SXPath - Extending XPath towards Spatial Querying on Web Documents

Ermelinda Oro[1] Massimo Ruffolo[1] Steffen Staab[2]

[1]Institute of High Performance Computing and Networking of CNR (ICAR-CNR)
University of Calabria, Italy

[2]Institute for Computer Science, University of Koblenz, Koblenz, Germany

VLDB 2011

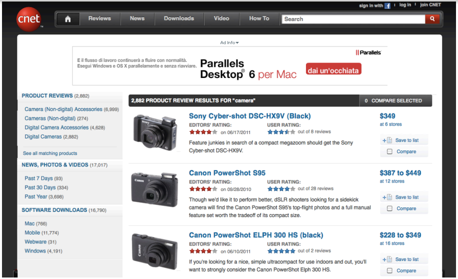
Outline

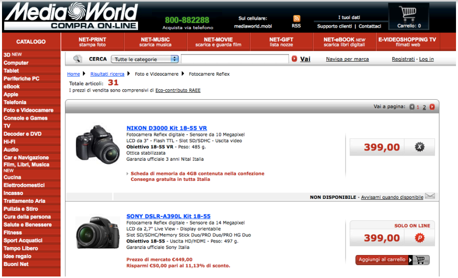
Outline

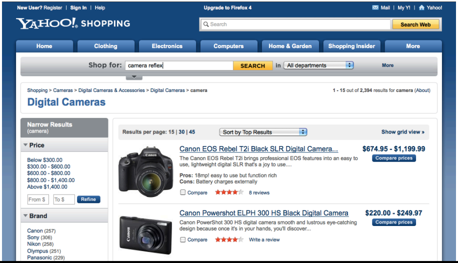
Motivations

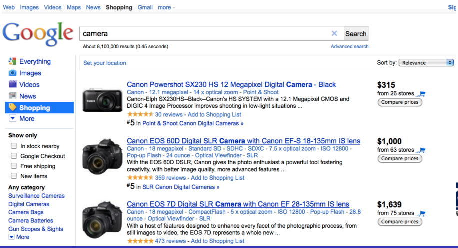
- Users need to access the Web and capture information in many application fields (e.g. business, competitive and military intelligence; content, document and knowledge management)
- Web pages are human oriented. The spatial arrangement of content items in Web pages produces visual cues that help human readers to make sense of document contents
- Well founded and known query formalisms, such as XPath and XQuery, do not consider spatial arrangements in querying Web pages

Presentation-Oriented Documents







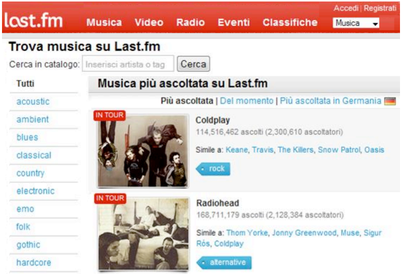


?

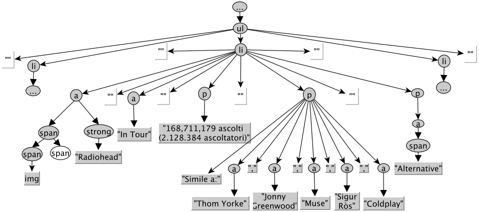
Presentation-Oriented Documents

HTML DOM allows only site-centric extraction

A Web Page



Document Object Model



Spatial arrangements are rarely explicit and frequently hidden in complex nestings of layout elements corresponding to intricate tree structures that are conceptually difficult to query

Outline

State of the Art

- Web Query language
  - *XPath* 1.0 and *XQuery* 1.0 represent well founded and known web query languages having very intuitive navigational features, but the intricate DOM structure makes difficult to pose queries
- Visual languages
  - *Spatial Graph Grammars* [Kong et al.] are quite complex in term of both usability and efficiency
  - *Algebras* for creating and querying multimedia interactive presentations (e.g. ppt) [Subrahmanian et al.] require database for multimedia presentation should be created for the whole Web
- Web wrapper induction exploiting visual interface [Gottlob et al.] [Sahuguet et al.]
  - generate XPath location paths of DOM nodes
  - can benefit from using Spatial XPath

Outline

1. Introduction
   - Motivations
   - State of the Art
   - SXPath Language
2. SXPath
   - Spatial Data Model
   - Syntax and Semantics
   - Complexity Issues
   - Implementation Issues and Experiments
3. Conclusions and Future Work

Extending XPath towards Spatial Querying

- As extension of XPath 1.0, *Spatial XPath* (SXPath):
  - adopts the intuitive path notation: `/axis::nodetest [pred`$_1$`]*`
  - adds new *spatial axes* and new *spatial position functions*
  - has a natural semantics that enables spatial querying
  - maintains polynomial time combined complexity
- Advantages:
  - it is easy to learn and easier to use than pure XPath on Web pages
  - it is more tolerant to modifications of the internal structure of Web pages
  - it enables users to spatial query Web documents on the base of what they see on the document
  - it is capable to provide benefits to some current Web contents manipulation and wrapper learning approaches

Presentation-Oriented Documents

A Web Page from the lastfm Web site (http://www.lastfm.it/)
Acquiring a music band profile: *A music band photo that has at east its descriptive information*

Example 1

Exploiting XPath

```
for $li in document
("last-fm.htm")
(1.1) //div[@id='content'] //ul/li
return
  <music-band>
(1.2)  <name>
         {$ li / a / strong / text()}
       </name>
  ...
  </music-band>
```

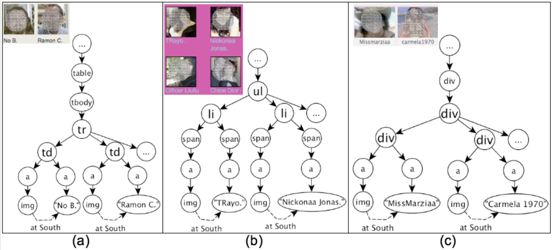
Exploiting SXPath

```
for $li in document
("last-fm.htm")
(2.1) / CD::img [N|S::img]
return
  <music-band>
(2.2)  <name>
         {$img/ E::text [N,1]}
       </name>
  ...
  </music-band>
```

Example 2

Acquiring friend lists from different social networks pages represented as couples `<photo, name>`.

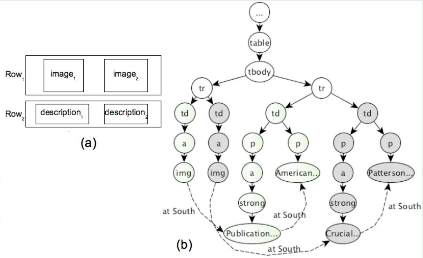
Friend lists from different social networks pages (a) Bebo (b) Care (c) Netlog.




```
for $img in document ("http://www.bebo.com/friendlist.html")
 (3.1) //img[ N|S|E|W::img  ]
   return
    <friend>
 (3.2) <photo> {$img} </photo>
 (3.3) <name> { $img/ S :: text() [N,1]} </name>
    </friend>
```

Example 2

- A single data record can be split in different sub-trees
- Wrapper induction techniques like DEPTA [Zhai et al.] recognize data records when they are encoded in the DOM as consecutive similar subtrees




```
for $img in document ("http://www.bebo.com/friendlist.html")
 (3.1) //img[ N|S|E|W::img  ]
   return
    <friend>
 (3.2) <photo> {$img} </photo>
 (3.3) <name> { $img/ S :: text() [N,1]} </name>
    </friend>
```

Outline

1. Introduction
   - Motivations
   - State of the Art
   - SXPath Language
2. SXPath
   - Spatial Data Model
   - Syntax and Semantics
   - Complexity Issues
   - Implementation Issues and Experiments
3. Conclusions and Future Work

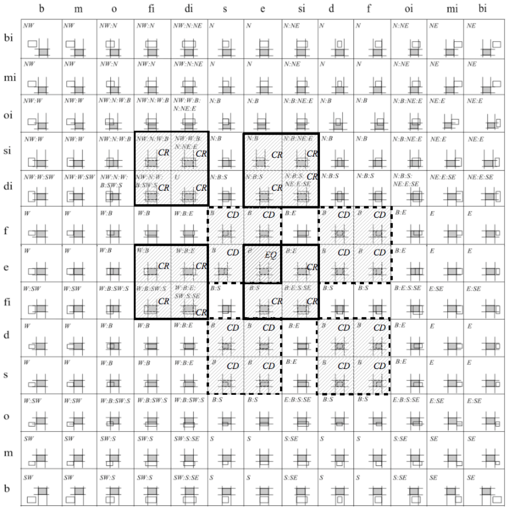
Spatial Data Model

- The Document Object Model (DOM) is the internal rapresentation of markup languages (XML, HTML)
- The tree-based structures of XML are often not convenient and not expressive enough in order to represent spatial arrangements
- The spatial arrangements are rarely explicit and frequently hidden into intricate tree structures that are conceptually difficult to query

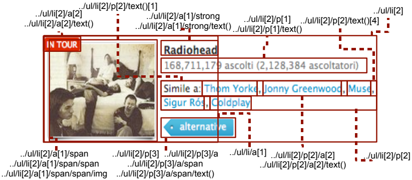
Spatial Relations among Nodes

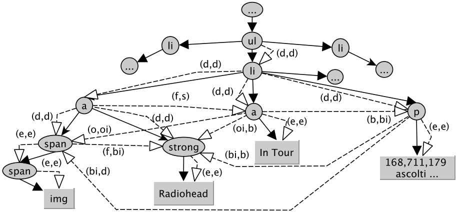
- The *Rectangular Algebra* (RA) [Balbiani et al.] extends Allen's temporal interval algebra (IA) to the 2-dimensional case
- RA is a very fine-grained and expressive model that allows the computations of spatial relations as well as algebraic optimizations
- RA holds many important properties (e.g. invertibility) that allows for optimized query evaluation

Relation	Symbol	Meaning	Inverse
before	b	s_1 s	bi
meets	m	s_1 s	mi
overlaps	o	s_1 s	oi
starts	s	s_1 s	si
during	d	s_1 s	di
finish	f	s_1 s	fi
equals	e	s_1 s	e



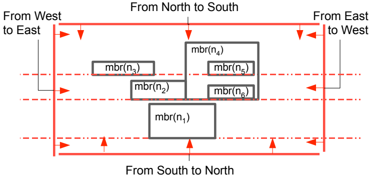
Spatial DOM (SDOM)





The SDOM extends the Document Object Model (DOM) by:

- RA relations existing between pairs of nodes visualized on screen
- spatial orders among nodes



$$n_1 \leqslant_{\uparrow} n_2 =_{\uparrow} n_4 =_{\uparrow} n_6 \leqslant_{\uparrow} n_3 =_{\uparrow} n_5$$

The Spatial DOM (SDOM)

Definition

SDOM is a node labeled sibling ordered tree enriched by RA relations

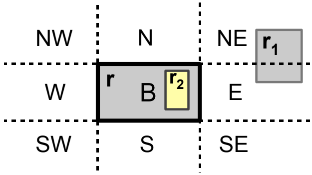
$$SDOM = \langle V, R_{\Downarrow}, R_{\Rightarrow}, A, f_s \rangle$$

where:

- V is the set of labeled DOM nodes. $V = V_v \cup V_{nv}$
- $R_{\Downarrow}$ is the *firstchild* relation
- $R_{\Rightarrow}$ is the *nextsibling* relation
- $A \subseteq V_v \times V_v$
- Let R_{rec} be the set of RA relations $f_s : A \rightarrow R_{rec}$

Qualitative Spatial Models

Rectangular cardinal relations



Topological relations, inspired by the Region Connection Calculus model:

- contained (CD)
- container (CR)
- equivalent (EQ)

Example

- r E:NE r_1
- r B r_2

Example

- r CD r_2
- r_2 CR r

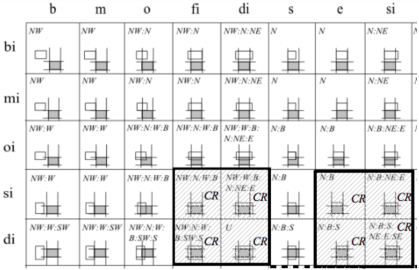
Spatial Navigation Axes

- As in XPath, SXPath primitives for navigating the SDOM are called axes
- Axes are interpreted binary relations $\chi \subseteq V \times V$. Let $self := \{\langle u, u\rangle | u \in V\}$ be the reflexive axis, remaining SXPath axes are partitioned in two sets: Δ_t and Δ_s
  - $\Delta_t = \{$*self, child, parent, descendant, descendant-or-self, ancestor, ancestor-or-self, following-sibling, preceding-sibling, following, preceding*$\}$ contains traditional XPath 1.0 axes
  - Δ_s is the set of novel spatial axes expressed by: basic and disjunctive RCRs and topological relations that are more intuitive than RA relations

Spatial Navigation Axes

Definition

SXPath spatial axes are interpreted binary relations $\chi_s \subseteq V_v \times V_v$ of the following form $\chi_s = \{\langle u, w\rangle | u, w \in V_v \wedge u\ \rho\ w \wedge \rho \in \mu(R)\}$. Where R is the RC or Topological Relation that names the spatial axis and μ is the mapping function



Outline

Syntax

- SXPath expressions have the same structure as the ones in XPath
  - Location paths are sequences of location steps separated by the navigation operator "/".
  - A locstep is `axis ::nodetest [pred`$_1$`]...[pred`$_n$`]`
- We enrich XPath 1.0 by
  - The new set of *spatial axes*
  - *Spatial position functions*
- Specific subsets of the language with attractive properties have been characterized for XPath 1.0 [4, 6]
  - *Core XPath* ⇒ *Core SXPath*
  - *Wadler Fragment*(WF) ⇒ *Spatial WF*

Semantics

- The main structural feature of SXPath are *expressions*, that return a value from one of the following four types: *node set*, *number*, *string*, or *Boolean*
- Every expression evaluates relative to a *context*, concept introduced by Wadler

Definition (Context)

The *context* is the following 12-tuple:

$$\vec{c} = \langle n, p_{<_{doc}}, s_{<_{doc}}, p_{\leqslant_\uparrow}, s_{\leqslant_\uparrow}, p_{\leqslant_\rightarrow}, s_{\leqslant_\rightarrow}, p_{\leqslant_\downarrow}, s_{\leqslant_\downarrow}, p_{\leqslant_\leftarrow}, s_{\leqslant_\leftarrow}, p_{\leqslant_t} \rangle$$

where:

- n is a *context node*
- $p_{\leqslant_z}$ are the *context positions* w.r.t. orders
- $s_{\leqslant_z}$ are the *context sizes*

Semantics

Definition (Location path semantics)

Let π, π_1, π_2 be location paths, let *locstep* be a location step over an axis χ, let *bexpr* be a boolean expression and let *n* be a context node, *P*: *LocationPath* → *node* → *nodeset* is defined as follows:

$P[\![/\pi]\!](n) := P[\![\pi]\!](root)$

$P[\![\pi_1/\pi_2]\!](n) := \{n_2 | n_1 \in P[\![\pi_1]\!](n) \wedge n_2 \in P[\![\pi_2]\!](n_1)\}$

$P[\![\pi_1|\pi_2]\!](n) := P[\![\pi_1]\!](n) \cup P[\![\pi_2]\!](n)$

$P[\![axis :: t]\!](n) := \{n' \mid [\![\mathbf{axis}]\!](n, n')\} \cap T(t)$

$P[\![locstep[bexpr]]\!](n) := \{n' \mid \vec{W} = P[\![locstep]\!](n) \wedge n' \in \vec{W} \wedge \varepsilon[\![bexpr]\!](\vec{c_{n'}}) = true \wedge$
$\quad \vec{c_{n'}} := \langle n', idx_\chi(n', \vec{W}), |\vec{W}|, pidx_{\leqslant_\uparrow}(n', \vec{W}) plast_{\leqslant_\uparrow}(\vec{W}), pidx_{\leqslant_\rightarrow}(n', \vec{W}), plast_{\leqslant_\rightarrow}(\vec{W}),$
$\quad pidx_{\leqslant_\downarrow}(n', \vec{W}), plast_{\leqslant_\downarrow}(\vec{W}), pidx_{\leqslant_\leftarrow}(n', \vec{W}), plast_{\leqslant_\leftarrow}(\vec{W}), pidx_{\leqslant_t}(n', \vec{W})\rangle\}$

The semantics of spatial axis is given in terms of spatial relations among nodes
$[\![$ **spatialAxis** $]\!]$**:=**$\{(n, n') \mid mbr(n)\ \rho\ \ mbr(n') \wedge \rho = \mu(spatialAxis)\}$

Semantics

Definition (Semantics of SXPath)

$$\varepsilon : \mathit{SXPathExpression} \to \mathbf{C} \to \mathit{SXPathType}$$

$$\varepsilon[\![\pi]\!](\vec{c}) := P[\![\pi]\!](n)$$

$$\varepsilon[\![position()]\!](\vec{c}) := p_{<_{doc}} \qquad \varepsilon[\![last()]\!](\vec{c}) := s_{<_{doc}}$$

$$\varepsilon[\![posFromN()]\!](\vec{c}) := p_{\leqslant_{\downarrow}} \qquad \varepsilon[\![lastFromN()]\!](\vec{c}) := s_{\leqslant_{\downarrow}}$$

$$\varepsilon[\![posFromS()]\!](\vec{c}) := p_{\leqslant_{\uparrow}} \qquad \varepsilon[\![lastFromS()]\!](\vec{c}) := s_{\leqslant_{\uparrow}}$$

$$\varepsilon[\![posFromW()]\!](\vec{c}) := p_{\leqslant_{\rightarrow}} \qquad \varepsilon[\![lastFromW()]\!](\vec{c}) := s_{\leqslant_{\rightarrow}}$$

$$\varepsilon[\![posFromE()]\!](\vec{c}) := p_{\leqslant_{\leftarrow}} \qquad \varepsilon[\![lastFromE()]\!](\vec{c}) := s_{\leqslant_{\leftarrow}}$$

$$\varepsilon[\![posSpatialNesting()]\!](\vec{c}) := p_{t}$$

$$\varepsilon[\![Op(e_1, ..., e_m)]\!](\vec{c}) := F[\![Op]\!](\varepsilon[\![e_1]\!](\vec{c}), \ldots, \varepsilon[\![e_m]\!](\vec{c}))$$

$F[\![$ *RelOp*: num $\times$ num $\to$ bool $]\!]$ (i_1, i_2) ::= i_1 *RelOp* i_2
$F[\![$ constant number i: $\to$ num $]\!]()$ ::= i
...

Outline

1. Introduction
   - Motivations
   - State of the Art
   - SXPath Language
2. SXPath
   - Spatial Data Model
   - Syntax and Semantics
   - Complexity Issues
   - Implementation Issues and Experiments
3. Conclusions and Future Work

Core SXPath Complexity

Theorem (Core SXPath Combined Complexity)

Core SXPath queries can be evaluated in time $O(|D|^2 * |Q|)$ where $|D|$ is the size of the XML document, and $|Q|$ is the size of the query Q

- **Proof Sketch** There are $O(|V_v|^2)$ many spatial relations to be considered in addition to the $O(|V|)$ many relations of the DOM incurring a higher polynomial worst case complexity

SWF and Full SXPath Complexity

Theorem (Spatial WF Combined Complexity)

time $O(max(|D|^3 * |Q|, |D|^2 * |Q|^2))$ *and space* $O(|D|^2 * |Q|^2)$*, where D is the XML document, and Q is a SWF query.*

Theorem (Full XPath Combined Complexity)

time $O(|D|^4 * |Q|^2)$ *and space* $O(|D|^2 * |Q|^2)$*, where D is the XML document, and Q is a Full SXPath query.*

- In order to obtain a polynomial-time combined complexity bound for SXPath query evaluation we use dynamic programming adopting the *Context-Value Table* (CV-Table) principle introduced by Gottlob et al.
- Position and size are computed on demand, we compute all spatial position functions in a loop for all pairs previous\current nodes
- Full SXPath computational costs are dominated by String Operations belonging to XPath 1.0
- In SWF the computation of spatial ordering generates a higher polynomial worst case than XPath 1.0

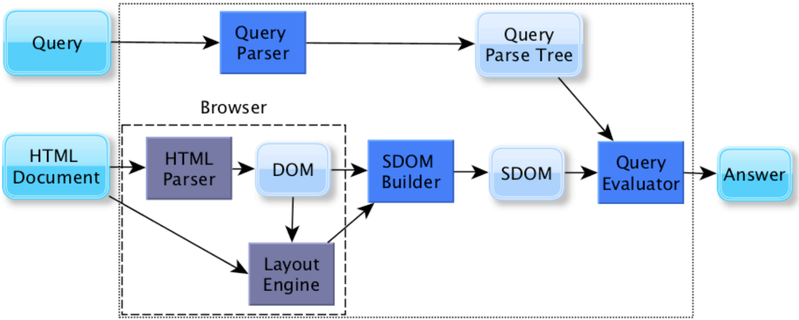
Complexity Results

Comparison between complexity bound of SXPath and XPath 1.0 for a XML document D and a query Q

	XPath 1.0		SXPath	
Space Time	Core	$O(\|D\| * \|Q\|)$ $O(\|D\| * \|Q\|)$	Spatial Core	$O(\|D\|^2 * \|Q\|)$ $O(\|D\|^2 * \|Q\|)$
Space Time	EWF	$O(\|D\| * \|Q\|^2)$ $O(\|D\|^2 * \|Q\|^2)$	SWF	$O(\|D\|^2 * \|Q\|^2)$ $O(max(\|D\|^3 * \|Q\|, \|D\|^2 * \|Q\|^2))$
Space Time	Full Xpath 1.0	$O(\|D\|^2 * \|Q\|^2)$ $O(\|D\|^4 * \|Q\|^2)$	Full SXPath	$O(\|D\|^2 * \|Q\|^2)$ $O(\|D\|^4 * \|Q\|^2)$

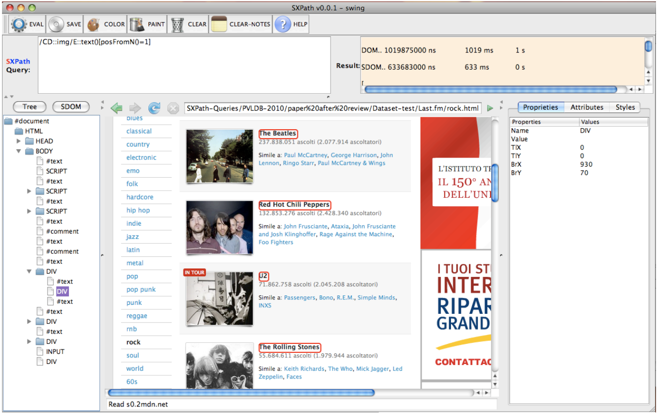
Outline

The SXPath System



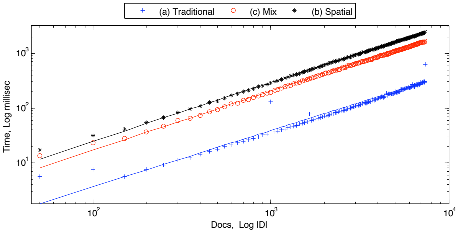
The SXPath System

GUI that supports Spatial Querying

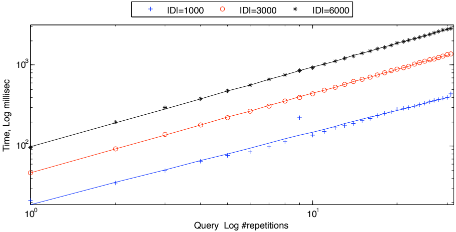


Results of Experiments

Data Efficiency of SXPath Query Evaluation



Query Efficiency of SXPath Query Evaluation



The curves grows linear on log-log scale indicating the polynomial growth

Results of Experiments

Evaluation of the Effort Needed for Learning and Applying SXPath

- We have defined the user task "identify product data records and extract product names and prices" from the Web site *www.bol.de*
- We have asked users to learn the SXPath language and complete the task by writing a sound and complete SXPath query looking only at the visualized Web page
- We have asked users to answer a questionnaire based on the seven-item Likert scale: very easy/satisfactory (3) ... very difficult/unsatisfactory (-3)

#user	Time (min)	Easiness/ Difficulty	Satisfaction/ Unsatisfaction	#attempts	
				name	price
1	75	2	0	7	6
2	45	3	2	4	2
3	65	1	1	5	4
4	40	2	1	2	3
5	50	3	2	4	4
6	30	3	3	2	1
7	125	-1	-1	9	8
8	50	2	1	3	4
9	35	3	2	2	2
10	55	2	1	5	2
Average	**57**	**2**	**1.2**	**4.3**	**3.6**
σ	**26**	**1.18**	**1.1**	**2.2**	**2**

Results of Experiments

Usability Evaluation of SXPath on Deep Web Pages

We have asked users to perform the extraction task “identify product data records and extract product names and prices” for each Web site in the dataset

- only by looking at the displayed Web pages by using at the most 5 attempts
- looking at both visualized Web pages and internal page structures (i.e. DOM and SDOM), by using at the most 10 minutes
- by applying the same location path for different Web sites in the dataset having the same visual pattern. We have observed that it is possible to use the same sound and complete spatial location path for Web sites having the same visual pattern. Instead, different XPath location paths are needed

Considering a set of Deep Web Sites	Querying Without DOM/SDOM						Querying With DOM/SDOM					
	SXPath			XPath			SXPath		Abs. XPath		Rel. XPath	
	Cr.	Wr.	Att.	Cr.	Wr.	Att.	Att.	Steps	Att.	Steps	Att.	Steps
Average			2			5	2.7	5.3	4.2	18.9	4	6.6
Total	2535	27.3/6		2506	3459.5/35							
Recall	100%			99%								
Precision	99%			42%								

Conclusions and Future Work

- We have extended XPath to include spatial navigation into the query mechanism
- The SDOM extends DOM for describing relationships between data entities
- SXPath query language is a stepping stone for future work on extracting information from presentation-oriented documents. It could be used and extended for
  - querying other *presentation-oriented documents* (e.g. PDF, Doc, etc.) or multimedia documents
  - recognizing and extracting ontology objects
  - automatically learning of wrappers and learning of ontology instances [Staab et Al.] by exploiting spatial patterns
  - navigating and accessing Deep Web data sources and dynamic components


Thank you

For Further Reading I

- S. Adali, M. L. Sapino, and V. S. Subrahmanian.
  An algebra for creating and querying multimedia presentations.
  Multimedia Syst., 8(3):212–230, 2000.

- P. Balbiani, J.-F. Condotta, and L. F. d. Cerro.
  A model for reasoning about bidimensional temporal relations.
  In *Proc. of KR-2008*, pages 124–130, 1998.

- R. Baumgartner, S. Flesca, and G. Gottlob.
  Visual web information extraction with lixto.
  In *VLDB*, pages 119–128, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

For Further Reading II

- M. Benedikt and C. Koch.
  Xpath leashed.
  ACM Comput. Surv., 41(1):1–54, 2008.
- G. Gottlob, C. Koch, and R. Pichler.
  Efficient algorithms for processing xpath queries.
  In *VLDB*, pages 95–106, 2002.
- G. Gottlob, C. Koch, R. Pichler, and L. Segoufin.
  The complexity of xpath query evaluation and xml typing.
  J. ACM, 52(2):284–335, 2005.
- J. Kong, K. Zhang, and X. Zeng.
  Spatial graph grammars for graphical user interfaces.
  ACM Trans. Comput.-Hum. Interact., 13(2):268–307, 2006.

For Further Reading III

S. Mir, S. Staab, and I. Rojas.
Unsupervised approach for acquiring ontologies and rdf data from online life science databases.
In *ESWC*, 2010.

A. Sahuguet and F. Azavant.
Building intelligent web applications using lightweight wrappers.
DKE, 36(3):283–316, 2001.

P. Wadler.
Two semantics for xpath.
Draft: http://homepages.inf.ed.ac.uk/~wadler/papers/xpath-semantics, 2000.

For Further Reading IV

Y. Zhai and B. Liu.
Extracting web data using instance-based learning.
In *WWW*, pages 113–132, 2007.

Example 2

Acquiring the table in the document as a set of triples of the form `<row-header, column-header, value>`.

Direct Energy Content [TJ]	1994	1996	1998	2000	2004	2005	2006	2007	Change '94 - '07
Total Gross Electricity Production	**144 708**	**192 879**	**147 998**	**129 776**	**145 583**	**130 468**	**164 199**	**140 964**	**-2.6%**
Oil	9 547	20 808	17 906	15 964	5 881	4 933	5 811	4 616	-51.7%
- Orimulsion	-	14 495	12 890	13 467	7	-	-	-	•
Natural Gas	8 206	20 442	29 260	31 589	35 807	31 606	33 903	24 886	203%
Coal	119 844	142 795	85 151	60 022	67 232	55 665	88 439	71 631	-40.2%
Surplus Heat	-	123	136	139	40	-	-	-	•
Waste, non-renewable	463	610	702	994	1 163	1 459	1 472	1 416	206%
Renewable Energy	**6 647**	**8 101**	**14 844**	**21 068**	**35 459**	**36 805**	**34 574**	**38 415**	**478%**
Solar Energy	0	1	1	4	7	8	8	9	3 017%
Wind Power	4 093	4 417	10 152	15 268	23 699	23 810	21 989	25 823	531%
Hydro Power	117	69	98	109	95	81	84	101	-14.1%
Biomass	2 116	3 207	3 911	4 936	10 646	11 889	11 517	11 504	444%
- Straw	293	748	960	654	3 057	3 088	3 359	3 185	988%
- Wood	429	340	512	828	3 546	3 730	3 041	3 398	691%
- Waste, renewable	1 393	2 120	2 439	3 454	4 043	5 071	5 117	4 921	253%
Biogas	321	407	682	751	1 013	1 017	976	978	205%

```
 for $rh in document ("table.pdf")
(2.1) //text [not(W::*)]
   return
 <table-triples>
   {
   for $ch at $j in document ("table.pdf")
(2.2) //text [not(N::*)]
     <row-header>
(2.3) {$rh}
    </row-header>
    <column-header>
(2.4) {$ch}
    </column-header>
    <value>
(2.5) {$rh/E::text [W,$j]}
    </value>
}
</table-triples>
```

Core SXPath

Definition

The syntax of Core SXPath is defined by the following EBNF grammar

```
locpath     ::= '/' locpath | locpath '/' locpath |
                locpath '|' locpath | locstep.
locstep     ::= axis '::'  t | locstep '[' bexpr ']'
bexpr       ::= bexpr 'and' bexpr | bexpr 'or' bexpr|
                'not(' bexpr ')' | locpath.
axis        ::= xpathAxis | spatialAxis.
xpathaxis   ::= 'self' | 'child' | 'parent' |
                'descendant' | 'descendant-or-self' |
                'ancestor' | 'ancestor-or-self' |
                'following' | 'following-sibling' |
                'preceding' | 'preceding-sibling'.
spatialAxis::= topAxis | dirAxis.
topAxis     ::= 'EQ' | 'CD' | 'CR'.
dirAxis     ::= 'B' | ··· | 'U'.
```

Spatial Wadler Fragment

Definition

The syntax of the SWF-Queries is defined by the Core SXPath grammar with the following extensions.

```
expr        ::= locpath | bexpr | nexpr
dirAxis     ::= 'B' | ··· | 'U' | disjDirAxis.
bexpr       ::= bexpr 'and' bexpr | bexpr 'or' bexpr|
                'not(' bexpr ')' | nexpr relop nexpr |
                sexpr relop sexpr | locpath |
                locpath relop sexpr |
                locpath relop number.
nexpr       ::= number | nexpr arithop nexpr.
                'position()'|'last()'|'posFromS()'|'lastFromS()' |
                'posFromN()'|'lastFromN()'|'posFromW()'|'lastFromW()'
                |'posFromE()'|'lastFromE()'| 'posSpatialNesting()'
sexpr       ::= string.
arithop     ::= '+' | '−' | '*' | 'div' | 'mod'.
relop       ::= '=' | '!=' | '<' | '<=' | '>' | '>='.
```

Input: A set of nodes Γ and an axis $\chi \in \Delta$
Output: $\chi(\Gamma)$
Method: $eval_{\chi}(\Gamma)$

(1.1) **function** $eval_{self}(\Gamma) := \Gamma$.
(1.2) **function** $eval_{\chi_t}(\Gamma) := eval_{E(\chi_t)}(\Gamma)$.
(1.3) **function** $eval_{\chi_s}(\Gamma) := eval_{\{\rho_i | \rho_i \in \mu(\chi_s)\}}(\Gamma)$.
(1.4) **function** $eval_{\chi_s^{-1}}(\Gamma) := eval_{\{\rho_i^{-1} | \rho_i \in \mu(\chi_s)\}}(\Gamma)$.
(1.5) **function** $eval_{\varrho}(\Gamma)$ **begin**
(1.6) $\Gamma' := \varnothing$;
(1.7) **foreach** $u \in \Gamma \cap u \in V_v$ **do**
(1.8) **foreach** $\rho_i \in \varrho$ **do**
(1.9) $\Gamma' := \Gamma' \cup_{set} f_{\rho_i}(u)$ **od od**
(1.10) **return** Γ'**end.**

(Location step evaluation algorithm)

Input: A set of nodes Γ and a location step $e = \chi :: \tau[e_1]\ldots[e_m]$

Output: $P[\![e]\!](\Gamma)$

Method: $eval(e, \Gamma)$ **begin**

(2.1) $Res := \varnothing$

(2.2) $W := \chi(\Gamma) \cap T(\tau)$;

(2.3) **for** each $u \in \Gamma$ **do**

(2.4) $W' := \{w \mid w \in W \wedge u \; \chi \; w\}$

(2.5) **for** each e_i with $1 \leqslant i \leqslant m$ (in ascending order) **do**

(2.6) $\vec{W} := layering(W')$

(2.7) $W' := \{w \mid w \in \vec{W} \wedge \varepsilon[\![e_i]\!](\vec{c_w}) = true \wedge$

$\vec{c_w} := \langle w, idx_\chi(w, \vec{W}), |\vec{W}|, pidx_{\leqslant_\uparrow}(w, \vec{W}), plast_{\leqslant_\uparrow}(\vec{W}),$

$pidx_{\leqslant_\rightarrow}(w, \vec{W}), plast_{\leqslant_\rightarrow}(\vec{W}), pidx_{\leqslant_\downarrow}(w, \vec{W}), plast_{\leqslant_\downarrow}(\vec{W}),$

$pidx_{\leqslant_\leftarrow}(w, \vec{W}), plast_{\leqslant_\leftarrow}(\vec{W}), pidx_{\leqslant_t}(w, \vec{W})\rangle\}$

od

(2.8) $Res := Res \cup W'$

od

(2.9) **return** *Res* **end**;