

A Graphical XML Query Language*

S. Flesca, F. Furfaro, S. Greco

DEIS - Università della Calabria - 87030 Rende - Italy

{flesca, furfaro, greco}@si.deis.unical.it

In this poster we informally present the \mathcal{XGL} query language. The main features of the language are described by means of two queries over the below document, named "bib.xml", and graphically described in Fig. 1.

```
<bib>
  <book year="1997">
    <title> A First Course in Database
      Systems </title>
    <author> Ullman </author>
    <author> Widom </author>
    <publisher> Prentice-Hall </publisher>
  </book>
  ...
</bib>
```

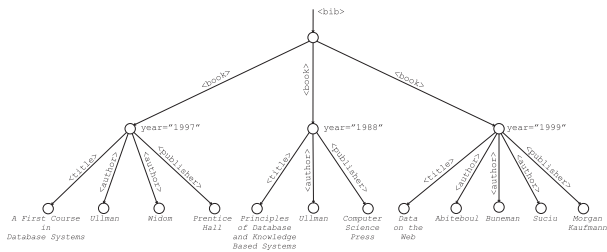


Figure 1: XML graph

The structure of a \mathcal{XGL} query consists of two parts: one querying and one constructing XML data graphs. The querying part is defined by means of a simplified form of graph grammar, which is made user-friendly by adding some syntactic simplifications to standard graph grammars.

The constructing part is defined by a graph which describes the structure and the content of the document which has to be created.

The querying and constructing parts are correlated by means of variables, which are defined in the querying part (where it is specified what kind of information each variable identifies) and then used in the constructing part (where variables refer to the extracted information).

In the extraction of graphs, graph grammars are coupled with first order formulas on such variables, in order to express conditions on data and filter them. In particular, every production rule is associated to a (possibly empty) first order formula which states under which conditions (regarding the data contained in the source graph) the rule can be applied.

*Work partially supported by MURST grants under the projects "Data-X" and "D2I". The third author is also supported by ISI-CNR.

Also the constructing rule defining the output graph is associated to a first order formula, which filters the extracted information and defines how to re-assemble it.

In the representation of graphs of both the querying and constructing parts, the language provides shortcuts associated to nodes and arcs. In particular, in order to identify paths in the source graph during the extraction phase, arcs may be labeled with regular expressions defined on a vocabulary of tags. Nodes may be marked with the symbol '+': such a marked node (called *grouping node*) represents a (possibly empty) set of nodes matching one or more nodes of the input graph.

Like most of the query languages for XML, \mathcal{XGL} queries consist of a "WHERE" clause defining the querying part (extraction of a subgraph) and a "CONSTRUCT" clause, which defines the structure and the content of the document that has to be returned.

Example 1 Construct a document containing for each author the titles of the books he has written

WHERE S IN "bib.xml" CONSTRUCT T

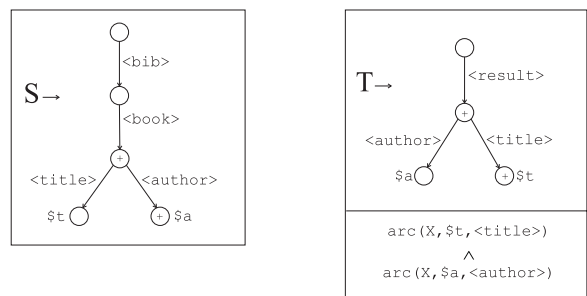


Figure 2: Query example

The rule **S** in Fig. 2 extracts for each book the title and the set of all of its authors. The symbol "+" inside the node ending the arc with tag <book> means that we are interested in all books, and the same symbol marking the nodes at the end of the arc with tag <author> means that we want to collect, for each book, all of its authors.

The rule **T** re-structures the extracted data. In particular, by putting the symbol "+" inside the node ending the arc marked with <title>, we collect for each author all the titles of the books he has written.