the site, which in turn gives Amazon.com a strong corporate identity. This strong identity and the large number of committed visitors also makes Amazon.com an attractive platform on which third parties can build their identity through personalized advertisements. This has an impact on our view of the Web from a design perspective: we need to acknowledge multiple interests and the implications for gathering, storing and utilizing information about both intentional user actions and the possibly unintended traces left behind.

For more details on this study, see the paper "Sowing the Seeds of Self: A Socio-Pragmatic Penetration of the Web Artefact", which recently won the Best Paper Award at the 2007 International Conference on the Pragmatic Web (http://www.pragmaticweb.info). The paper is available from the authors on request and from the ACM digital library (http://www.acm.org/dl/).

Please contact:
Pär J. Ågerfalk
Lero – The Irish Software Engineering Research Centre, University of Limerick and Uppsala University
Tel: +353 61 213543
E-mail: par.agerfalk@lero.ie

Jonas Sjöström
Jönköping International Business School and Linköping University
Tel: +46 36 101784
E-mail: jonas.sjostrom@jibs.hj.se

Understanding the Hidden Web

by Pierre Senellart, Serge Abiteboul and Rémi Gilleron

A large part of the Web is hidden to present-day search engines, because it lies behind forms. Here we present current research (centred around the PhD thesis of the first author) on the fully automatic understanding and use of the services of the so-called hidden Web.

Access to Web information relies primarily on keyword-based search engines. These search engines deal with the 'surface Web', the set of Web pages directly accessible through hyperlinks, and mostly ignore the vast amount of highly structured information hidden behind forms that composes the 'hidden Web' (also known as the 'deep Web' or 'invisible Web'). This includes, for instance, Yellow Pages directories, research publication databases and weather information services. The purpose of the work presented here, a collaboration between researchers from INRIA project-teams Gemo and Mostrare and the University of Oxford (Georg Gottlob), is the automated exploitation of hidden-Web resources, and more precisely the discovery, analysis, understanding and querying of such resources.

An original aspect of this approach is the avoidance of any kind of human supervision, which makes the problem quite broad and difficult. To cope with this difficulty, we make the assumption that we are working with services relevant to a specific domain of interest (eg research publications) that is described by some domain knowledge base in a predefined format (an ontology). The approach is content-centric in the sense that the core of the system consists in a content warehouse of hidden-Web services, with independent modules enriching the knowledge of these services so they can be better exploited. For instance, one module may be responsible for discovering relevant new services (eg URLs of forms or Web services), another for analysing the structure of forms and so forth.

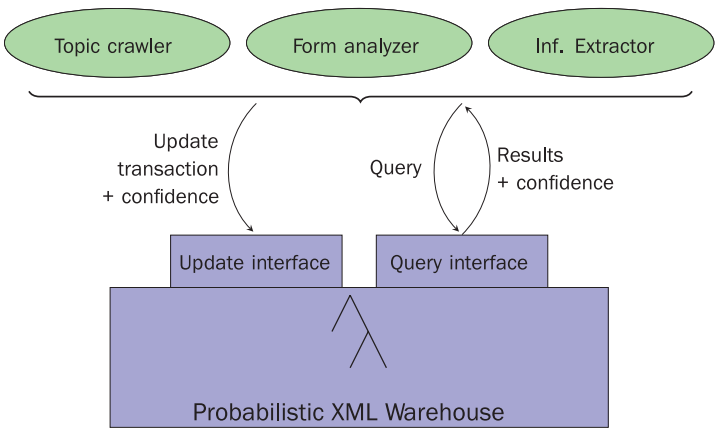

Figure 1: Probabilistic content warehouse, updated and queried by various modules that analyse the hidden Web.

Typically the data to be managed is rather irregular and often tree-structured, which suggests the use of a semi-structured data model. We use eXtended Markup Language (XML) since this is a standard for the Web. Furthermore, the different agents that cooperate to build the content warehouse are inherently imprecise since they typically involve either machine-learning techniques or heuristics, both of which are prone to imprecision. We have thus developed a probabilistic tree (or XML) model that is appropriate to this context. Conceptually, probabilistic trees are regular trees annotated with conjunctions of independent random variables (and their negation). They enjoy nice theoretical properties that allow queries and updates to be efficiently evaluated.

Consider now a service of the hidden Web, say an HTML form, that is relevant to the particular application domain. In order that it can be automatically reused, an understanding of various aspects of the service is necessary; that is, the structure of its input and its

output, and the semantics of the function that it supports. The system first tries to understand the structure of the form and relate its fields to concepts from the domain of interest. It then attempts to understand where and how the resulting records are represented in an HTML result page. For the former problem (the service input), we use a combination of heuristics (to associate domain concepts to form fields) and probing of the fields with domain instances (to confirm or invalidate these guesses). For the latter (the service output), we use a supervised machine-learning technique adapted to tree-like information (namely, conditional random fields for XML) to correct and generalize an automatic, imperfect and imprecise annotation using the domain knowledge. As a consequence of these two steps, the structure of the inputs and outputs of the form are understood (with some degree of imprecision of course), and thus a signature for the service is obtained. It is then easy to wrap the form as a standard Web service described in Web Service Definition Language (WSDL).

Finally, it is necessary to understand the semantic relations that exist between the inputs and outputs of a service. We have elaborated a theoretical framework for discovering relationships between two database instances over distinct and unknown schemata. The problem of understanding the relationship between two instances is formalized as that of obtaining a schema mapping (a set of sentences in some logical language) so that a minimum repair of this mapping provides a perfect description of the target instance given the source instance. We are currently working on the practical application of this theoretical framework to the discovery of such mappings between data found in different sources of the hidden Web.

At the end of the analysis phase we have obtained a number of Web services, the semantics for which is explained in terms of a global schema specific to the application. These services are described in a logical Datalog-like notation that takes into account the types of input and output, and the semantic relations between them. The services can now be seen as views over the domain data, and the system can use these services to answer user queries with almost standard database techniques.

Links:
http://pierre.senellart.com/phdthesis/
http://treecrf.gforge.inria.fr/
http://r2s2.futurs.inria.fr/

Please contact:
Pierre Senellart
INRIA Futurs, France
Tel: +33 1 74 85 42 25
E-mail: pierre@senellart.com

Static Analysis of XML Programs

by Pierre Genevès and Nabil Layaïda

Static analysers for programs that manipulate Extensible Markup Language (XML) data have been successfully designed and implemented based on a new tree logic by the WAM (Web, Adaptation and Multimedia) research team, a joint lab of INRIA and Laboratoire d'Informatique de Grenoble. This is capable of handling XML Path Language (XPath) and XML types such as Document Type Definitions (DTDs) and XML Schemas.

Since its introduction a decade ago, Extensible Markup Language (XML) has gained considerable interest from industry and now plays a central role in modern information system infrastructures. In particular, XML is the key technology for describing and exchanging a wide variety of data on the Web. The essence of XML consists in organising information in tree-tagged structures conforming to some constraints, which are expressed using standard type languages such as DTDs, XML schemas and Relax NG. XML processing can be seen as transforming these structures using tree-oriented query languages such as XPath expressions and XQuery within full-blown transformation languages such as XSLT.

With the ever-increasing information flow in the current Web infrastructure, XML programming is becoming a key factor in realizing the trend in Web services aimed at enhancing machine-to-machine communication. There still exist important obstacles along this path: performance and reliability. Programmers are given two options, Domain-Specific Languages such as XSLT, or general-purpose languages augmented with XML application programming interfaces such as the Document Object Model (DOM). Neither of these alternatives is a satisfactory answer to performance and reliability, nor is there even a trade-off between the two. As a consequence, new paradigms are being proposed and all have the aim of incorporating XML data as first-class constructs in programming languages. The hope is to build a new generation of tools that are capable of taking reliability and performance into account at compile time.

One of the biggest challenges in this line of research is to develop automated and tractable techniques for ensuring static-type safety and optimization of programs. To this end, there is a need to solve some basic reasoning tasks that involve very complex constructions such as XML types (regular tree types) and powerful navigational primitives (XPath queries). In particular, every future compiler of XML programs will have to routinely solve problems such as:

- XPath query emptiness in the presence of a schema: if one can decide at compile time that a query is not satisfiable then subsequent bound computations can be avoided
- query equivalence, which is important for query reformulation and optimization
- path type-checking, for ensuring at compile time that invalid documents can never arise as the output of XML processing code.

All of these problems are known to be computationally heavy (when decid-