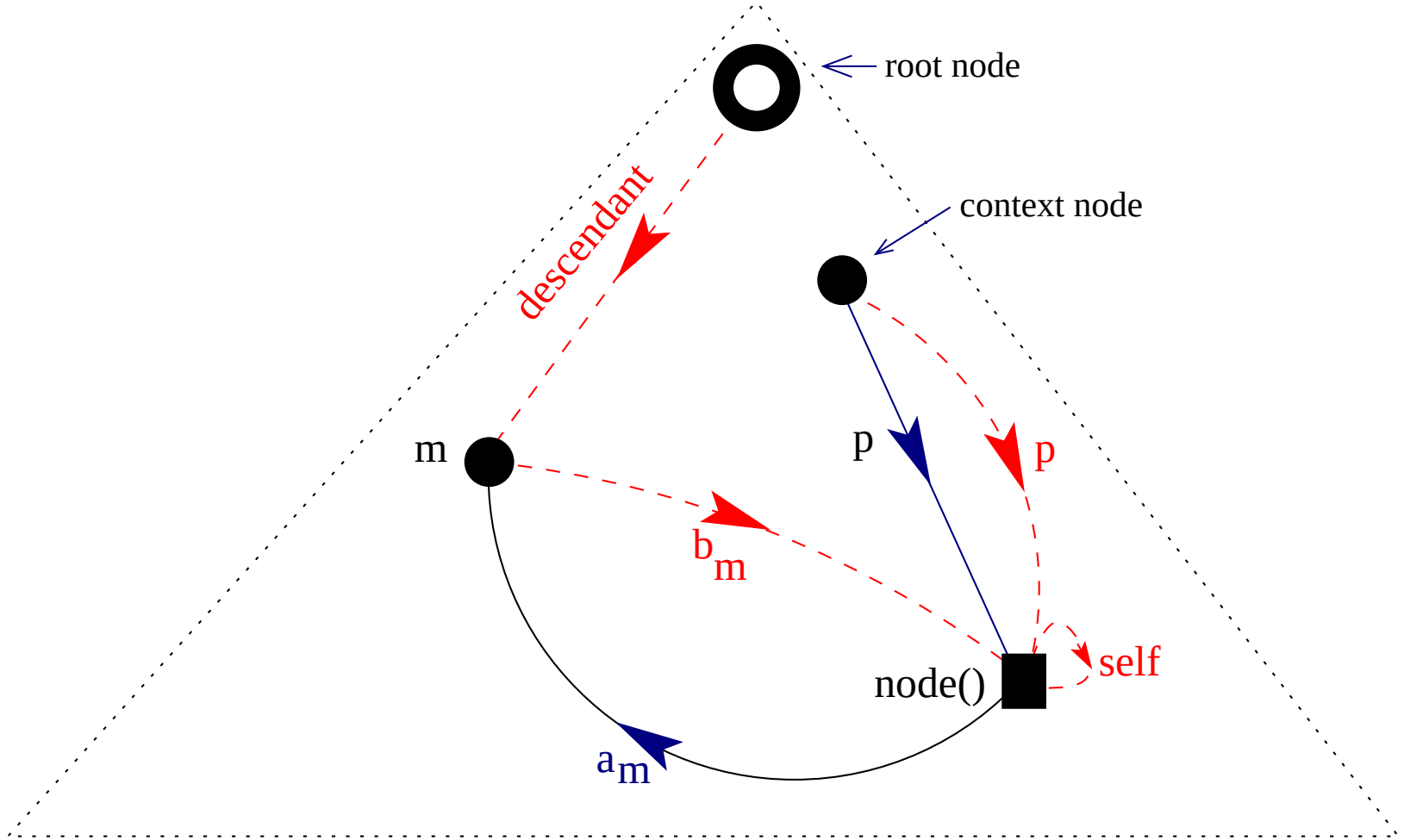
Let p be a relative path, m and n node tests, a_n a forward axis, a_m a reverse axis and b_m the symmetrical axis of a_m.



$$/\ p\ /\ a_n :: n\ /\ a_m :: m \quad \equiv \quad /\ \text{descendant} :: m\ [\ b_m :: n == /\ p\ /\ a_n :: n\ ]$$

General Equivalences for Qualifiers with Reverse Steps

Let p be a relative path, m a node test, a_m a reverse axis and b_m the symmetrical axis of a_m.



p [a_m:: m] $\equiv$ p [/ descendant :: m / b_m:: node() == self :: node()]

Specific Equivalences

Provide interaction between **every** reverse step L_r and **every** forward step L_f:

$$p/L_f/L_r \equiv p' \quad \text{or} \quad p/L_f\,[L_r] \equiv p'$$

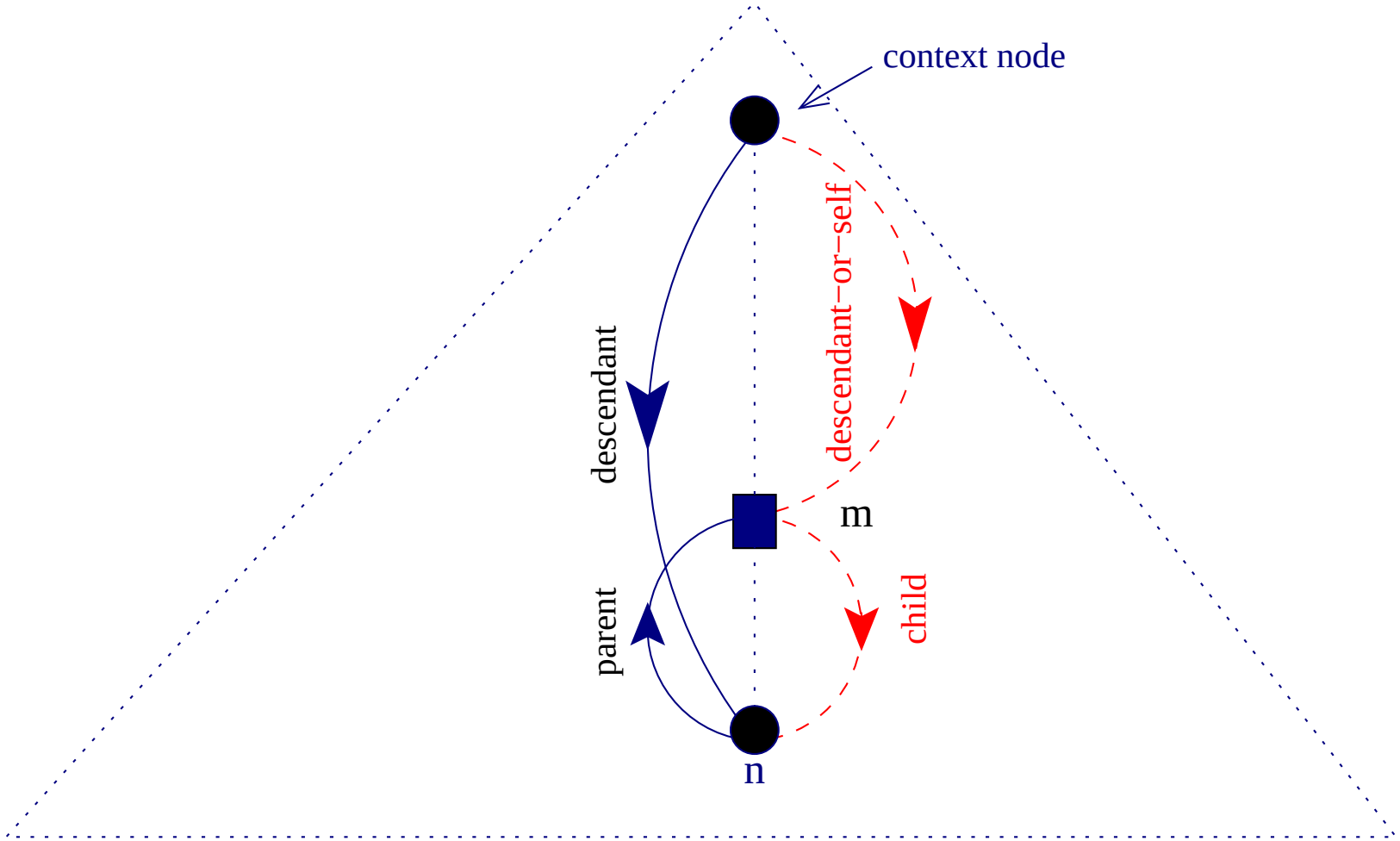
The equivalent location path p' can be of three kinds:

- without the reverse step L_r or
- with the reverse step L_r , but pushed leftwise or
- with the reverse step L_r , at the same position from left to right in the location path, but without the leading L_f.

There are (at least) 10 specific equivalences for each reverse step.

Specific Equivalences for Parent steps

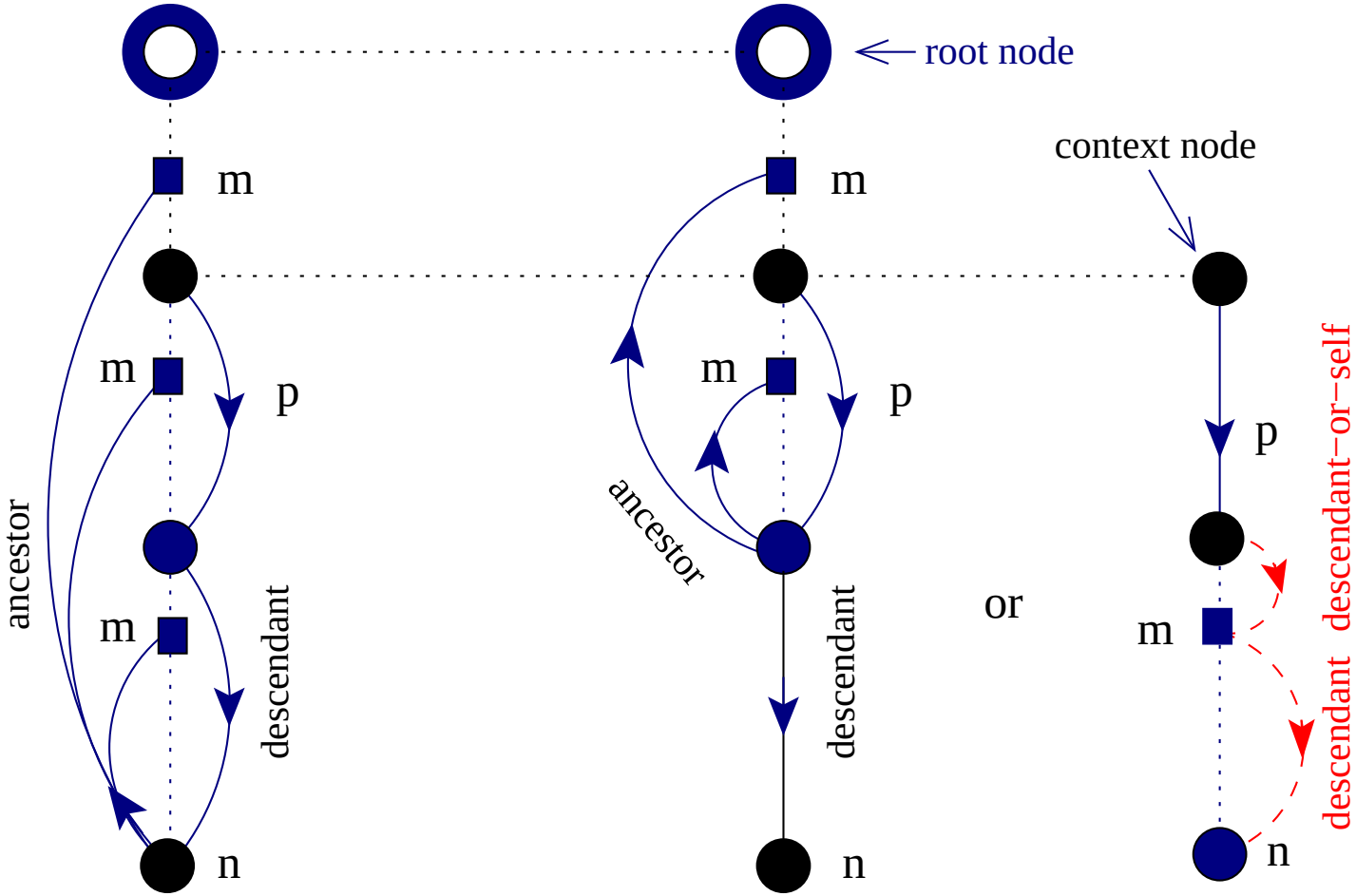
Consider the interaction between a `descendant` and a `parent` step outside qualifiers:



descendant :: n / parent :: m ≡ descendant−or−self :: m [child :: n]

Specific Equivalences for Ancestor steps

Consider the interaction between a `descendant` and an `ancestor` step outside qualifiers:

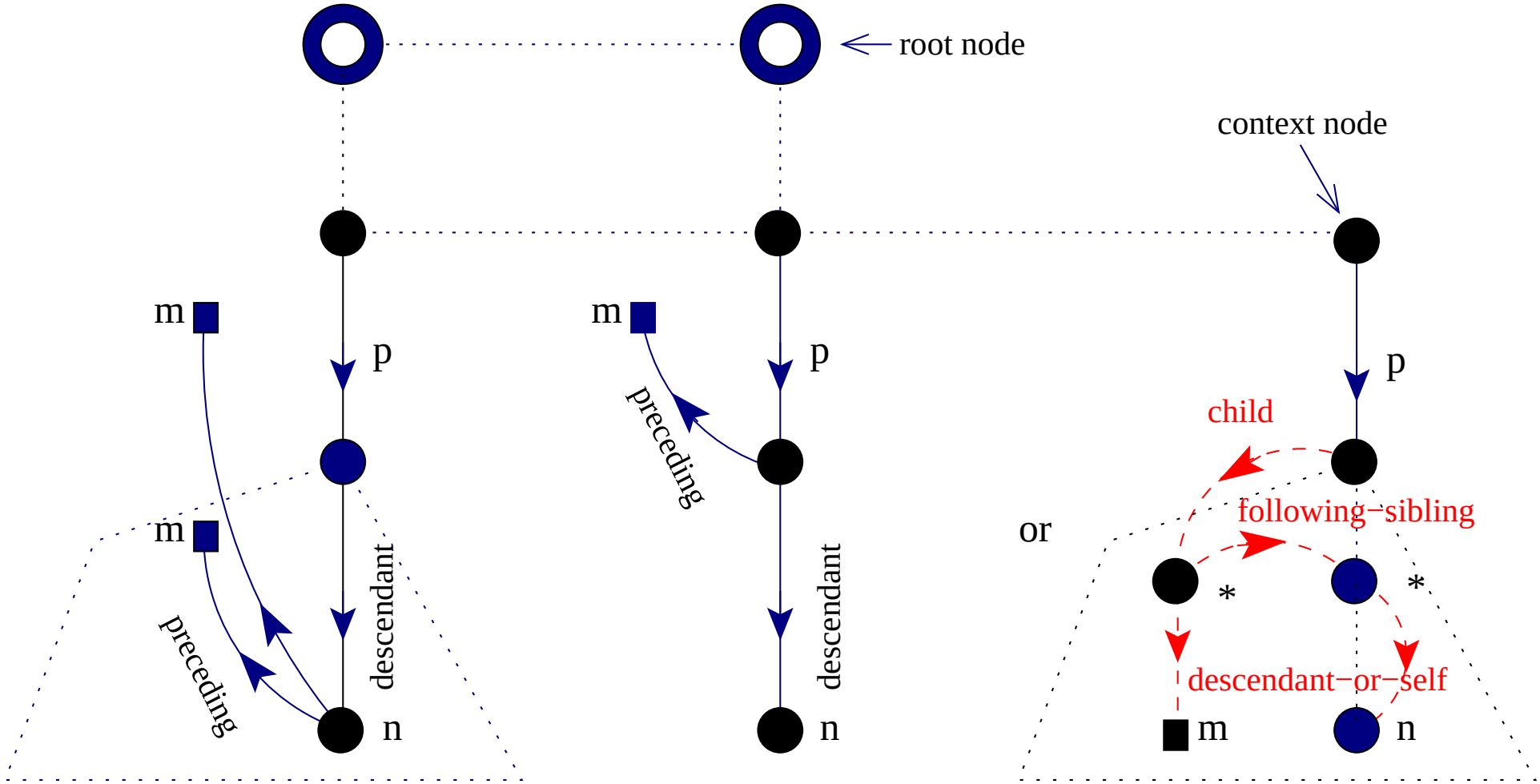


p / descendant :: n / ancestor :: m ≡ p [descendant :: n] / ancestor :: m

| p / descendant−or−self :: m [descendant :: n]

Specific Equivalences for Preceding steps

Consider the interaction between a `descendant` and a `preceding` step outside qualifiers:

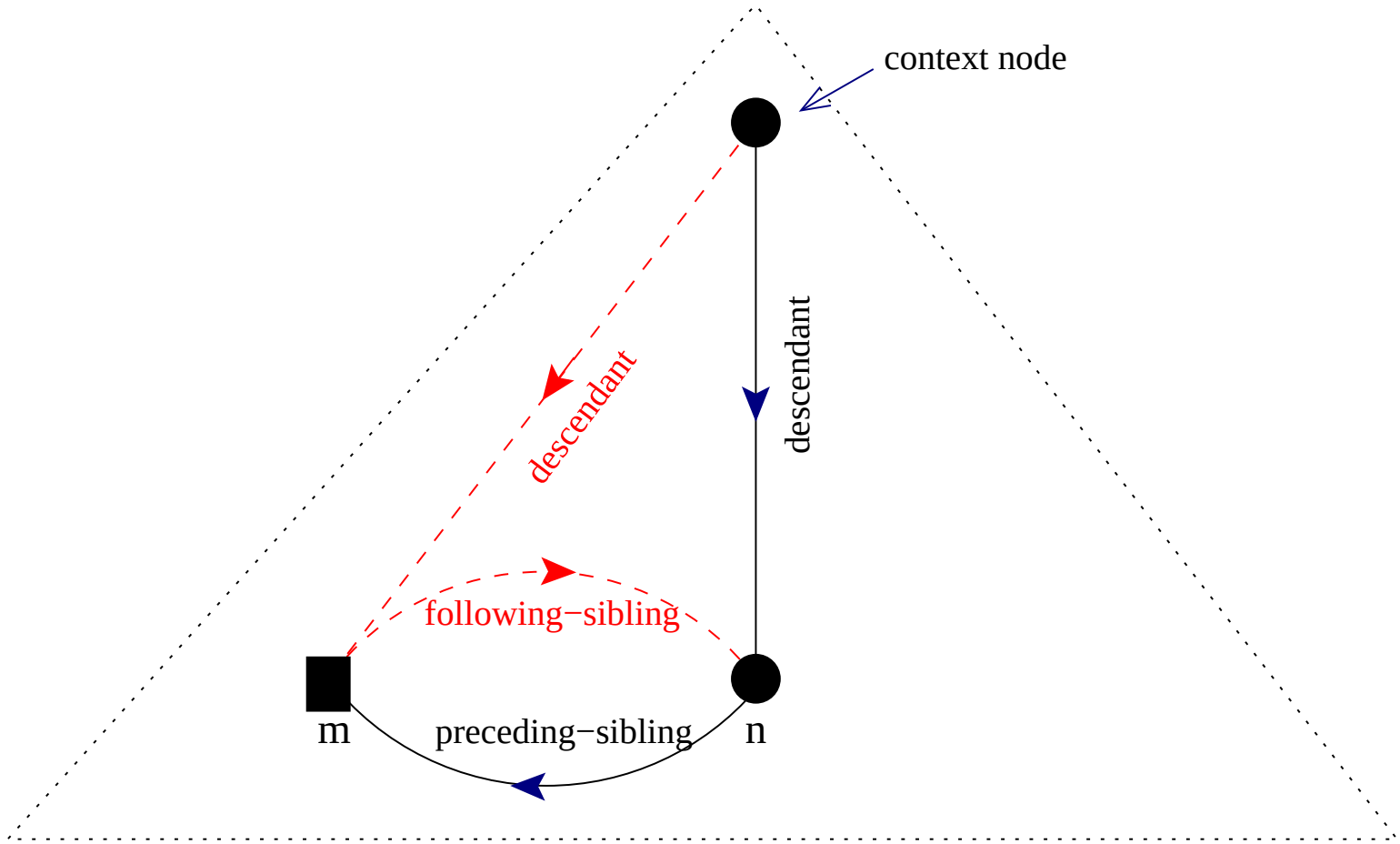


p / descendant :: n / preceding :: m ≡ p [descendant :: n] / preceding :: m

| p / child :: * [following−sibling :: * / descendant−or−self :: n] / descendant−or−self :: m

Specific Equivalences for Preceding-Sibling steps

Consider the interaction between a `descendant` and a `preceding-sibling` step outside qualifiers:



descendant :: n / preceding−sibling :: m ≡ descendant :: m [following−sibling :: n]

Equivalences using Variables

General and Specific Equivalences cover all absolute location paths involving reverse steps, except those having RR joins within qualifiers.

RR join

$p_1\ \theta\ p_2$ where $\theta \in \{==, =\}$, and p_1 and p_2 are **R**elative paths such that at least one of them contains a **R**everse step.

Example

`/descendant::`n`[self::`n` = preceding::`n`]`

Solution

variables from XPath 2.0 or host languages, e.g. XQuery, XSLT.

Example

```
for $m in /descendant::n
   return $m/following::n[$m = self::n]
```

Location Path Rewriting

Each equivalence $p_l \equiv p_r$ gives rise to a rewriting rule $p_l \rightarrow p_r$.

- RuleSet_1 contains the general rules, RuleSet_2 the specific rules

Consider p an absolute location path without qualifiers containing RR joins, $p_l \rightarrow p_r$ a rule either from RuleSet_1 or RuleSet_2 and q the *result* of the application of this rule to p.

Rule application

- If $p = p_l/p'$, then $q = p_r/p'$.
- If $p = p'/p_l/p''$, then $q = p'/p_l/p''$.

Properties of rule application

1. If p contains a reverse step, then a rule from RuleSet_1 and a rule from RuleSet_2 is applicable to p.
2. The result of a rule application to the first reverse step in p is an absolute location path without qualifiers containing RR joins.
3. If q is the result of a rule application to p, then $p \equiv q$.

Location Path Rewriting (II)

Consider an absolute location path p without qualifiers in which RR joins occur. There exists an absolute location path p' with no reverse steps such that $p \equiv p'$.

Using RuleSet$_1$

- p' has a length and can be computed in a time linear in the length of p.
- p' contains as many identity-based joins as reverse steps in p.

Using RuleSet$_2$

- p' has a length and can be computed in a time exponential in the length of p in the worst case.
- p' might contain unions.

Rewriting Examples

Consider a query asking for all `sections` appearing immediately under `titles`, that appear before a `name` and are inside `journals`:

```
/descendant::name/preceding::title[ancestor::journal]/child::section
```

- Using RuleSet_1, the equivalent rewritten path is

```
/descendant::title[following::name == /descendant::name]
[/descendant::journal/descendant::node() == self::node()]/child::section.
```

- Using RuleSet_2, the equivalent rewritten path is

```
/descendant::journal/descendant::title[following::name]/child::section.
```

Future Work

- adding to xPath the **position** functions
- simplifying the XPath location paths obtained as result of rewriting
- further comparison of the two rule sets
- design and implementation of a stream-based XPath processor
  http://www.pms.informatik.uni-muenchen.de/lehre/projekt-diplom-arbeit/streamedxpath.html
- find more in
  http://www.pms.informatik.uni-muenchen.de/publikationen/PMS-FB/PMS-FB-2001-16.pdf