1 XML: Model, Schemas, Types, Logics, and Queries

Nils Klarlund¹, Thomas Schwentick², and Dan Suciu³

¹ AT&T Labs – Research, USA
   email: klarlund@research.att.com
² University of Marburg, Germany
   email: tick@Mathematik.Uni-Marburg.de
³ University of Washington, USA
   email: suicu@cs.washington.edu

1.1 Introduction

It is intriguing that something as bland as a syntax for trees has become one of the leading buzzwords of the Internet era. XML (eXtended Markup Language) is the name of this notation, which evolved from document processing needs. This mystery is deepened by the apparent antiquity of the central idea — that linear syntax can represent trees. S-expressions, invented some 40 years ago, are at least as general, but never became the universally accepted foundation for data interchange and programming, despite their enormous influence in programming language theory and artificial intelligence research. Additionally, trees have been studied intensively in computer science ever since, so we might suspect that the reason for the excitement is simply that practitioners are catching up with methods of abstraction and representation via trees that are well known in academia.

In this chapter, we shall see that the suspicion is easily dispelled. We look at techniques now used in practice for dealing with XML trees, and we note how they depart from old-fashioned uses. Because trees are objects that are very complicated to manipulate directly through pointer updates, declarative techniques are becoming increasingly important, especially when it comes to exploring, mining, and constructing tree-shaped data. In particular, we will contrast conventional concepts of database theory such as relational calculus with that of more procedural notations for trees. We explore why the essential problem of locating data in trees is intimately linked with tree automata and decidable logics, somewhat in parallel to the link between query algebras and first-order logic in relational database theory. So, we shall see why logic and automata create interesting new research opportunities.

1.1.1 XML: An Important Practical Paradigm

XML [28] is a convention for representing labeled or tagged trees as a text, that is, as a sequence of characters. In essence, the representation is a Dyck
language, where the text corresponding to a node is delimited by beginning and ending parentheses each of which mentions the label. The text between these parentheses describes the children of the node.

XML is in essence a simplified form of SGML (Standard Generalized Markup Language) [44], a notation that has been in existence for a number of years in the specialized area of professional document preparation. The W3C (World Wide Web Consortium, www.w3.org) launched HTML, a specific SGML markup vocabulary, as the document standard that soon would carry most information on the Internet. Later, the W3C saw the potential of a data and content distribution format that would be more general, and it oversaw the pruning of SGML into XML. Most XML-related core standards are a result of work done under the auspices of the W3C, although we shall also discuss alternative technology that has emerged as a result of the perceived unnecessary complexity of essential XML specifications.

The labeling of data that is inherent in XML has led to the frequent characterization of XML data as “self-describing.” Usually, this is an exaggeration: the “description” is merely the labeling of the tree, something that by itself carries no meaning to either humans or machines. Nevertheless, the tagging of data is essential to a main application of XML, the representation of irregular data that do not easily conform to the tabular view of relational databases. The labeling makes irregularities amenable to automated processing. As an example, assume that information tagged book has a certain subtree structure, including descendants named author and title. Then such a subtree may be inserted flexibly in various structures that need to mention books, and applications can easily identify the subtree structure of a book. In general, there is, of course, a system to the irregularities. The set of possible structures is defined in notations called schema languages. These notations define the types of nodes and their relations in terms of their labels.

In general, labeled graphs for representing information are called semi-structured data. XML is superficially less general because it emphasizes trees. But just as the relational data model can represent arbitrary graphs by using attribute values as identifiers, XML allows arbitrary cross edges, something anticipated at the most fundamental level.

1.1.2 Overview of This Chapter

We conclude this introduction with some motivating examples of XML applications found in current and emerging business practice. Then, in Sect. 1.2 we provide a walk-through of the fundamentals of XML. We discuss the syntax and present a mathematical model that succinctly summarizes recent official W3C attempts to capture the essence of XML. Next, in Sect. 1.3, we discuss already deployed schema notations such as DTDs, RELAX NG, and XML Schema, and we hint at some interesting theoretical issues. In Sect. 1.4, we introduce the recent program notations that are of key practical significance: