SXPath: a Spatial Extension of XPath

Ermelinda Oro¹, Massimo Ruffolo³, and Steffen Staab²

¹ High Performance Computing and Networking Institute, Italian National Research Council
Via P. Bucci 41/C, Rende (CS), 87036, Italy
² Institute for Computer Science, University of Koblenz
Universitätsstraße 1, PO Box 201 602 56016. Koblenz, Germany email: {oro,ruffolo}@icar.cnr.it
email: staab@uni-koblenz.de

Abstract. We report on a recently introduced extension of XPath, called SXPath, which is a new framework for querying Web documents by considering tree structures as well as spatial relationships between laid out elements. The underlying rationale is that frequently the rendering of tree structures is very involved and undergoing more frequent updates than the resulting layout structure. In this paper, we present the syntax and the semantics of the language that are based on a combination of a spatial algebra with formal descriptions of XPath navigation. Such language is intuitive and general enough to capture most frequent extraction patterns. Moreover, we show that the language maintains polynomial time combined complexity. Practical experiments demonstrate the usability of SXPath. This work is a short version of [11].

1 Introduction

Web designers plan Web pages contents in order to provide visual patterns that help human readers to make sense of document contents. This aspect is particularly evident in Deep Web pages [9], where designers always arrange data records and data items with visual regularity to meet the reading habits of humans. In the past, manual wrapper construction (e.g. [14]), or wrapper induction approaches (e.g. [4,12,16]) have exploited regularities in the underlying document structures, which led to such similar layout, to translate such information into relational or logical structures. However, surveying a large number of real (Deep) Web pages, we have observed that the document structure of current Web pages has become more complicated than ever implying a large conceptual gap between document structure and layout structure. Thus, it has become very difficult: (i) for human and applications aiming at manipulating Web contents (e.g. [5,7,14]), to query the Web by language such as XPath 1.0; (ii) for existing wrapper induction approaches (e.g. [12,16]) to infer the regularity of the structure of deep Web pages by only analyzing the tag structure. Hence, the effectiveness of manual and automated wrapper construction are limited by the
requirement to analyze HTML documents with increasing structural complexity. In the literature, approaches aimed at manipulating Web pages by leveraging the visual arrangement of page contents [7,14], and frameworks for representing and querying multimedia and presentation databases [1,8] have been proposed. However such approaches and frameworks provide limited capabilities in navigating and querying Web documents for information extraction purposes. Therefore, we propose to extend XPath 1.0 by new spatial navigation primitives, namely: (i) Spatial Axes, based on topological [13] and rectangular cardinal relations [10], that allow for selecting document nodes that have a specific spatial relation w.r.t. the context node. (ii) Spatial Position Functions that exploit some spatial orderings among document nodes, and allow for selecting nodes that are in a given spatial position w.r.t. the context node.

This paper is organized as follows. In Sec. 2 we describe SXPath data model, syntax, semantics and complexity issues. Sec. 3 reports notes on the implementation and results of experiments aimed at evaluating processing performances, SXPath usability, and qualitative enhancement in real applications. Sec. 4 concludes the paper. We have several results that, for space reason, we do not include here. We refer the reader to [11] for more details.

2 The SXPath Language

The SXPath language extends the W3C’s XPath 1.0 [15] with spatial capabilities. Intuitive navigational features and querying capabilities of XPath 1.0 are central to most XML-related technologies. For this reason XPath 1.0 has attracted great attention in the computer science research community. The SXPath language adopts the path notation of XPath 1.0 augmented by a user-friendly syntax having a natural semantics that enables spatial querying. In this section, we first define the SXPath data model and describe its new spatial capabilities. Then we provide syntax, semantics, and complexity issues of the language.

2.1 Data Model

In this section we present the SXPath data model, namely Spatial DOM (SDOM). The SDOM considers relations existing among the visual representation of DOM nodes defined as follows.

**Definition 1.** Let n be a node in the DOM of a Web page, the minimum bounding rectangle (MBR) of n is the minimum rectangle r that surrounds the contents of n and has sides parallel to the axes (x and y) of the Cartesian plane. The function mbr(n) returns the rectangle r assigned to a DOM node by the layout engine of a Web browser. We call \( r_x \) and \( r_y \) the segments that are obtained as the projection of r on the x-axis and the y-axis respectively. Then, each side of the rectangle is represented by the segments \( (r_x^-, r_x^+) \) and \( (r_y^-, r_y^+) \), where \( r_x^- \) (resp. \( r_y^- \)) denote the infimum on the x-axis (y-axis) and \( r_x^+ \) (resp. \( r_y^+ \)) denote the supremum on the x-axis (y-axis) of the segments \( r_x \) and \( r_y \).
Considering the function $mbr(n)$ given in Def. 1, a Web page can be modeled as a DOM enriched by spatial relations existing between MBRs. For representing such spatial relations we adopt the Rectangle Algebra (RA) qualitative spatial model [3], which allows for representing all possible relations between rectangles the sides of which are parallel to the axes of some orthogonal basis in a 2-dimensional Euclidean space. RA is a straightforward extension of the standard model for temporal reasoning, the Interval Algebra (IA) [2], to the 2-dimensional case. IA models the relative position between pairs of segments by a set of 13 atomic relations (\(\mathcal{R}_{\text{int}}\)), namely before (\(b\)), meet (\(m\)), overlap (\(o\)), start (\(s\)), during (\(d\)), finish (\(f\)), together with theirs inverses \(\{b^i, m^i, o^i, s^i, d^i, f^i\}\) and the relation equal (\(e\)). Let \(s\) and \(s_1\) be two segments the IA relation \(s \ b \ s_1\) represents that the segment \(s\) is preceded by the segment \(s_1\). Let \(a\) and \(b\) be two rectangles, a RA relation between them is written as \(a \ \rho \ b\) where \(\rho = (\rho_x, \rho_y)\) is a pair of IA relations. The RA relation holds iff the IA relations \(a_x \ \rho_x \ b_x\) and \(a_y \ \rho_y \ b_y\) hold for segments that are obtained as projections of rectangle sides along \(x\) (i.e. \(a_x, b_x\)) and \(y\) (i.e. \(a_y, b_y\)) respectively. The expressiveness of RA covers the modeling of all qualitative spatial relations between two MBRs.

**Definition 2.** SDOM is a node labeled sibling tree that provides orders among nodes, as described in [11], enriched by RA relations. It is described by the following 5-tuple:

\[
\text{SDOM} = h V, R_{\Rightarrow}, R_{\Leftarrow}, A, f \]

where:

- \(V\) is the set of labeled DOM nodes. \(V = V_v \cup V_{nv}\) where \(V_v\) is the set of nodes visualized on screen, and \(V_{nv}\) is the set of nodes that are not visualized.
- \(R_{\Rightarrow}\) is the firstchild relation. Let \(n\) and \(n^0\) be two nodes in \(V\), \(n R_{\Rightarrow} n^0\) holds iff \(n^0\) is the first child of \(n\).
- \(R_{\Leftarrow}\) is the nextsibling relation. Let \(n\) and \(n^0\) be two nodes in \(V\), \(n R_{\Leftarrow} n^0\) holds iff \(n^0\) is the next sibling of \(n\).
- \(A \subseteq V_v \times V_v\) is the set of arcs that represent spatial relations between pairs of nodes visualized on screen.
- Let \(R_{\text{rec}}\) be the set of RA relations, \(f : A \rightarrow R_{\text{rec}}\) is the function that assigns to each element in \(A\) a RA relation in \(R_{\text{rec}}\). So, let \(n\) and \(n^0\) be two nodes in \(V_v\), we have \(a = (n, n^0) \in A\) holds iff \(mbr(n) f(a) mbr(n^0)\).

### 2.2 Spatial Axes

RA relations, stored in the SDOM, represent all qualitative spatial relations between MBRs, but they are too fine grained, verbose and not intuitive for querying. Therefore, for defining SXPath spatial axes we consider the more synthetic and intuitive Rectangular Cardinal Relation (RCR) [10] and Rectangular Connection Calculus (RCC) [13] models. In particular, RCRs express spatial axes that represent directional relations between MBRs. RCRs are computed by
analyzing the 9 regions (cardinal tiles) formed by the projections of the sides of the reference MBR (i.e. \( r \)). The atomic RCRs are: \( \text{belongs to} \) (B), \( \text{South} \) (S), \( \text{SouthWest} \) (SW), \( \text{West} \) (W), \( \text{NorthWest} \) (NW), \( \text{North} \) (N), \( \text{NorthEast} \) (NE), \( \text{East} \) (E), and \( \text{SouthEast} \) (SE). Using the symbol "::" it is possible to express conjunction of atomic RCRs. Furthermore, the three relations inspired by the RCC calculus, namely: \( \text{contained} \) (CD), \( \text{container} \) (CR), and \( \text{equivalent} \) (EQ), allow for expressing spatial axes that represent topological relations between MBRs. For instance, \( r\text{ CD} r_2 \) means that the rectangle \( r_2 \) is spatially contained in the rectangle \( r \). Each spatial axes (expressed by a RCR or a topological relation) corresponds to a set of RA relations computed by means the mapping function \( \mu \) [11].

Like in XPath 1.0, SXPath axes are interpreted binary relations \( \chi \subseteq V \times V \). Let \( \text{self} := \{ h_u, w | u \in V \} \) be the reflexive axis, remaining SXPath axes are partitioned in two sets: \( \Delta_t \) and \( \Delta_s \). The set \( \Delta_t \) contains traditional XPath 1.0 axes (forward, e.g. child, descendant, and reverse, e.g. parent, ancestor) that allow for navigating along the tree structure. They are encoded in terms of their primitive relations (i.e. \( \text{firstchild} \), \( \text{nextsibling} \) and their inverses), as shown in [6]. The set \( \Delta_s \) contains the novel (directional and topological) spatial axes corresponding to the RCRs and Topological Relations that allow for navigating along the spatial RA relations. In the following we formally define spatial axes in terms of their primitive RA relations stored in the SDOM.

**Definition 3.** SXPath spatial axes are interpreted binary relations \( \chi \subseteq V \times V \), of the following form \( \chi = \{ h_u, w | u, w \in V \land \text{mbr}(u) \rho \text{mbr}(w) \land \rho \in \mu(R) \} \). Here, \( R \) is the RCR or topological relation that names the spatial axis relation and \( \mu \) is the mapping function.

### 2.3 Syntax and Semantics

In this section we present the syntax of SXPath and give basic ideas which the language semantics is based on. Like XPath 1.0, the SXPath language allows for selecting sets of SDOM nodes by means of expressions. SXPath expressions have the same structure as the ones in XPath. SXPath extends XPath by means of: (i) A new set of spatial axes that can be used in location steps in the same way as traditional XPath axes. (ii) New node set functions, named spatial position functions, that allow for expressing predicates working on positions of nodes on the plane. These new spatial features enable spatial navigation and querying by exploiting spatial relations and spatial orders stored in the SDOM.

A SXPath location step has the following syntax \( \chi :: t[p_1]...[p_n] \) where: (i) \( \chi \) can be either a traditional XPath axis or a spatial axis. (ii) \( t : \text{L} \cup \text{?text} \to \text{2}^{\text{is the node test function that returns the set of nodes that have a given label. Special labels ? and text identifies all nodes (} t(?) = V \text{) and text nodes respectively. (iii) Predicates can be based on spatial position functions. SXPath expressions return a value from one of the following types: node set, number, string, or boolean. Every expression evaluates relative to a context that extends the context of traditional XPath by considering spatial positions.
For instance, given the Web page Figure 1, a human reader can interpret the spatial proximity of images and nearby strings as a corresponding aggregation of information, namely as the complete record describing the details of a music band profile and its photo.

Fig. 1. A Page of the http://www.lastfm.it/ Web Site

The following XQuery exploits SXPath for extracting details of music bands by exploiting only the DOM nodes of type img and text, and their spatial relations.

Example 1. XQuery 1.0 and SXPath

```xquery
for $img in document ("last-fm.htm")
  1) /CD::img[N|S::img]
     return <music-band>
  2)<name> {$img/E::text[posFromW()=1][posFromN()=1]} </name>
     <similar-bands>
  3){ $img/E::*[posFromW()=1][posFromN()=3][posSpatialNesting()=1]/CD::text }
     </similar-bands>
  </music-band>
```

The spatial location path 1 returns images that form a vertical sequence. The spatial location path $img/E::text$ in pattern 2 and 3 returns nodes that lie on east (spatial axis E) of the context node represented by the variable $img$ (photos of music bands). Among these nodes the predicates select the name of the bands and its similar bands.

2.4 Complexity Issues

This section summarizes the computational complexity results of the SXPath query evaluation problem. We have considered two important fragments: (i) Core SXPath that is the navigational core of SXPath. It is obtained extending Core XPath [6] (the navigational core of XPath 1.0) by spatial axes introduced in Sec. 2.2. (ii) Spatial Wadler Fragment (SWF) that is the spatial extension of the Extended
**Wadler Fragment** (EWF) [6]. It adds to Core XPath positional, logical and arithmetic features. Tab. 2.4 shows that Core SXPath, **Spatial Wadler Fragment** (SWF), and Full SXPath allow polynomial time combined complexity query evaluation with increasing degree of the polynomial. These results are compared with the fragment of XPath 1.0 that they extend. We denote by $D$ the XML document, which has size $\Theta(|V|)$, where $|V|$ is the number of nodes of its SDOM representation. It is noteworthy that the SDOM (see Sec. 2.1) has size $O(|V|^2)$. $|Q|$ is the number of nodes of the parse tree for an input query $Q$.

<table>
<thead>
<tr>
<th></th>
<th>XPath 1.0</th>
<th>SXPath</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>$O(</td>
<td>D</td>
</tr>
<tr>
<td>Time</td>
<td>$O(</td>
<td>D</td>
</tr>
<tr>
<td><strong>Core</strong></td>
<td>$O(</td>
<td>D</td>
</tr>
<tr>
<td><strong>Spatial</strong></td>
<td>$O(</td>
<td>D</td>
</tr>
<tr>
<td><strong>SWF</strong></td>
<td>$O(</td>
<td>D</td>
</tr>
<tr>
<td><strong>Full</strong></td>
<td>$O(</td>
<td>D</td>
</tr>
</tbody>
</table>

**Table 1.** Comparison between complexity bound of SXPath and XPath 1.0 for a XML document $D$ and a query $Q$. 
3 Implementation and Experiments

We have implemented the language in a system that embeds the Mozilla browser\(^1\) and computes the SDOM in real time at each variation of visualization parameters (i.e. screen resolution, browser window size, font type and dimension). In this way for each Web page and visualization condition there is a unique corresponding SDOM that enables the user to query the Web page by considering what s/he sees on the screen.

**System Efficiency.** For evaluating the SXPath system efficiency we have performed experiments that evidence the practical system behavior for both increasing document and query sizes. The runtime requirements for the SDOM construction and query evaluation is polynomial. For evaluating query efficiency we tested the whole system with fixed document sizes \(|D|=1000, |D|=3000, |D|=6000\) and increasing query sizes. Whereas for evaluating data efficiency of the query evaluator, we have used a fixed SWF query having size \(|Q|=167\). Then, we have computed the needed query time for increasing documents sizes from \(|D|=50\) to double the maximum size we found on real-world Web pages, i.e. \(|D|=7500\) nodes. The obtained curves on log log scale have shown that time grows linearly with the query size (and the document size respectively) that indicate polynomial time, as described in [11].

**Language Usability.** Moreover we have performed experiments aimed at assessing the usability of our approach and the enhancements provided by SXPath language over XPath 1.0. For the evaluation we have considered the situation of an expert user aiming at manually developing Web wrappers for Deep and Social Web sites. Experiments have involved ten users who where students well trained in XPath with no experience in SXPath. Each user visualizes and explores Web pages by using the SXPath system. In the experiments we have used a dataset of 125 pages obtained by collecting 5 pages per site from 25 Deep Web sites already exploited for testing wrapper learning approaches [5,12,16]. By carried out experiments we observed that:

1. Modifications in screen resolution and font type do not affect query results, whereas changes in browser window size and font dimension could affect the query result. However, this aspect does not impact SXPath usability because the SXPath system: (i) Embeds the browser and computes the SDOM at each changing of visualization parameters. So, users can query what they see on the screen at each moment. (ii) SXPath queries are stored with visualization parameter settings adopted by the user during the query design process. Thus, when a query is reused on a Web page the embedded browser is set with visualization parameters for which the query has been designed.

2. The language was assessed as easy to learn and quite satisfactory to use.

3. The language is suitable for manual wrapper construction, giving the expert the possibility to look only at the visualized Web page, in comparison to XPath.

\(^1\)https://developer.mozilla.org/en/XULRunner 1.9.2 Release Notes
In fact, by using 2 attempts on average users were able to define a good SXPath query, whereas all the 5 available attempts were not enough for finding a good pure XPath query for all Web pages in the dataset.

4. Manually writing queries in pure XPath is more “complex” in comparison to SXPath because XPath requires the navigation of very intricate DOM structures, whereas SXPath mainly requires to look at the displayed page.

5. Even though the internal tag structure of various Web pages differ strongly (so different pure XPath queries are needed), all users have been able to use almost the same SXPath query for Web sites with similar visual arrangement. This experiment points out that SXPath allows for more general and abstract queries, that are independent from the internal structure of Web pages, in comparison to XPath.

Experiments provide a strong evidence for believing that humans aiming at manually defining Web wrappers and manipulating Web pages, may benefit from using SXPath navigation instead of pure XPath navigation. Moreover, the transportability of SXPath queries from one Web site to the next simplify manual definition of Web wrappers and can also support wrapper induction from sparsely annotated data, while the lack of such transportability observed for pure XPath is detrimental for both manual wrapper definition and wrapper induction. Details and rationales about all performed experiments are given in [11].

4 Conclusion and Future Work

In this paper, we have surveyed recent results about SXPath, the language that extends XPath to include spatial navigation into the query mechanism. We have used spatial algebras to define new navigational primitives and mapped them for query evaluation onto an extension of the XML document object model (DOM), i.e. the SDOM. Thus, we have given a formal model of the extended query language and have evaluated theoretical complexity. The theory has been implemented in a SXPath tool. Empirical evaluation have evidenced practical applicability of SXPath. The language can still be handled efficiently, yet it is easier to use and allows for more general queries than pure XPath. The exploitation of spatial relations among data items perceived from the visual rendering allows for shifting parts of the information extraction problem from low level internal tag structures to the more abstract levels of visual patterns. In the future research, we aim at making SXPath the query language for any Presentation Oriented Documents, such as PDF and PPT in addition to HTML. Moreover, we will investigate the introduction of powerful construct in SXPath in order to automatically define Web wrappers based on visual features.
Acknowledgements

Work done by Steffen Staab was partially funded by German National Science Foundation (DFG) in the project 'Multipla'. Work done by Ermelinda Oro and Massimo Ruffolo was partially funded by Regione Calabria in the projects 'Voucher Ricercatori' and 'EasyPA - ID: 1220000277'.

References