# MOSTRARE

## Modeling Tree Structures, Machine Learning, and Information Extraction

**February 2003, revised September 2003**

http://www.grappa.univ-lille3.fr/mostrare/

Project proposal within the
INRIA Research Unit FUTURS by

R. Gilleron, Grappa group, ea 3588, Lille 3 University

**theme 3:** human-computer interaction, images processing, data management, knowledge systems

**theme 3A:** Databases, Knowledge Bases, Cognitive Systems

**Abstract**

*The Web of data with meaning in the sense that a computer program can learn enough about what the data means to process it.*

*Tim Berners-Lee, Definition of semantic Web from his book Weaving the Web, published 1999/2000*

During the last decade, the World Wide Web has evolved into the most important public data store on world. An important challenge for computer science today is to develop accurate information extraction and question answering mechanisms for the Web. Berners-Lee points out the difficulty of that task, and that it might even require more adequate formats of Web data representation. Information must be structured, and structure should reflect semantic information, so that machines can learn enough about what the data means.

The standard document formats of the Web today, `HTML` and `XML`, rely on tree structures that encompass textual information. In this project we want to incorporate novel approaches for modeling tree structure and emerging techniques for machine learning into adaptive information extraction systems for the Web. In the future, we might also have to account for semantic information.

# Research Team

The research team is a joint project team with the LIFL (CNRS and Lille 1 University) and the GRAPPA GROUP (Lille 3 University). The project will be located in Lille 3 University.

**Scientific Director**

| | | | |
|---|---|---|---|
| Rémi | GILLERON | full professor | GRAPPA EA 3588 |

**Members**

| | | | |
|---|---|---|---|
| Anne-Cécile | CARON | assistant professor | LIFL UMR CNRS |
| Aurélien | LEMAY | assistant professor | GRAPPA EA 3588 |
| Joachim | NIEHREN | research assistant[1] | MOSTRARE |
| Jean-Marc | TALBOT | assistant professor | LIFL UMR CNRS |
| Isabelle | TELLIER | assistant professor – délégation INRIA | GRAPPA EA 3588 |
| Sophie | TISON | full professor | LIFL UMR CNRS |
| Marc | TOMMASI | assistant professor – délégation CNRS | GRAPPA EA 3588 |

**Ph.D Students**

| | | | |
|---|---|---|---|
| Iovka | BONEVA | bourse MESR – monitrice – sept 02 | LIFL UMR CNRS |
| Denis | DEBARBIEUX | bourse MESR – moniteur – sept 02 | LIFL UMR CNRS |
| Julien | CARME | bourse MESR – moniteur – sept 02 | GRAPPA EA 3588 |
| Patrick | MARTY | bourse région NPdC & INRIA – sept 03 | GRAPPA EA 3588 |

[1] Joachim NIEHREN is research assistant in the Programming Systems Lab, Department of Computer Science of Saarbrücken. He works in the MOSTRARE project with the support of "région Nord-Pas de Calais" from march 03 to september 03, with the support of Lille 3 university from october 03 to december 03, and with the support of RU INRIA FUTURS from january 04 to september 04. He aims at obtaining a position in the MOSTRARE project. With such a position, he will become Permanent Director

# Contents

# 1 Introduction

During the last decade, the World Wide Web has evolved to the most important public data store on world. The recent data formats used on the web are heterogeneous and still evolving. There is pure text in natural language, `HTML` documents which add layout structure to pure text, semistructured `XML` data where information structure is available, and also all kind of binary data formats.

The web community is highly interested in adequate information representation so that information on the web can be accessed more easily. We therefore suppose that the amount of available `XML` documents will increase in the near future. A motivating power might be the semantic web initiative which heads for adding semantic information to semistructured data.

> *The eXtended Markup Language is accepted as THE emerging standard for data interchange on the web.* `http://www.semanticweb.org/knowmarkup.html`

Existing information extraction systems are by far too limited to account for accurate question answering for the web. This is pointed out by Berners-Lee's citation that we put in front of this proposal. Better algorithms are needed so that machines can learn enough about what the data means. The task will alter in the future through changes of information presentation, for instance, through web ontology languages[2] as proposed by the semantic web initiative.

A major challenge in that perspective is adaptive information extraction that can fully exploit the tree structure of web documents. Tree structure is present in the recent web formats, `HTML` and `XML`, in order to encompass textual information. In this project, we want to further integrate models of tree structures and emerging machine learning techniques into adaptive information extraction systems. Our approach should be sufficiently flexible so that we can integrate semantic information on the web once available.

**Information extraction.** Information extraction means to populate a database with values extracted from a collection of documents. Documents may have various formats: pure texts in natural language, web pages, `HTML`-documents, `XML` documents with or without data type descriptors (`DTD`). Collections of documents may be homogeneous or heterogeneous. The values to be extracted may be tuples or records (also called frames) of some given type.

An example document is given in Figure 1. This web page is built dynamically from a back-end database. This fact ensures regular document structure that will facilitate the task of information extraction. In this example, the task consists in filling the frame in Figure 2 with information about the persons mentioned on the page. Even though this example is quite simple, it already illustrates various requirements on information extraction systems:

- Identify all positions in the text where information to extract are located. Some locations are situated between structural tags (invisible in the figure), others may appear as chunks of longer texts.
- Handle heterogeneous data. Optionally, one might want to extract the information from parts of texts. In the example, the affiliation *UFR IDIST* is embedded in the sentence *maître de conférences en informatique, UFR IDIST, délégation INRIA à compter de octobre 2002, Lille 3.*
- Handle missing informations. For instance homepage and email addresses are missing for *Daniela Dudau* and there is no affiliation.
- Handle complex data structures. The web page of Figure 1 contains nested structures: the list of members has been firstly grouped by status (*Responsable*, *Membres*, *Membres associés*, *Doctorants*). For each status a nested list occurs.
- Combine information extraction tools. The web page of Figure 1 also contains a context that indicates that the page contains searched information (a title contains *Composition de l'équipe*).

Other requirements which are not illustrated by our example are: adaptiveness to format changes which are always to be expected on the web; diversity of data sources.

[2] `http://www.w3.org/2001/sw/WebOnt/`

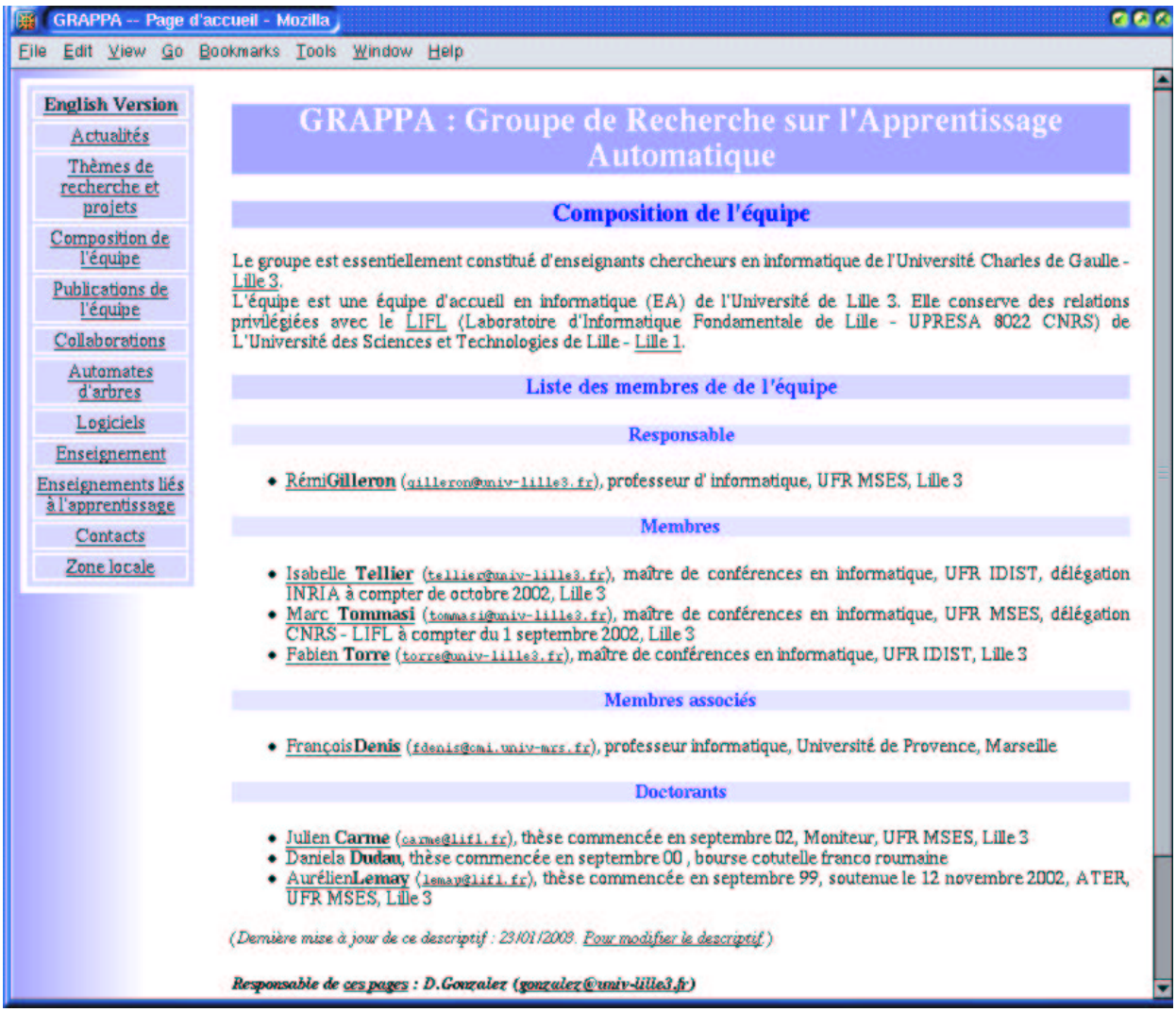


Figure 1: Example document generated by a database.

Programs that extract information are called wrappers. Writing wrappers by hand is inappropriate, extremely laborious and error prone. Therefore we consider the automatic design of information extraction systems from examples using machine learning techniques. Previous approaches are studied in the field of *wrapper induction*. Data formats on the web are diverse, but tree structured data will become prominent. Thus we are mainly concerned with wrappers using the underlying tree structure of semistructured data. Optionally, we will supply support for semantic information available on meta level or from web ontologies. The originality of our approach lies in the novel combination of machine learning techniques with modeling tree structures in adaptive extraction information systems.

**Goals Summarized.** The main objective of the present project is to develop adaptive information extraction systems for semistructured data that make use of the underlying tree structure. In the future, we might also have to incorporate semantic information.

The goals are the following:

**Modeling Tree Structures for Information Extraction:** the definition and investigation of models for tree structures and tree transformations relevant to the field of information extraction, including the design of efficient algorithms and software components.

Figure 2: A frame to be filled with information from Figure 1

```
Person data
   status:
   name:
   email:
   homepage:
   position:
   affiliation:
```

**Machine Learning for Information Extraction:** the development of machine learning algorithms that can induce models of tree structures and tree transformations. Their application in the field of information extraction, or more broadly to information retrieval and text classification. The combination of learning algorithms to define wrappers over diverse data sources and over heterogeneous data.

# 2 State of the Art

We first discuss the state of the art in information extraction, then present existing models for tree structures that are relevant in this context, and finally turn to machine learning approaches.

## 2.1 Information Extraction

Information extraction is a highly active research area. The recent interest is raised by the enormous development of the World Wide Web. There are millions of users who want to access and provide huge amounts of information. But existing information retrieval methods are often too limited to find available information. Professional Web services can handle only specialized cases and might be quite costly. We hope that information extraction techniques will help to solve some of the remaining problems.

### Task

Extracting information means to distinguish relevant pieces of information in a collection of documents. The relevant objects are usually tuples or frames, i.e., records that might be flat or nested. The type of these objects belongs to the input of the problem. The technical challenge is to discover the boundaries of meaningful segments in a Web document. Once extracted, information can be represented in formats that can be queried conveniently. This mostly means that extracted data will populate a traditional or semistructured database.

The task of information extraction scales with document formats. Obvious success criteria are precision, recall, and efficiency. Less obvious but perhaps more important is adaptiveness: it must be easy to adapt information extraction systems to changes of Web pages since these tend to happen frequently.

### Systems

The task of information extraction depends on the format of information presentation. Information in pure text can be located by using natural language technology. Tools of this kind are: RAPIER [29], SRV [90] and WHISK [161].

The `HTML` format is richer than pure text. It permits to structure information through layout information. Layout aware tools for `HTML` documents are W4F [151], Roadrunner [45], and XWRAP [117].

Semi-structuring documents in the `XML` format further improves the quality of presentation, since information structure can be represented independently of layout. Both formats, `HTML` and `XML`, use trees that are stored as strings to structure information presentation. More recent information extraction tools work on these trees obtained through parsing, for instance, Stalker [129], $WL^2$ [41], and Lixto [18]. The former two use machine learning

for wrapper generation. Lixto, in contrast, relies on user interaction for wrapper specification, and simplifies this task through a graphical user interface to a powerful tree logic (see below).

The XML format, however, solves only half of the presentation problem, as long as XML tags are difficult to interpret. Semantic information on XML tags can be represented in some web ontology languages. But there are only few semantic information on the Web so far. An information extraction tool that exploits web ontologies is described in [72].

### Applications

Information extraction is used in information mediator services, components of information retrieval tools, and classification systems.

*Information Mediator Systems* provides a query-only intermediate layer between the clients and a large number of heterogeneous informations sources, including databases and other sources on the Web. Given a query, information mediators determine which information sources to use, how to obtain the desired information, how and where to temporarily store and manipulate data ([97]). Examples of mediators are Ariadne [104], CQ [118], Internet Softbots [111] and Tsimmis [116].

*Information retrieval* implies many tasks from data storage, text processing, feedback handling, search strategies, etc., to natural language processing and content analysis [173]. Text processing includes classification, segmentation, routing and ranking which are essentially realized by learning algorithms using an appropriate document abstraction (document vectors, bag of words,...), distances and statistics ([99, 120]). Such learning algorithms proceed by a shallow analysis of textual data but are often limited because of the huge dimensions of the problem: a large number of words, long texts. Information extraction can participate in improving classification and segmentation systems while reducing these dimensions, providing better abstractions of textual data. For instance, extract relevant parts of documents (title, abstract, introduction, ...) or extract vector descriptions of (parts of) documents. Connections between information extraction and information retrieval for XML documents are discussed in [79].

### Wrappers and Queries

Programs that extract information from documents are often called wrappers. We distinguish wrappers for different documents formats. String wrappers apply to pure text and tree wrappers to different kinds of trees. Queries specify the extraction task of a tree wrapper. They define properties of nodes in trees that are subject to extraction.

*String Wrappers* apply to text including HTML or XML documents. String wrappers can be specified through regular expressions that delimiters relevant information in text. Let us consider the example in Figure 3. Information in this document is best represented by the following record:

$$\{\texttt{Congo} : 242, \texttt{Egypt} : 20, \texttt{Belize} : 501, \texttt{Spain} : 34\}$$

A simple wrapper to extract these information is specified by the following regular expression:

$$\texttt{<name>}\ *_1\ \texttt{</name> <nb>}\ *_2\ \texttt{</nb>}$$

This wrapper validates all attribute value pairs $(*_1, *_2)$ in matching substrings of the XML document. It belongs to the class of Kushmerick's left-right wrappers (LR-wrappers) [111] with the following set of left-right delimiters:

$$\{(\texttt{<name>}, \texttt{</name>}); (\texttt{<nb>}, \texttt{</nb>})\}$$

Kushmerick also considers slightly more expressive wrapper classes, which allow for disjunctions, sequences of delimiters, multiple value attributes, etc. But these wrappers are still fairly limited. Extracting nested structures by wrappers, for instance, turned out to be difficult.

*Wrapper induction.* Writing wrappers by hand is cumbersome and error prone. The same holds for special

```
<codes>
<info>Some Country Codes</info>
<collec>
  <code> <name>Congo</name> <nb>242</nb> </code>
  <code> <name>Egypt</name> <nb>20</nb>  </code>
  <code> <name>Belize</name> <nb>501</nb> </code>
  <code> <name>Spain</name> <nb>34</nb> </code>
</collec></codes>
```

Figure 3: An XML document

purpose programming support [93] beyond Perl or Java. The programming approach furthermore conflicts with the adaptedness requirement.

Adaptive information extraction systems can be built by machine learning techniques. An expert annotates relevant portions of text in a corpus of training documents. The learning system generalizes to produce a wrapper that extracts similar contents from other documents.

The problem was first addressed for natural free-text documents (see the Message Understanding Conference and the MUC challenge [126]). Following approaches have introduced techniques for string wrapper induction. It has been shown that string wrappers can be automatically learned for highly regular documents, such as Web pages generated by programs from back end databases [110, 97, 130, 39]. These techniques were successfully extended to natural language text domains [76, 161] to avoid the complexity of linguistically intensive approaches.

The tree structure underlying Web pages, however, is not exploited by string wrappers, so that these cannot extract nested information structures easily. Furthermore, Kushmerick [110] shows that the algorithmic complexity of the extraction process increases with the complexity of the learned string wrappers. Again, the tree structure of Web pages could be helpful.

According to the recent survey [110], challenges remain: find new methods for minimizing the amount of training data required to achieve a satisfactory level of generalization; find methods dealing with large numbered and varied information sources.

*Queries* are frequently used to select information in databases. They are often hidden behind graphical search templates, as known for instance from electronic phone books or bus servers.

Query languages for semistructured documents have been actively studied. They treat XML documents as rooted trees or graphs and can describe acceptable paths by navigating through the document. In [2], the authors consider semistructured data as an edge-labeled directed graph. A path expression is a regular expression used to represent a set of paths in the graph. It can be seen as a query for the set of nodes that can be reached over these paths. Languages such as Lorel [3] or UnQL [28] rely on such path expressions.

Standards of the W3C like XSLT, XQuery, are based on XPath, which specifies navigation paths in XML documents. Figure 3 presents an XML document – example from Kushmerick [111] – and Figure 4 its tree representation. The following three queries expressed in XPath apply to this XML document:

1. `/descendant::collec`
2. `/descendant::collec/child::code[position()=3]`
3. `/descendant::name[number(following-sibling::nb) = 501]`

The first query selects the node with label `collec`. The second one designates the third child with label `code` of the node with label `collec`. The third query returns the node with label `Belize`: descend to a node with label `name`, such that the following `nb` sibling contains the number `501`.

Queries can also be seen as logical formulas describing properties of nodes in a tree. Such logics (like monadic second order logic or Datalog) are logical counterparts of tree automata and will be discuss in Section 2.2.

XQuery [175] or XDuce [96] are typed functional query languages for XML. Their type systems are based on tree automata. XQuery relies on named types such as XML-Schema, whereas XDuce uses structural types and tree matching (see [159] for more details on these comparisons). CDuce [20], a ML-like language for semistructured data, extends XDuce with first-class and late-bound overloaded functions, and generalizes the boolean connectives

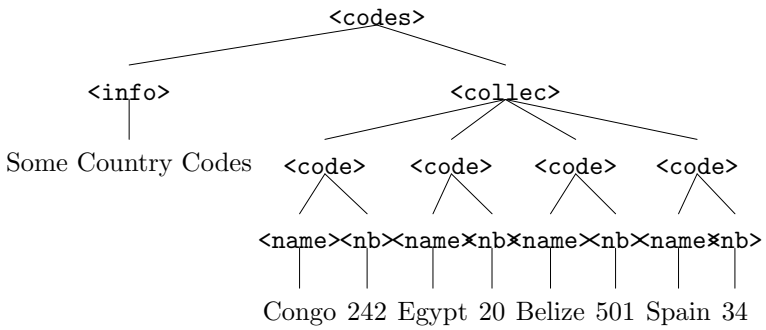


Figure 4: Tree representation

(explicit union, intersection, negation types).

*Query specification* is the approach behind the information extraction system Lixto [18]. Lixto users are supposed to define the distinguishing properties of informative nodes in HTML or XML trees, supported by a graphical interface which indicates potential properties. The system generates Datalog queries and applies them to the actual document. The user corrects those properties leading to unexpected results.
An important advantage of the Lixto approach is the enormous expressive power of Datalog which permits to specify queries precisely. But due to the absence of machine learning techniques, pure query specification systems seem less favorable with respect to adaptability.

## 2.2 Modeling Tree Structures

The evolution of XML into the major data presentation language has reawaken a strong interest in modeling tree structures. The main objectives are to query for nodes in trees (XPath), describe sets of trees and recognize membership (DTD), and to transform trees into others (XSLT). Modeling approaches rely on finite automata, monadic second order logics, modal logics, pattern languages, attribute grammars, or trees transducers.

### Tree Structures

Tree structures in the context of XML differ in several aspects from other tree structures used in other applications in computer science. The existing language models have to be adapted to these changes.

*Ranked and Unranked Trees.* Trees in rewriting, theorem proving, or type theory are usually ground terms built over a ranked signature of function symbols, each of which has a fixed arity. In XML however, trees are built over an unranked signature of symbols. Node labels do not determine the number of node children. Indeed, the number of node children is not bounded by the signature. As an example, consider the list encoded in the XML tree depicted in Figure 4. Node <collec> has an unbounded number of successors depending on the number of items in the list.
Of course, unranked trees can be encoded into ranked trees. An unranked tree $f(x_1, ..., x_n)$, for instance, can be mimicked by the binary tree $f(x_1, f(x_2, \ldots, f(x_{n-1}, x_n)\ldots))$. Alternatively, one can use another representation by a binary tree where each node has one edge to the first child and one to his next younger sibling (see for instance [133]).
Encodings have many consequences. They don't preserve all tree structure. Paths are usually altered, i.e., the relative addresses of nodes starting from the root. Paths are used in many tree models, for filtering, querying, or tree transformations; such models have to be carefully adapted. Encodings may add additional nodes, which have no meaning with respect to the original document. They somehow have to keep track of two different levels of recursion, the one in the width and the other in the depth. They have some impact on the representation size and also on the complexity of recognition or querying.

*Ordered and Unordered Trees.* Children ordering is not always mandatory or relevant to queries. Suppose for instance, one wants to check whether the collection stored in the XML document of Fig. 4 is the same than

in some other document. It might then be valuable to ignore children ordering, i.e., to treat these collections as unordered trees.
Moreover, XML-Schema supports unordered trees. One can specify that a document contains elements $e_1, \ldots, e_n$ regardless of their order. Furthermore, attributes of XML elements are unordered. Thus, mixing ordered and unordered nodes in a tree is essential in XML framework.
A traditional approach to model unordered trees is to use associative and commutative function symbols. That is to say that trees belong to a non free algebra governed by axioms of the form $f(x_1, x_2) = f(x_2, x_1)$ and $f(f(x_1, x_2), x_3) = f(x_1, f(x_2, x_3))$.

*Finite or Infinite Alphabets.* The label sets for XML trees are not necessarily finite. This problem can already be introduced by attributes of XML. For instance in the tag <input type="text" value="Bonjour">, value of the attribute type is "text" but could take about ten other values. The value of attribute value is "Bonjour" but could be any other element of the infinite set of strings.

**Tree Automata**

The use of tree automata for structured documents was already advocated by Kilho Shin in 1992. Many researchers followed this approach in the context of XML. Here, one of the main needs is to account for unranked trees.

*Unranked Trees.* Automata for unranked trees including the concept of unranked trees were introduced by Thatcher in 1967 [169, 168] and reconsidered by Takahashi in 1975 [162]. Other researchers have furthered this theory, like Barrero [16, 125].
In the XML context, automata for unranked trees were initiated by Murata, Brüggemann-Klein and Wood [27]. They coined the term *hedge automata* for Thatcher's notion of tree automata on unranked trees and introduced more general two-way hedge automata. Closure and decidability properties of regular tree languages have been extended from the ranked case to the unranked case [133].
Query automata [136] are two-way hedge automata that can answer queries about nodes in a tree. The idea is to select appropriate nodes during a run moving up and down through a tree. In the ranked case, query automata compute exactly MSO definable unary patterns —*i.e.* queries with a single free variable which are definable in monadic second-order logic over trees. In the unranked case, unary pattern queries can be computed by a little more sophisticated device, strong query automata.
Neumann and Seidl [132, 131] proposed *pushdown forest automata* which operate on unranked trees or more general hedges, i.e., sequences of unranked trees. The term *forest* refers to hedges rather then to sets of trees. Pushdown forest automata traverse hedges depth-first and left-first. They can always refer back to the previous state seen during the traversal. This can be implemented by looking up the pushdown of the evaluator of a forest automaton. Pushdown forest automata were applied to querying XML documents, possibly on the fly during parsing. The goal is to minimize the number of necessary traversals through the tree (to either two or one).
Murata, Lee, and Mani in [127] have defined different subclasses of regular tree grammars in connection with schema language proposals, studying the corresponding algorithms and their complexity for document validation and type assignment. Segoufin and Vianu use two way automata for validation [157].
As pointed out by F. Neven, XSLT is essentially a tree-walking tree-transducer with registers and look-ahead. Caterpillar expressions [26] were a first instance of tree-walking in the context of unranked trees. It's still an open question whether tree-walking cannot define all the regular languages (conjectured by Engelfriet and Hoogeboom). Several extensions of tree walking automata have been defined and their connections with logic have been studied. For instance, F. Neven has equipped them with a finite number of registers to deal with data values.

*Ordered and Unordered Trees.* Extensions of tree automata that recognize languages of terms modulo such equational theories were studied by e.g. [174, 92, 145]). This idea permits to mix associative-commutative symbols and free symbols. Equational tree automata, however, are less suitable for associative symbols; not surprisingly A-regular tree languages inherit the negative results of context-free languages [146].
Lugiez and Dal Zilio proposed *sheaves Automata* [178] quite recently which are closely related to the automata of Seidl, Schwentick, and Muscholl [158]. Both approaches add arithmetic constraints to recognize multisets of trees. They allow both associative and associative-commutative symbols but their restriction to the associative-

commutative case is close to multitree automata defined in [140].

*Infinite alphabets.* To account for infinite set of attribute values, one can associate with each node a value which can be chosen from an infinite set: this leads to the notion of attributed tree defined *e.g.* in [135]. An other approach consists in dealing with trees over infinite alphabets, a label expressing both the value of the attribute and `HTML` tags. This approach has been initiated byN even, Schwentick and Vianu [137]. They define conservative extensions of classical automata and logics (MSO). Their results hold only for words but should be easily extended to the tree case. Another point of view is to encode the attribute assignment by a list of characters as subtree [91].

**Tree Logics**

Queries about nodes of a tree can be expressed by logical formulas with node variables and tree models, as widely accepted in the `XML` community [134, 135, 91, 19].

*Monadic Second Order Logic (MSO).* The MSO over unranked trees was used by Gottlob and Koch to study the expressiveness of query languages such as `XPath` [91]. In the terms of MSO, querying is often called model checking. Monadic Datalog over unranked trees is a logic programming sublanguage of MSO with the same querying power. Monadic Datalog is the query language behind the Lixto system.
MSO over ranked trees was introduce in the seminal work of Thatcher and Wright in 1968 [170]. MSO sheds a logical perspective on tree automata. Both formalisms can describe the same tree languages. Query tree automata have the same querying power as MSO [135].
MSO talks about the ancestor relation between the nodes of a tree. This relation is also called dominance relation in the context of dominance constraints that are prominent in computational linguistics [150, 81, 128, 15]. Efficient algorithms for solving dominance constraints could be found only recently [7].

*Modal Tree Logics* for describing trees are traditionally used in *feature logics*. Feature logics was invented by Kaplan [101] and Kay [102] and investigated by Ait-Kaci [6] and Smolka [160]. Its formulas describe record like structures such as frames, feature graphs, or feature trees. Feature trees are trees with labeled edges and nodes that are unbounded, unranked, and unordered. Feature trees were invented to represent semi-structured information. Features are often seen as modal operators relating nodes to their successor.
Michael Benedikt and Leonid Libkin [19] investigated feature logics from the `XML` perspective; they considered unranked trees such as $a(b, a, b, a)$ as feature trees $a(1 : b, 2 : a, 3 : b, 4 : a)$ and investigated ordering constraints over feature trees [122] in query languages.
*The ambient calculus* [32] is another formalism useful for tree description that relying on modal logics. It was introduced by Cardelli and Gordon in order to extend the $\pi$-calculus [119] and aims to model mobility for both hardware and software system. The ambient calculus adds explicit locations (*ambients*) to the $\pi$-calculus. Nested ambients represent a tree of concurrent processes; mobility is expressed by transforming this tree.
Recently, Cardelli and Ghelli [31, 30] proposed to use (shapes of) ambient processes as descriptions for sets of semistructured documents and the spatial part of the ambient logic as a query language, TQL. Ambient trees are pretty close to multiset feature trees considered in [139].

## 2.3 Machine Learning

Our project is concerned with information extraction and the main application domain is mining of the World Wide Web for information mediator services. Thus, we study learning from multiple databases containing texts, documents and semi-structured data and are faced with the current challenges in machine learning [121, 62]: scaling up algorithms to large databases, designing algorithms from non-attribute-vector data (sequences, texts and documents, images, ...), designing algorithms from heterogeneous data, learning from an open-ended stream of data, interfacing algorithms with the human user.

Many of the approaches to wrapper induction discussed in Section 2.1 can be categorized as finite state approaches, in that the learned procedures are formally equivalent to – possibly stochastic – grammars, automata and transducers. It should be noticed that most systems learn simple wrappers that define specific token sequences preceding and following the target field. Only very recent works consider the tree structure of the input documents.

When the tree structure is considered, algorithms are straightforward extension of the string case to the tree case.

We first discuss the state of the art in grammatical inference, then present known results for inference of tree structures, and finally we discuss recent advances in machine learning which should be helpful for our project.

**Grammatical Inference - Exact Learning** The process of learning finite state machines from data is referred to as grammatical inference. The first theoretical foundations were given by Gold [87] and first applications were designed in the field of pattern recognition. Grammatical inference is now a specific research domain with new application domains such as computational linguistics, natural language processing, speech recognition and bio-informatics among others. Grammatical inference refers to the following situation: a class of possible languages is specified, the learner can access information about an unknown target language, the learner must output an hypothesis language in the class. Many definitions of learnability are possible depending on the information presentation, and on the criterion of success. Among all definitions, we only consider learning models which are relevant for the application domain of information extraction, and are mainly interested in the design of efficient algorithms relative to the task.

The first learning model is usually called *learning from text* or *identification in the limit from positive data*. It was first defined by Gold in his seminal paper [87]. A class of languages is chosen together with a representation schema – for instance the class of regular languages and deterministic finite state automata. The information presentation is a text which generates the strings of the target language in any order such that every string occurs at least once. At each time the learner receives a string and is to make a guess about the target language. This process continues forever. The class of languages is considered learnable if there is an algorithm that the learner can use to make his guesses, the algorithm having the following property: given any language of the class, there is some finite time after which the guesses will all be the same and they will be correct. The learner is said to be efficient if the time needed to compute the $n$th guess is polynomial in the overall length of the $n$ strings seen up to this point. This learning model is very simple and natural, but has the drawback that not all regular languages are identifiable [11]. A learnable class is the class of $k$-reversible languages [12].

A second learning model called *identification from given data* was defined by M. E. Gold in [88]. A sample $S$ of labeled strings containing positive examples (strings in the target concept) and negative examples (strings in the complement of the target concept) is presented to the learner. The consistent learner must return in polynomial time a guess compatible with the examples. A class of languages is learnable if, for each language in the class there exists a characteristic sample with which the consistent learner returns a correct guess, and this must be monotone in the sense that if correctly labeled examples are added to the characteristic set, then the consistent learner infers the correct guess. Deterministic finite automata are learnable from given data [88, 147], but finite automata are not [50].

Another model was defined in the *active learning* framework where the learner may ask questions to a teacher. The presentation is no more arbitrary but can be controlled by asking queries to an oracle. The two most important sorts of queries are the membership queries and equivalence queries. In a membership query, a string is proposed to the oracle which returns its correct label. An equivalence query consists in proposing a guess to the oracle which either accepts it as a correct guess for the target language, or returns a counter-example, that is, a string from the symmetric difference of the guess and the target language. The class of languages is *exactly learnable from equivalence and membership queries* if there is an algorithm having the following property: given any language of the class, there is some finite time after which the algorithm halts with a correct guess, and at any stage the running time of the algorithm is polynomial in the size of the target language and the size of the longest counter-example seen up to this point. Deterministic finite automata are exactly learnable from equivalence and membership queries [13].

**Grammatical Inference - Statistical Learning** Another major research topic in grammatical inference – and in statistical learning – is stochastic modeling and training of stochastic grammars and stochastic automata. Stochastic modeling has become increasingly important for applications such as speech recognition, natural language processing and biological sequences analysis. Stochastic modeling is also used for adaptive information systems [77]. A stochastic automaton is obtained by specifying a probability for each transition of an automaton. A stochastic automaton assigns a probability to each string which it recognizes and hence defines a probability distribution over the set of strings. Learning stochastic automata is related to the difficult problem of learning probability distributions. Stochastic automata are also known as *Hidden Markov Models* (HMMs). The problem

of learning stochastic automata from examples has two aspects: determine the discrete structure or topology of the automaton and estimate probability parameters of the stochastic automaton. When the structure is fixed – chosen by the learner with some extra-knowledge – the Baum-Welch algorithm [17] and variants apply and are widely used. Inferring the model topology remains a difficult problem. Another widely used model is stochastic context-free grammar; the associated learning paradigm is studied [113].

**Inference of tree structures** In the present project we make the assumption that the tree structure of documents should be helpful for the design of adaptive information extraction systems. This assumption is supported by theoretical results in the field of grammatical inference for which tree structures have been shown to be helpful. Let us consider the problem of learning context-free languages from text (from positive-only examples). According to a theoretical result by Gold, the class of context-free languages cannot be learned from positive examples. This fact seems to show that learning from positive examples is too weak to find practical and interesting applications. However, learning from positive examples is very useful and important for practical use in text mining because labeled examples (positive and negative) are costly (a human expert must annotate the documents) or even unavailable, while positive examples are freely available.

Sakakibara [152] considered the problem of learning context-free grammars from their structural descriptions. A structural description is a derivation tree whose internal nodes have no labels. It is shown that reversible context-free grammars are efficiently learnable from positive samples of their structural descriptions. The algorithm is based on an extension of the algorithm for 0-reversible – string – languages [12] to the case of 0-reversible tree languages. Also, categorial grammars are learnable if the learner is given structures of derivation trees in the target grammar [100]. These two theoretical results about context-free grammars and categorial grammars show that additional structural information in the form of tree structures help learning.

Grammatical inference mostly focused on learning string languages, but recent works are concerned with learning tree languages. Some theoretical research papers extend results from the string case to the tree case. For instance, following Sakakibara's algorithm for 0-reversible tree languages, Fernau [75] studies learning tree languages from text where a text generates the trees of the target tree language in any order such that every tree occurs at least once. Also, results on pattern languages [10, 149] were extended to the tree case [89]. Other learnability results were obtained in the exact learning model with queries for tree patterns in the case of unranked trees [8, 14]. These results are motivated by information extraction methods for `XML` data, but the results are only theoretical for at least two reasons: the learning model is not suited for information extraction and tree pattern languages are too restricted.

Another interesting line of research is the inference of `DTD`s or `XML` schema from `XML` documents. Despite of their importance, however, `DTD`s are not mandatory and it is frequently possible that only specific portions of `XML` repository have associated `DTD`s. Thus, it is important to devise algorithms that can infer a `DTD` for a collection of `XML` documents. Some naive tools are available: see for instance the IBM Alphaworks DDbE product. Also a system XTRACT was defined in a data mining project at Bell Laboratories, but to our knowledge the tool is not available [82]. New `XML` schema languages have been proposed, and inference of `XML` schema has also been studied [40].

To end up this paragraph, we mention a work by Kosala et al on information extraction using tree automata induction [109] which is directly related to the objective of the present project. The experimental results show that the approach compares favorably with previous approaches. In this work, trees are finite ranked ordered trees which, however, are not fully adequate to model semi-structured documents (see Section 2.2 below). A very recent work by the same authors [108] considers unranked ordered trees. Inference of tree languages is an emerging field. To our knowledge, many algorithms are extensions to the tree case of previously known grammatical inference algorithms and mostly finite ordered ranked trees are considered.

**Current challenges for machine learning and information extraction** Let us come back to adaptive information extraction systems. A key bottleneck with these systems is obtaining the labeled training data. The use of machine learning is motivated by the fact that the cost of labeling documents is usually considerably less than the cost of writing extraction rules (wrappers) by hand. Nevertheless, labeling documents is generally tedious, and can be too expensive. Thus, techniques aimed at minimizing the amount of training data required for generalization are important. We next describe three recent research topics whose aim is to reduce the number of training data. In each of the three frameworks, it is supposed a – small – pool of labeled data together with a – large – set of unlabeled data.

Research on *semi-supervised learning* is concerned with the design of learning algorithms from both labeled and unlabeled data. Among different approaches, several supervised learning algorithms with the help of unlabeled data have been defined for text learning tasks. Such approaches include using *expectation maximization* to estimate maximal a posteriori parameters [143], using transductive inference for support vector machines [98], using the unlabeled data to define a metric or a kernel function [95], using a partition of the set of features into two disjoint sets of features [21, 144].

The second one is *co-training* which was first defined in [21]. In the co-training setting, two views over examples are assumed and co-training algorithms explicitly use the two views. The idea is to incrementally build classifiers over each of the two views: each classifier is initialized using just the small input labeled data set; then, at every step of co-training, each of the two classifiers chooses unlabeled examples to add to the set of labeled examples; then, each classifier rebuilds from the augmented labeled data set, and the process repeats. A co-training method has been used previously to train classifiers in applications like text classification [21], word-sense disambiguation [176] and named-entity classification [42].

The third one is *active learning*. The basic idea of active learning is to start with a small set of labeled data, run the learning algorithm, and then use the learned procedure to predict which of the instances in the set of unlabeled data is more informative in the sense of helping the learning algorithm to generalize most with one additional training document, and the process repeats. It is said to be active because the learner asks the human expert for the label of an unlabeled example, the question corresponds to a membership query. Active learning is an important topic for all machine learning paradigms, see for instance the ECML-PKDD 2001 Workshop on Active Learning, Database Sampling, Experimental Design: Views on Instance Selection.

Last but not least, a central challenge to information extraction is the ability to scale with the number and variety of information sources. Again, recent machine learning developments deal with combination of learning algorithms and learned procedures. The co-training setting deals with the combination of two learning algorithms over two views over examples. Co-training algorithms can be viewed as boosting algorithms. *Boosting* [78, 156] is a procedure for improving performance of a "weak" machine learning algorithm by repeatedly applying it to the training set, at each step modifying training example weights to emphasize examples on which the weak learner has done poorly in previous steps. The ability of boosting to improve upon the performance of the underlying weak learner has been verified in a wide range of of empirical studies in recent years (see the overview [154]). The learned classifier is a weighted majority vote of many based classifiers. Thus, boosting can be viewed as a particular case of *ensemble methods* for combining machine learning algorithms. All these methods should be investigated for combining adaptive information systems over different information sources or with different views over information sources.

# 3 Previous Own Research

## 3.1 Modeling Tree Structures

The development of the state of the art in modeling tree languages with automata, logics, or constraints was clearly promoted through the contributions of members of this project over the last years.

#### Tree Automata and their applications

In collaboration with H. Comon, F. Jacquemart and Denis Lugiez, we have written a book on tree automata [43]. This book is available on the web and its content evolves with the suggestions of our readers.

We have enriched tree automata with different constraints, automata with equality or disequality constraints on direct subterms [25, 24], or encompassment automata which test subterms at different levels [35, 46]. We generalized the notion of encompassment automata while keeping good closure and decidability properties [34], and we used them to prove decidability of encompassment theory.

We have investigated the use of tree automata as decision algorithms, in particular in rewriting theory and for solving set constraints. In this framework, we have first defined ground tree transducers which are very simple transducers which have good decidability properties and enable to simulate ground rewriting. More recently, we have defined generalized classes of tree automata which recognize generalized sets of trees, *i.e.* the language recognized is a set of tree languages [86, 83, 84, 85, 171].

For some classes of generalized set constraints, *e.g.* generalized definite and co-definite constraints which are interesting for program analysis, we have designed by tree-automata techniques new algorithms for computing the least (resp. the greatest) solution [163, 60, 61].

In [9, 47], we use word automata techniques to solve the boundedness property for regular path queries, in the context of semistructured data modeled as rooted directed edge-labeled graphs [4]. Now, we study how to extract a schema and in particular a set of inclusion constraints and inverse constraints from a semistructured data, or a set of semistructured data [1].

**Logics and Constraints**

*Dominance constraints.* We have investigated complexity and algorithms for dominance constraints or logics, talk about the ancestor relation in ground terms. We have also applied dominance constraints for modeling scope underspecification in natural language [70, 73, 105, 71].
We showed in 1998 that the satisfiability problem of dominance constraints is NP-complete [107]. In 2000, we could nevertheless present a constraint solver for dominance constraints based on constraint programming with finite sets [64], which is sufficiently efficient for most applications in semantic underspecification.
In 2001, we could distinguish the sublanguage of *normal dominance constraints* [7], which suffices for most applications. We have presented the first polynomial time algorithm that tests the satisfiability of normal dominance constraints. It is a graph algorithm that we have implemented in the LEDA library for C++.

*Monadic Second-Order Logic.* We have shown dominance logic equivalent to the weak second order monadic logic over ground terms, or the binary tree [142], and thus clarified their relationship to tree automata. We have also related dominance constraints to context unification [138] and hole semantics [106].

*Equality constraints.* We have invented parallelism constraints [74] which extend on dominance constraints, but can express equality of tree segments in addition. Parallelism constraints are equal in expressive power to context unification [141].
Parallelism constraints are part of the constraint language for lambda structures [70] that we developed as uniform foundation for semantic underspecification. Parallelism constraints have two main applications in CLLS: modeling ellipsis in natural language, and underspecified beta reduction [22, 23].

*Feature constraints* describe attributed trees that are also called records, or frames, or feature trees. Already in [139] we proposed a notion of tree automata for record structures called feature trees. Later on, we proposed ordering constraints over feature trees [123] and investigated their expressiveness and complexity in depth [124, 122]. A closely related language was proposed by Benedict and Libkin for `XML` querying [19].

*Ambient logic.* We investigate decidability and complexity issues about verification in terms of model-checking for fragments of ambient calculus against the ambient logic [38, 37, 36] and the satisfiability problem of the ambient logic for which very few and restricted works have been carried [37].

## 3.2 Machine Learning

The research field investigated by the *Grappa team* is Machine Learning. This theme is developed according to two directions of research : learning from heterogenous data and applications to data mining; grammatical inference and natural language learning.

**Learning from heterogenous data** Recently there has been significant interest in learning algorithms that combine information from labeled and unlabeled data. This research area is motivated by the fact that it is tedious and expensive to hand-label large amount of training data, specially for text learning tasks. For many text learning tasks, such as document retrieval and classification, one goal is the efficient classification and retrieval of interests of some user. Positive information are readily available and unlabeled data can easily be collected. Let us describe in more details two examples of such learning tasks. A first example is learning to classify web pages as “interesting” for a specific user. Documents pointed by his bookmarks define a set of positive examples because they correspond to interesting web pages for him and negative examples are not available at all. Nonetheless,

unlabeled examples are easily available on the World Wide Web. A second example is retrieval of bibliographic references. Positive examples are the bibliographic references which are in the target user's bibliographic database. Unlabeled examples are easily available in the bibliographic databases accessible via the World Wide Web. In both cases, learning algorithms from positive and unlabeled data apply. We have defined a learning model from positive and unlabeled examples and we have designed a decision tree learning algorithm from positive and unlabeled examples in [48], we have obtained theoretical results in [115], a synthesis should appear in [53]. We have also designed a text learning algorithm [54]. Also we have designed a co-training algorithm for texts from positive and unlabeled examples [52].

We have also considered heterogenous data in the classical sense. We have designed classification algorithms from tabular data together with text data. Our algorithm ADTBoost [49] can handle tests on (continuous or discrete) tabular data as well as tests on text data. This is particularly valuable in medicine where descriptions of patients combine diagnostic analysis, comments, dosages, measures, and so on. For instance, a rule can be built on both temperature and diagnostic, *if temperature > 37.5 and diagnostic contains "Cardiovascular" then* .... This kind of combination in a unique rule is not considered by others algorithms like Boostexter [155] which implements AdaBoost.MH [156]. Moreover, our algorithm is adapted to the multi-label case in which examples can belong to more than one class.

**Language Learning** Most algorithms in grammatical inference are based on representation schemes which have some kind of determinism. This limits their application field either on regular languages or on limited subclasses of more expressive language classes. Moreover it is known that the size of a deterministic automaton can be exponential in the size of an equivalent non-deterministic automaton. Our main idea is to define non-deterministic representation schemes that allow learnability. The class of Residual Finite State Automata (RFSA) [57, 58] is a class of non-deterministic finite automata such that any regular language can be represented by a RFSA. This class has some interesting properties both from a language theory and a grammatical inference point of view. For example, every regular language admits a unique minimal canonical RFSA which can be far smaller than the minimal deterministic automaton. This representation also allows to define new learning algorithms for regular languages in different learning models [55, 56, 59]. The next step is to find classes of representation allowing learnability for classes of languages more expressive than regular languages, following ideas used to define RFSA. Works are already in progress for stochastic automata [51], transducers and tree automata [33].

We are also involved in learning formal grammars used in natural language processing. Natural language learning is a universal human capacity whose mechanism remains mainly mysterious. The computational modeling of this process is a challenge recently addressed. In this vast domain, we restrict ourselves to the acquisition of syntactic rules, *i.e.* natural grammatical inference. The domain imposes specific constraints on candidate learning algorithms among which are:

- admit only positive examples;
- learn a non trivial class of grammars including ones generating at least context-free languages;

Our original contribution to this domain is to take into account lexical semantic information, supposed to be first acquired, to help the syntax learning process. This strategy takes advantage of the Principle of Compositionality to impose constraints on admissible tree structures underlying the input data. The idea is that only the syntactic structures that are compatible with a semantic interpretation are to be considered as possible analysis trees. This is a way to restrict the combinatorial explosion between a linear sentence and its possible syntactic structures. Our approach includes two learning algorithms, theoretical learnability results and implementations [165, 164, 166, 167, 67, 68, 69, 65, 66].

## 3.3 Software

The members of the Mostrare team are experienced in developing middle and large scale software systems.

**Tree Automata.** A toolbox that implements main algorithms on tree automata was developped[3]. We are currently designing a toolbox for unranked tree automata [4].

[3] `http://www.lifl.fr/~debarbie/TATA/`

[4] `http://www.grappa.univ-lille3.fr/twiki/bin/view/Public/Software`

**Mozart/Oz.** The Mozart Programming System [5] is based on the Oz language, which features concurrent, constraint, and distributed programming. J. Niehren has participated in the design of Oz since 1991, tested all versions in early stages over the years, and developed many libraries for the system, and large scale applications.

The Mozart Programming System offers an advanced development platform for intelligent, distributed applications. The system is the result of a decade of research in programming language design and implementation, constraint-based inference, distributed computing, and human-computer interfaces. As a result, Mozart is unequaled in expressive power and functionality. Mozart has an interactive incremental development environment and a production-quality implementation for Unix and Windows platforms. Mozart is the fruit of an ongoing research collaboration by the Mozart Consortium in Saarbrücken, Louvain, and Stockholm.

**Constraints.** J. Niehren is the main implementor and coordinator of the Chorus demonstration system, which illustrates semantical processing of natural language with constraints. The system development started in 1997 and was implemented by several researchers, Phd students, and master students.

The demonstrator is developed with Mozart/Oz. It contains a unification based parser for English written in the HPSG grammar formalism, a syntax semantics interface, and a collection of constraint solver for different sublanguages of the constraint language for lambda structures (CLLS). There is a solver for dominance constraints based on finite set constraint programming, two solver for normal dominance constraints relying on efficient graph algorithms, and a semi-decision procedure for the whole of CLLS.

**Speech control for the elevator.** J. Niehren headed the software development for a speech controlled elevator. The system is up and running in the elevator the building for computational linguistics in Saarbrücken. If one asks the elevator to go the office of professor Pinkal, it will answer that this office is in the ground floor, and go there. The system was presented to the public and discussed by several articles in local newspapers. The development took around 2 years (2000-2002) and was powered by several master students.

The system can be used via microphone and loudspeaker in the lift cabin that are connected by analogue tone techniques to the sound cards of a computer standing in the ground floor. Interfaces to the lift control and speech recognizer were developed. The core component however consists of a dialog system that we designed and implemented with finite state technology.

**Machine Learning algorithms.** Along our research and for some projects, we have designed different software systems: **ADTBoost**[6] is a multilabel supervised classification algorithm from structured data and texts, it was designed and used in the project DATADIAB ; **C4.5POS**[7] is a decision tree learning algorithm from positive and unlabeled examples ; **NBPOS**[8] is a naive Bayes learning algorithm NBPOS from positive and unlabeled documents ; a prototype for learning categorial grammars has been designed for testing hypotheses concerning the learning of the syntax helped by semantic types ; stochastic algorithms for classification have been designed in the firm Pertinence Data Intelligence along original ideas of our research team (Fabien Torre is scientific advisor).

# 4 Scientific Goals and Software Development

The main goal of the project is to design adaptive information extraction systems that fully exploit the tree structures of `XML` or `HTML` documents. In our approach we want to combine novel models of tree structures and machine learning techniques. We will define induction algorithms learning wrapper or queries, and combine tree wrappers with string wrappers. We will develop experimental software systems for information extraction.

## 4.1 Modeling Tree Languages

Appropriate models of tree languages should satisfy a number of properties imposed by adaptive information extraction. Possible trade-offs between expressiveness, learnability, and efficiency are to be understood.

[5] `http://www.mozart-oz.org`
[6] available at `www.grappa.univ-lille3.fr/` and referenced on the site Boosting.org `http://www.boosting.org/`
[7] available at `www.grappa.univ-lille3.fr/`

**Requirements.** We need to model languages of unranked trees with infinite alphabets and possibly unordered edges. We will have to restrict ourselves to learnable subclasses of regular languages. Machine learning requires a number of operations on language models. Tree wrapper or query induction on basis of grammatical inference will frequently test for *membership* in tree languages given a model. It computes a fixed-point; whether the fixed-point is reached amounts to testing for *equivalence* of tree languages given their models. Learning algorithms generalize examples into models. *Inclusion* testing can be used to avoid over-generalization especially when learning from positive examples only.
The size of models is important with respect to efficiency, thus, model *minimization* becomes an issue. Minimization for automata, for instance, yields small canonical language representations with reduced costs of inclusion testing. *Incrementality* is an issue in interactive learning processes, such as active learning. Here it is important to update language models incrementally.
Language models should be as *readable* as possible so that they can be presented to users. This is relevant to interactive wrapper specification, but also during development cycles with automatic wrapper generation tools. Language should be *robust against noise*, i.e., errors in training data. In the case of tree wrapper induction, training data consists of trees with segments of interest annotated by human experts, which tend to make errors.

**Tree automata.** Tree automata provide a powerful framework for defining classes of tree wrappers, that yield a good balance between learnability, expressiveness and efficiency.
Existing notions of tree automata for unranked trees were discussed in Section 2.2. The most popular are hedge automata and pushdown forest automata. These approaches mostly adopt operational or syntactical perspectives, i.e., how to traverse trees with finite memory and how to define automata by using regular expressions.
Such approaches neglect algebraic properties of unranked trees, i.e., how they are constructed. The importance of the algebraic view in automata theory was already argued in Thatcher's seminal works on traditional tree automata [170] and in Courcelle's generalisations for other data structures [44]. Therefore, we want to develop an algebraic automata notion for unranked trees. Such tree automata should satisfy all closure and algorithmic properties required above, simply because of their algebraic nature. They should permit to define the same classes of tree languages or queries as previous automata notions for unranked trees. Furthermore, grammatical inference relies on algebraic properties; thus, an algebraic notion of tree automata should support the development of such learning algorithms.
We want to develop classes of queries starting from deterministic top-down tree automata. These automata enjoy incremental membership testing, since the paths of the input tree can be checked independently. This test requires linear space in the depth of the tree and not in its size. Unfortunately, deterministic top-down tree automata are not expressive enough for some useful applications. But we hope that we will find suitable extensions that can be learned by algorithms with low complexity.

**Tree patterns.** Tree patterns are an interesting form of queries that define local properties of nodes in trees. Even though restricted, they are sufficiently expressive to describe nested structures such as nested pairs, as long as these are not recursive (such as nested lists). Tree patterns are thus promising for defining tree wrappers for information extraction.
We want to investigate pattern languages with context variables $C$ which denote segments of a tree. A typical example is:

$$\texttt{<html>}(\texttt{<body>}(C(\texttt{<p>}(\texttt{<i>}(\texttt{Name}), X_1, \texttt{<b>}(\texttt{Age}), X_2))))$$

This tree pattern requires relevant information to occur in a context of the form $\texttt{<html>}(\texttt{<body>}(C))$ where $C$ is variable.
We call a tree pattern *linear* if its variables occur at most once. Linearity restricts patterns to describe only local properties of trees, even though several locations can be mentioned simultaneously. Multiple occurrences of the same context variables could also be of interest. This expresses equality of contexts, which is a global property.
Alternatively, we can add regular constraints on contexts which state that a context must be recognized by some tree automaton. It is also easy to equip contextual patterns with a notion of outputs, and to support equalities as backward references as in a grep-like tool.
Algorithms for tree patterns can often be reduced to algorithms for tree automata. This holds for instance for

membership, inclusion or equivalence tests on linear patterns, even if additional regular constraints are permitted. With such implementation at hand, we can reduce wrapper combination to Boolean operations on tree automata.

**Tree Logic.** Monadic second-order logic over unranked trees (MSO) is an expressive query language. The Datalog sublanguage of MSO underlies the Lixto system for XML wrapper specification. It might be of interest to apply machine learning approaches to MSO to improve the adaptiveness of such wrapper specification system. We will consider extensions of Datalog or MSO for query specification. One idea is to employ equality constraints, i.e. parallelism constraints in the constraint language CLLS defined by Egg *et al* in [70]. This might be useful for information extraction where two relevant segements share structure. Operationally, we are interested in model checking rather then solving parallelism constraints. The former task is dramatically simpler than the later.
Onother idea is to investigate modal logics for wrapper specification. We will investigate the modal logics behind the ambient calculus, and study its expressiveness. We might also investigate feature logical approaches.
Finally, the constraint and logic approaches support incremental constructions or updates of tree wrappers by means of adding or removing constraints.

## 4.2 Learning Models of Tree Languages

Learnable classes of languages of unranked trees have not been studied. We want to investigate such classes and develop the corresponding algorithms. We hope that an algebraic notion of tree automata will help.

We suppose that a collection of tree-structured documents is given as input together with annotated positions of interest. The task is to learn a tree wrapper from this collection that can identify relevant positions in un-seen documents. Tree wrappers can be described by tree automata, pattern, or logic formulae.

One characteristic of real data is the presence of noise. Even in the case where data are automatically generated by database queries, noise can be introduced by the labeling process which is often hand made. Learning stochastic data models will provide a good way to overcome to problem of learning in the presence of noise. Another direction is to build procedures that are robust against noisy data. Therefore, noise has an impact either on the choice of the tree language representation or on the learning procedures. In our work we will investigate both cases.

**Learning tree automata or tree patterns.** We will define new algorithms that learn models of languages of semistructured data. We plan to extend results on grammatical inference for regular tree languages [153, 114] to the case of unranked trees, starting from the algebraic notion of automata that we will introduce. The algebraic nature might permit us to generalize from learning procedures known for words or ranked trees.

Inference of tree languages has two main applications. In the wrapper induction problem: the wrapper is a representation of a tree language or a representation of a tree transduction. In the problem of `DTD` or schemas inference: `DTD` and schemas are specific tree representations of tree languages.

The induction of tree automata from many sources of data is also a way to represent or unify (unknown) `DTD` or schema in one unique object. This might be useful for querying heterogeneous data.

**Learning logical models of tree languages** Maybe, a unified framework for different approaches on the definition of tree languages is the use of logics and constraints. For instance, monadic second-order logic of unranked trees can express `XPath` queries. The constraint approach supports modular enrichment, that might be suitable for processes that frequently update. Even in a general way, if there is no hope to obtain good properties from the machine learning point of view, the constraints approach surely supports heuristics. Inductive logic programming provides good illustrations of how well chosen heuristics give interesting results in machine learning. To this aim, we want to reuse the ideas developed in the GloBo system [172]. The system proceeds in an ascending way, *i.e.* through a generalization process, starting from very specific representations, that exactly fit with the data, to more general ones. This approach needs the formalism to support the computation of least upper bounds between systems of constraints. We think that CLLS constraints, or some restrictions of them, can support these operations.

## 4.3 Wrapper Induction

**From sequences to trees.** Most of the wrapper induction systems work on string data and from the machine learning point of view, some interesting ideas have been exploited in such systems. As a first step, a wrapper

induction system that takes into account the tree structure of data can realise the lift from strings to trees. For instance, string regular expression based wrappers could be translated into wrappers based on tree patterns or regular expressions. Alternatively, wrappers that classify delimiters of texts (in classes `start of extraction` or `end of extraction` like in BWI [76]) could be adapted in order to classify positions in trees.

The lift from string to trees often relies on machine learning procedures that are able to infer representation of tree languages like tree patterns or tree automata. Therefore, such an extension can be considered as an application of tree languages induction as explained below (Section 4.2).

**Wrapper combination.** In real situations, data often stem from multiple sources. For instance data from several web sites or web services are given as input of a mediator system. It is worth noting that data are heterogeneous or contain natural division of their features. For instance XML data give both structural and textual information. The combination of many learning algorithms can be used either to solve the problem of learning from several sources of data, or to solve the problem of learning from heterogeneous data or to reach better performance. The purpose here is to use recent work on machine learning —particularly in the application framework of text classification, in the task of information extraction.

The first key point is to note that information extraction can be closely related to classification when appropriate encoding have been made. Therefore, techniques commonly used in classification can be used in wrapper induction. Another point to note is that we are faced with a small amount of tagged data (labeled data) and a large amount of untagged data (unlabeled data). This corresponds to the semi-supervised learning framework.

In the presence of heterogeneous data, a combination of learning algorithms can be considered in several ways. Learning while combining machine learning tools: *e.g.* first use wrapper induction to discover pieces of relevant parts of texts in semistructured data then use wrappers on natural language data. Learning while combining several views on the same data, like in the co-training paradigm: *e.g.* we can build a learning model from structural information only, a learning model from text data, and combine them. To achieve better performance, we can also combine learning algorithms using boosting methods.

## 4.4 Software Development

**Tree Automata Library.** Most of the algorithmic manipulations of semistructured data that are involved in wrapper induction or wrapper applications rely on primitive operations on sets of trees, patterns or constraint systems. As explained in Section 4.1, we need for instance fast algorithms to check basic properties as inclusion, membership, equality for tree automata, and matching, implication, unification in the constraint formalism. We will develop a library for such operations maybe as a continuation of the software described page 15, depending on the chosen data model.

**Wrapper Induction.** The target of the first class of learning algorithms we want to develop is wrappers for data intensive Web servers, cgi-based web servers or Web services, that is sets of `XML` or `HTML` documents whose structure is quite uniform. This will be the starting point of a more general tool.

As a first aim, the underlying structure to model and manipulate data will be simple pattern languages. The induction process will rely on tree automata inference tools [153, 33]. With the help of the toolbox, we want to improve the simple wrapper implementation while considering more intricate classes of tree languages and more specific learning algorithms.

The Lixto approach is also one of the starting point of this work and one aim is to reduce human interaction in the construction of Lixto wrappers and to take noise (errors) into account.

**Wrapper combinations.** Wrappers automatically built from texts may not be accurate enough. Tree structures available in XML or HTML data can help as well as meta data about the documents (e.g. semantic informations). Therefore, we either need to merge several views or sources of the data or we need to combine several wrappers in order to build an accurate wrapper.

We have started a work based on the BWI system (boosted wrapper induction by Freitag and Kushmerick [76]). This tool will exploit ideas of ensemble methods and co-training.

### 4.5 Perspectives

**Semantic Web.** Our approach for the design of adaptive information extraction systems is modular. Moreover the use of different views over data and documents is essential in the present project. Therefore our approach should be sufficiently flexible so that it can integrate semantic information on the web once available. This means that our approach could be extended to handle extra semantic information when available and also could be extended to the new standards for data and documents.

**Link Structure.** The proposal is focused on tree structures. We did not consider the (hyper)link structure. But the link structure has to be considered in further works. Indeed, the link structure is an informative extra-knowledge about data and documents which could be used by information extraction systems. Also, the information to be extracted could be distributed over different sources and the link structure could be helpful to find these different sources.

**Syntactic and Semantic Structures.** Today, most systems for information extraction and information retrieval consider texts as bags of words or, at best, sequences of lexical items. Sometimes extra information in the form of flat syntactic or semantic tags associated with the items are used. But the syntactic and the semantic structures of sentences or of the whole text are seldom used, mostly because of computational complexity problems. A notable exception is the domain of question answering (see Section 5.1) for which the trivial "key words" or "bag of words" approaches are clearly insufficient. As a matter of fact, most questions require a precise identification of the syntactic and semantic roles to be understood and answered correctly. A classical example is "Who killed Lee Harvey Oswald ?" which should not be answered "John Fitzgerald Kennedy" ! A complete analysis of the question and of the candidate answers in a text are thus necessary to reach a good level in the domain of question answering. We suspect that deep analyses of (portions of) texts are to be more and more frequent for applications requiring a high level of understanding. This evolution is another possible source of tree structures as, of course, the syntactic structure of a sentence is usually displayed as a decorated tree.

In some of our previous works (see Section 3.2) we also argued that semantic information about words and the sentence they constitute were indirect indications about its syntactic structure. This is a direct consequence of the Principle of Compositionally, which states that the meaning of a sentence only depends of the meaning of the individual words it is made of and of its syntactic structure. Semantic representations themselves do not necessarily take the form of trees. But semantic representations can be considered as filters for some possible syntactic structures. Thus the availability of semantic resources is another potential (indirect) provider of tree-like structures.

## 5 Environment and Collaborations

We want to improve on learning tools by accounting for available tree structures, which become more explicit with new document formats such as `docbook`, `XML` or `SGML`. This can be done in the context of wrapper induction but tree structures are also useful in other application domains. We discuss these applications in Section 5.1. The next four sections first present the main planned collaborations of the MOSTRARE project, some related INRIA projects and the collaborations of our research teams.

### 5.1 Satellite research

**Text classification.** Tasks of information extraction and text classification have several common points. When data are of diverse structures and heterogeneous, text classification can be a preprocessing task. It becomes easier to build one wrapper per class rather than a global wrapper. On the contrary, information extraction systems can serve classification tasks because they furnish preprocessed data that are now described by well chosen parts of texts rather than full texts. There is therefore at least two reasons to reexamine the text classification procedure in the context of semistructured data. We want to combine our experience and tools developed in the wrapper induction project to improve classification (see the ACI-MDD project). We also want to reuse our algorithms for text classification in the wrapper induction framework. We have developed an algorithm for text classification in the presence of a small amount of information from one class (the positive class) and a large pool of unlabeled

data (see Section 3.2). We want to adapt this procedure in the case of semistructured data. The first idea is to use wrappers to extract information from structured texts in order to reduce the dimension of the classification problem. For instance, one can extract only the introduction and the conclusion of texts and then combine several or apply one text classification procedure. Another way is to use co-training techniques and combine one classifier built using the structural information and another classifier built using textual information. The first classifier will rely on techniques and tools developed for wrapper induction.

**Information Retrieval.** Classical Information Retrieval methods were developed for unstructured documents only. Query languages for semistructured documents were developed without taking care of the content of documents. As XML is about to become the standard format for semistructured documents, there is an increasing need for appropriate Information Retrieval methods. Fulltext retrieval should allow for selection of appropriate parts of a document in response to a query, such as by returning a section or a paragraph instead of the complete document. Query languages should be extended to handle search by content operators. Thus there is extensive research on XML Information Retrieval (see for instance SIGIR conferences and associated workshops), one proposition is for instance the `Xirql` system [80]. We have already mentioned connections between Information Extraction and Information Retrieval in Section 2.1. we are member of a research project (ACI-MDD) in which semistructured Information Extraction and semistructured Information Retrieval should fertilize each other.

**Question Answering on the Web.** Sometimes, to satisfy a request, a precise factual answer is required rather than the reference of a text which contains this answer. The task of searching or inferring such precise factual answer from a corpus is known as "Question Answering". This domain is now mature enough to scale to the Web as a corpus ([112]) and various systems are already available or under development (among the best known projects are: "mulder", "answerbus" and "Webclopedia"). The strategy used in such systems consists in a sequence of modules, achieving sub-tasks including the following:

1. syntactic analysis of the question;
2. classification of the question into one of possible predetermined classes;
3. transformation of the question into requests to search engines;
4. extraction of candidate answers from the texts proposed by the search engines (usually with regular patterns deduced from the question);
5. selection of the answer among the candidate ones.

As precise answers are required, a full syntactic analysis of the question (sub-task 1) is useful to distinguish, say: "Who Lee Harvey Oswald killed ?" from "Who killed Lee Harvey Oswald ?". This analysis produces a syntactic tree of the question. Once this analysis is achieved, the following sub-tasks will be all the more efficient as they take advantage of this syntactic structure. So, at least sub-tasks number 2, 3 and 4 deal with structural information. And all of them would also benefit from a machine learning approach instead of hand-made rules. First rudimentary steps in this direction have already been tried ([5, 148, 94, 177]) but none of them really takes into account the tree structure of the question in the learning process. The domain of Question Answering on the Web, of growing interest, could then greatly benefit from our project. We are member of a research group on the subject.

## 5.2 MOSTRARE Collaborations

**ACI-MDD** We are involved in a French research project "ACI masse de données – ACI-MDD – Accès au Contenu Informationnel pour les Masses de Données et Documents". This research project (2003–2006) is directly related with the MOSTRARE project. The aim of the project is the design of algorithmic tools for Information Retrieval, Information Extraction and Text Classification for semistructured documents. Our partners are research centers on machine learning: Patrick GALLINARI (LIP6) and Marie-Christine ROUSSET (LRI and GEMO INRIA project).

**LIXTO** The Lixto company is developing the Lixto information extraction tool; it is a spin-off of Georg Gottlob's research group at the TU-Wien. Lixto relies on tree wrapper specification with Datalog programs over unranked trees. We intend to cooperate with the Lixto people on how to incorporate machine learning techniques into Lixto-like systems.

**XRCE** A closely related project is the project "Intelligent Wrapper Learning Tools" in the Xerox Research Centre Europe XRCE in Grenoble. The IWrap tools are used for generation of wrappers for accessing heterogeneous information sources. The learning program copes mainly with the "automatic" `HTML` files. We have initiated contacts with Boris Chidlovskii which is the leader of this project and collaborations are foreseeable.

**Regional companies** We have contacts with ALICANTE and ARCHIMED. ALICANTE is a company in the EURASANTE park. This company is concerned with information technology for healthcare professionals. The design of a semi-automatic tool for business intelligence for healthcare professionals is matter of a collaboration. ARCHIMED designs information systems for libraries and university web sites with `XML` technology.

**Mobyle** We are involved in a proposal "ACI IMPbio – mobyle – ". The aim of the Mobyle project is to build a web services integration system for structural bioinformatics. This requires to analyse information returned by web services, extract and communicate data in structured standard form (maybe to discover). Because of the large number of web services, building wrappers by hand is not feasible. Five teams are involved in this project, three in bioinformatics and two in computer science.

**RIP-WEB** We are member of a French research group on Question Answering (RIP-WEB: Recherche d'Information Précise sur le WEB: http://www.limsi.fr/Individu/monceaux/RIP-Web/rip-web.html) whose leader is Brigitte GRAU, in LIMSI, and whose purposes include to evaluate what machine learning techniques can bring to Question Answering systems.

## 5.3 Related INRIA Projects

The four most related projects are the following:

**Gemo – RU Futurs** Data handled by information systems is increasingly complex, distributed, heterogeneous, replicated, changing, and multi-formatted. The goal of the Verso project is to study fundamental problems and develop novel techniques for the management of heterogeneous, distributed, evoluting and complex databases. Specific research topics are: querying `XML` documents and heterogeneous databases, query optimization, Active `XML`, data modeling. And a new research direction on the creation and mining of thematic data collections.

**Orpailleur – RU Lorraine** The general goal is the design of intelligent systems: knowledge systems, information systems, and data mining systems. The main research topics are: Knowledge representation and Knowledge management, data mining and text mining, and semantic Web.

**Texmex – RU Rennes** The goal is the efficient exploitation of large multimedia databases with three application domains: image retrieval, description for texts and images, and the addition of semantic capacities to the textual search engines.

**Exmo – RU Rhône Alpes** The goal of Exmo is the development of theoretical and software tools for qualifying the manipulation of knowledge chunk. The main topics are: semantic properties in knowledge representation language translation, semantic adaptation of multimedia documents and model alignment for interoperability.

Also the MOSTRARE is related with INRIA projects on knowledge systems and knowledge management such as ACACIA – RU Sophia Antipolis, on the web such as WAM – RU Rhône-Alpes. Tree automata and constraints are a common tool with the project SECSI – RU Futurs – on security of information systems. The tools developped in the MOSTRARE project should be integrated in the platform of the french corporate action SYNTAX.

## 5.4 Other and Previous Collaborations

**DATADIAB** is a project with the research center CERIM – LILLE which is specialized in biomathematics and medical information technology. It is an "ACI télémédecine et technologies pour la santé". The objective is the prediction of complications for diabetic patients. The project was closed in November 2002.

**INDANA** our partners are research centers on machine learning (LIP6 and LRI in Paris) and medical research centers (SPIM – Paris and SPC – Lyon). It is an "ACI télémédecine et technologies pour la santé" and an INSERM project. INSERM is the French Institute of Health and Medical Research. The objective is the design of a score for predicting cardiovascular risk.

**Learning categorial grammars** collaborations in this research theme are developped in the Cooperative Research Action GRACQ "Acquisition de Grammaires Catégorielles". The partners of this action are french research centers in computer science IRISA in Rennes, IRIN in Nantes, and LORIA in Nancy. Also collaborations have been developped with computer research centers in Europe: Programming Systems Lab Department of Computer Science Universität des Saarlandes Saarbrücken, Germany; Utrecht Institute of Linguistics (OTS), Nederland; faculty of Computer Science of Iasi, Romania.

**Tree automata** the book Tree Automata Techniques and Applications was written with D. Lugiez, F. Jacquemart and H. Comon. D. Lugiez is Professor in Marseille University, and collaborations are planned because he is currently working with S. D. Zilio on semistructured data. H. Comon is Professor in ENS Cachan and member of the SECSI INRIA project, and collaborations are planned with members of this project because MOSTRARE share common tools with the SECSI project. Among the other current collaborations, let us cite common work with W. Charatonik (MPI Saarbrücken until September 2002, now Wroclav University) on ambient logic and with Hitoshi Ohsaki (NAIST,Japan) on equational tree languages.

**Constraints** we cooperate with Manuel Bodirksy at the Humbold Universität in Berlin, Denys Duchier from Inria Loraine, Andreas Podelski for the Max Planck institut in Saarbrücken, Zhendong Su from the Berkeley University, Ralf Treinen from the ENS Cachan, and Mateu Villaret from the University of Girona.

**Machine learning** we have collaborations with machine learning research teams in the labs LIP6 and LRI in Paris in the project INDANA and in the specific action "AS STIC CNRS Gafodonnées". We also have regular collaborations with members of these teams. For instance, we have collaborations with M. Sebag LRI on boosting techniques and inductive logic programming. She has recently designed – common work with A. Termier and M. C. Rousset – TREEFINDER which is a system for searching frequent trees in a collection of labeled trees. We have scientific collaborations with P. Gallinari LIP6 whose recent works deal with structural information for information retrieval tasks. In the field of grammatical inference, we have scientific collaborations with the research team EURISE in Saint Etienne and LIM in Marseille. We are members of the specific action "AS STIC CNRS Apprentissage et Biologie".

**Natural Language Technology** We cooperate with Prof. Manfred Pinkal and his team from the University Saarbrücken and with Feiyu Xu from the German Center for Artificial Intelligence (DFKI).

## 5.5 Teaching and Scientific Diffusion

- TATA [43][8] is a numeric textbook on tree automata.
- S. TISON gave an invited tutorial "Tree automata and rewriting" in the conference RTA'2000 and in the conference RPC'2001. R. GILLERON – joint work with F. DENIS – gave a tutorial "Learning from labeled and unlabeled examples" in the conference ECML-PKDD'2001.
- R. GILLERON and M. TOMMASI – joint work with F. DENIS – wrote teaching reports on supervised learning and data mining[9]. They lectured in the *École des jeunes chercheurs en algorithmique et calcul formel 2002* on *Recent developments in Machine Learning*.

[8] available at `http://www.grappa.univ-lille3.fr/tata/`

[9] available in french at `http://www/grappa/index.php3?info=apprentissage`

- J. NIEHREN is one of the authors of the lecture script *Constraint programming in Oz for computational linguistics* [63]. Furthermore, he lectured in the European summer school on logic, language, and information (ESSLLI'99) on *Scope Underspecification and Processing*. He also tought in the *École des jeunes chercheurs en programmation 1999* on *Logiques et Contraintes*.

### 5.6 Scientific Animation

**program committees** S. TISON was co-chair of STACS'1999, chair of STACS'2000, chair of RTA'2002, PC member of RTA'2003, member of the editorial board of RAIRO - Theoretical Informatics and Applications anf of the "SPECIF Best thesis Award" jury.

R. GILLERON was PC member of RTA'1999 and was director of the scientific committee of CAP'2003 (french conference on machine learning). J. NIEHREN was PC member of RTA'2002 and PC member of the workshops on unification UNIF'2003, computational semantics (ICOS'2000), and on robust methods in analysis of natural language data ROMAN'2000. Jean-Marc Talbot was PC member of LPAR'2003 and is member of CSL'2004.

**conference organization** we have organized STACS'2000 in Lille.

**french scientific responsibilities** S. TISON is head of the STC team, vice-director of the LIFL (computer science department in Lille) and director of the doctoral school SPI of the university Lille 1 . R. GILLERON is head of the grappa team, member of the scientific committee of Lille 3 university, and member of the scientific committee of the RTP STIC CNRS "découvrir et résumer" (french national action of the CNRS on machine learning and data mining).

## References


[1] A.-C.Caron, D.Debarbieux, and Y.Roos. Modéles de données semi-structurées et contraintes d'inclusion. In *Extraction et Gestion de Connaissances*, volume 17, pages 461–472. Hermès, 2003.

[2] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web*. Morgan Kaufmann Publishers, 2000.

[3] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener. The lorel query language for semistructured data. *Journal of Digital Libraries*, 1(1):68–88, 1997.

[4] S. Abiteboul and V. Vianu. Regular path queries with constraints. In *Proc. of ACM Symposium on Principles of Database Systems*, 1997.

[5] E. Agichtein, S. Lawrence, and L. Gravano. Learning search engine specific query transformations for question answering. In *Proceedings of WWW10*, 2001.

[6] H. Aït-Kaci. *A Lattice-Theoretic Approach to Computation Based on a Calculus of Partially Ordered Type Structures*. PhD thesis, University of Pennsylvenia, Philadelphia, PA, 1984.

[7] E. Althaus, D. Duchier, A. Koller, K. Mehlhorn, J. Niehren, and S. Thiel. An efficient graph algorithm for dominance constraints. *Journal of Algorithms*, 2003. Special issue of SODA'2001. In Press.

[8] T. R. Amoth, P. Cull, and P. Tadepalli. On exact learning of unordered tree patterns. *Machine Learning*, 43(3):211–243, 2001.

[9] Y. André, F. Bossut, and A. Caron. On decidability of boundedness property for regular path queries. In *Proceedings of DLT'99*, Aachen, Germany, 1999. Development in Language Theory, World Scientific Publishing Co.

[10] D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21:46–62, 1980.

[11] D. Angluin. Inductive inference of formal languages from positive data. *Inform. Control*, 45(2):117–135, May 1980.

[12] D. Angluin. Inference of reversible languages. *J. ACM*, 29(3):741–765, July 1982.

[13] D. Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, Nov. 1987.

[14] H. Arimura, H. Sakamoto, and S. Arikawa. Efficient learning of semi-structured data from queries. In *Proc. 12th International Conference on Algorithmic Learning Theory*, volume 2225 of *Lecture Notes in Artificial Intelligence*, pages 315–331, 2001.

[15] R. Backofen, J. Rogers, and K. Vijay-Shanker. A first-order axiomatization of the theory of finite trees. *Journal of Logic, Language, and Information*, 4:5–39, 1995.

[16] A. Barrero. Unranked tree languages. *Pattern Recognition Letters*, 24(1):9–18, 1991.

[17] L. E. Baum. An equality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3:1–8, 1972.

[18] R. Baumgartner, S. Flesca, and G. Gottlob. Visual web information extraction with lixto. In *The VLDB Journal*, pages 119–128, 2001.

[19] M. Benedikt and L. Libkin. Tree extension algebras: Logics, automata, and query languages. In *Proceeding of the $9^{th}$ Logic in Computer Science Conference*, pages 203–214. IEEE Comp. Soc. Press, 2002.

[20] V. Benzaken, G. Castagna, and A. Frisch. Cduce: a white paper. Presented at the workshop PLAN-X: Programming Language Technologies for XML, 2002.

[21] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. 11th Annu. Conf. on Comput. Learning Theory*, pages 92–100. ACM Press, New York, NY, 1998.

[22] M. Bodirsky, K. Erk, A. Koller, and J. Niehren. Beta reduction constraints. In A. Middeldorp, editor, *$12^{\text{th}}$ International Conference on Rewriting Techniques and Applications*, number 2051 in Lecture Notes in Computer Science, pages 31–46, Utrecht, The Netherlands, May 2001. Springer.

[23] M. Bodirsky, K. Erk, A. Koller, and J. Niehren. Underspecified beta reduction. In *Proceedings of the 39th Annual Meeting of the Association of Computational Linguistics*, pages 74–81, Toulouse, France, July 2001.

[24] B. Bogaert, F. Seynhaeve, and S. Tison. The recognizability problem for tree automata with comparisons between brothers. In *Foundations of Software Science and Computation Structure*, pages 150–164, 1999.

[25] B. Bogaert and S. Tison. Equality and disequality constraints on direct subterms in tree automata. In P. Enjalbert, A. Finkel, and K. W. Wagner, editors, *9th Annual Symposium on Theoretical Aspects of Computer Science*, volume 577 of *Lecture Notes in Computer Science*, pages 161–171, 1992.

[26] A. Brüggemann-Klein, S. Herrmann, and D. Wood. Context and caterpillars and structured documents. In *Proceedings of Principles of Digital Document Processing; 4th International Workshop (PODDP'98)*, volume 1148 of *Lecture Notes in Computer Science*. Springer Verlag, 1998.

[27] A. Brüggemann-Klein, M. Murata, and D. Wood. Regular tree and regular hedge languages over unranked alphabets: Version 1. url: citeseer.nj.nec.com/451005.html.

[28] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu. A query language and optimization techniques for unstructured data. In *SIGMOD*, pages 505–516, Montreal, 1996.

[29] M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. In *Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, pages 6–11, Menlo Park, CA, 1998. AAAI Press.

[30] L. Cardelli. Describing semistructured data. *SIGMOD Record*, 30(4), 2001. Database Principles Column. ACM.

[31] L. Cardelli and G. Ghelli. A query language based on the ambient logic. In *Proceedings of the 9th European Symposium on Programming ESOP'01*, volume 2028 of *Lecture Notes in Computer Science*, pages 1–22. Springer Verlag, 2001.

[32] L. Cardelli and A. Gordon. Mobile ambients. *Theorical Computer Science*, 240:177–213, 2000.

[33] J. Carme, R. Gilleron, A. Lemay, A. Terlutte, and M. Tommasi. Residual finite tree automata. In *Proceedings of the seventh int. conf. developments in Language Theory DLT'03*, number 2710 in Lecture Notes in Computer Science, pages 171 – 182. Springer Verlag, 2003.

[34] A.-C. Caron, H. Comon, J. Coquidé, M. Dauchet, and F. Jacquemard. Pumping, cleaning and symbolic constraints solving. In *Proceedings, International Colloquium Automata Languages and Programming*, volume 820 of *Lecture Notes in Computer Science*, pages 436–449, 1994.

[35] A.-C. Caron, J. Coquidé, and M. Dauchet. Encompassment properties and automata with constraints. In Kirchner [103], pages 328–342.

[36] W. Charatonik, A. D. Gordon, and J.-M. Talbot. Finite-control mobile ambients. In *Proceedings of the European Symposium on Programming (ESOP'02)*, number 2305 in Lecture Notes in Computer Science, pages 295–313. Springer Verlag, 2002.

[37] W. Charatonik and J.-M. Talbot. The decidability of model checking mobile ambients. In *Proceedings of Computer Science Logic (CSL'01)*, volume 2142 of *Lecture Notes in Computer Science*, pages 339–354. Springer Verlag, 2001.

[38] W. Charatonik, S. D. Zilio, A. D. Gordon, S. Mukhopadhyay, and J.-M. Talbot. The complexity of model checking mobile ambients. In *Proceedings FoSSaCS'01*, volume 2030 of *Lecture Notes in Computer Science*, pages 152–167. Springer Verlag, 2001.

[39] B. Chidlovskii. Wrapping web information providers by transducer induction. In *Proc. European Conference on Machine Learning*, volume 2167 of *Lecture Notes in Artificial Intelligence*, pages 61 – 73, 2001.

[40] B. Chidlovskii. Schema extraction from xml collections. In *Proc. ACM/IEEE-CS Joint Conf. Digital Libraries*, 2002.

[41] W. W. Cohen, M. Hurst, and L. S. Jensen. A flexible learning system for wrapping tables and lists in html documents. In *Proceedings of the eleventh international conference on World Wide Web*, pages 232–241. ACM Press, 2002.

[42] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100 – 110, 1999.

[43] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: http://www.grappa.univ-lille3.fr/tata, 1997.

[44] B. Courcelle. Monadic second-order logic and hypergraph orientation. In *Proceedings of the Symposium on Logic in Computer Science*, pages 179–190. IEEE Computer Society Press, 1993.

[45] V. Crescenzi, G. Mecca, and P. Merialdo. Roadrunner: Towards automatic data extraction from large web sites. In *The VLDB Journal*, pages 109–118, 2001.

[46] M. Dauchet, A.-C. Caron, and J.-L. Coquidé. Reduction properties and automata with constraints. *Journal of Symbolic Computation*, 20:215–233, 1995.

[47] D.Debarbieux, Y.Roos, S.Tison, Y. Andre, and A.-C.Caron. Path rewriting in semistructured data. In *Proceedings of Words'03*, 2003.

[48] F. de Comité, F. Denis, R. Gilleron, and F. Letouzey. Positive and unlabeled examples help learning. In *ALT 99, Tenth International Conference on Algorithmic Learning Theory*, number 1720 in Lecture Notes in Artificial Intelligence, pages 219–230. Springer Verlag, 1999.

[49] F. de Comite, R. Gilleron, and M. Tommasi. Learning multi-label alternating decision trees from texts and data. In *Proceedings of Intern. Conference on Machine Learning and Data Mining*, number 2734 in Lecture Notes in Artificial Intelligence, pages 35–49. Springer Verlag, 2003.

[50] C. de la Higuera. Characteristic sets for polynomial grammatical inference. *Machine Learning*, 27:125–137, 1997.

[51] F. Denis, P. Dupont, Y. Esposito, and A. Lemay. Learning probabilistic residual finite state automata. In *ICGI'2002*, number 2484 in Lecture Notes in Artificial Intelligence, pages 77–91. Springer Verlag, 2002.

[52] F. Denis, R. Gilleron, A. Laurent, and M. Tommasi. Text classification and co-training from positive and unlabeled examples. In *Proceedings of the ICML-2003 workshop: the Continuum from labeled data to unlabeled data in Machine Learning and Data Mining*, pages 80 – 87, 2003.

[53] F. Denis, R. Gilleron, and F. Letouzey. Learning from positive and unlabeled examples. *Theorical Computer Science*, page to appear, 2003.

[54] F. Denis, R. Gilleron, and M. Tommasi. Text classification from positive and unlabeled examples. In *IPMU'02, 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2002.

[55] F. Denis, A. Lemay, and A. Terlutte. Learning regular languages using non deterministic finite automata. In *ICGI'2000, 5th International Colloquium on Grammatical Inference*, volume 1891 of *Lecture Notes in Artificial Intelligence*, pages 39–50. Springer Verlag, 2000.

[56] F. Denis, A. Lemay, and A. Terlutte. Learning regular languages using rfsa. In *ALT 2001*, number 2225 in Lecture Notes in Artificial Intelligence. Springer Verlag, 2001.

[57] F. Denis, A. Lemay, and A. Terlutte. Residual finite state automata. In *STACS 2001, 18th Annual Symposium on Theoretical Aspects of Computer Science*, number 2010 in Lecture Notes in Computer Science, pages 144–157. Springer Verlag, 2001.

[58] F. Denis, A. Lemay, and A. Terlutte. Residual finite state automata. *Fundamenta Informaticae*, 51(4):339–368, 2002.

[59] F. Denis, A. Lemay, and A. Terlutte. Some language classes identifiable in the limit from positive data. In *ICGI 2002*, number 2484 in Lecture Notes in Artificial Intelligence, pages 63–76. Springer Verlag, 2002.

[60] P. Devienne, J.-M. Talbot, and S. Tison. Solving classes of set constraints with tree automata. In G. Smolka, editor, *Proceedings of the* $3^{th}$ *CP*, Lecture Notes in Computer Science, oct 1997.

[61] P. Devienne, J.-M. Talbot, and S. Tison. Co-definite set constraints with membership expressions. In *Joint International Conference and Symposium on Logic Programming*, pages 203–214. IEEE Press, July 1998.

[62] P. Domingos. Machine learning. In W. Klosgen and J. Zytkow, editors, *Handbook of Data Mining and Knowledge Discovery*. Oxford University Press, 2003.

[63] D. Duchier, C. Gardent, and J. Niehren. Concurrent constraint programming in Oz for natural language processing. Available at `http://www.ps.uni-sb.de/Papers/abstracts/oznlp.html`, 2002.

[64] D. Duchier and J. Niehren. Dominance constraints with set operators. In *Proceedings of the First International Conference on Computational Logic*, number 1861 in Lecture Notes in Computer Science, pages 326–341. Springer, July 2000.

[65] D. Dudau, I. Tellier, and M. Tommasi. A learnable class of ccg from typed examples. In *Proceedings of the eigth conference on Formal Grammar*, pages 77–88, 2003.

[66] D. Dudau, I. Tellier, and M. Tommasi. Une classe de grammaires catégorielles apprenable à partir d'exemples typés. In *Proceedings of CAP 2003, Conférence en APprentissage.*, pages 169–184, 2003.

[67] D. Dudau-Sofronie, I. Tellier, and M. Tommasi. From logic to grammars via types. In *Proceedings of Learning Language in Logic (LLL) 2001*, pages 35–46, 2001.

[68] D. Dudau-Sofronie, I. Tellier, and M. Tommasi. Learning categorial grammars from semantic types. In *13th Amsterdam Colloquium*, pages 79–84, 2001.

[69] D. Dudau-Sofronie, I. Tellier, and M. Tommasi. A tool for language learning based on categorial grammars and semantic information. In *ICGI 2002*, volume 2484 of *Lecture Notes in Artificial Intelligence*, pages 303–305. Springer Verlag, 2002.

[70] M. Egg, A. Koller, and J. Niehren. The constraint language for lambda structures. *Journal of Logic, Language, and Information*, 10:457–485, 2001.

[71] M. Egg, J. Niehren, P. Ruhrberg, and F. Xu. Constraints over lambda-structures in semantic underspecification. In *Proceedings of the Joint 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics*, pages 353–359, Montreal, Canada, 1998.

[72] D. W. Embley, D. M. Campbell, Y. S. Jiang, S. W. Liddle, Y.-K. Ng, D. Quass, and R. D. Smith. Conceptual-model-based data extraction from multiple-record web pages. *Data and Knowledge Engeneering*, 3(31):227–251, 1999.

[73] K. Erk, A. Koller, and J. Niehren. Processing underspecified semantic representations in the constraint language for lambda structures. *Research on Language and Computation*, 1(1), 2002.

[74] K. Erk and J. Niehren. Parallelism constraints. In *International Conference on Rewriting Techniques and Applications*, volume 1833 of *Lecture Notes in Computer Science*, pages 110–126, Norwich, United Kingdoms, 2000. Springer.

[75] H. Fernau. Learning tree languages from text. In *Proc. 15th Annual Conference on Computational Learning Theory, COLT 2002*, pages 153 – 168, 2002.

[76] D. Freitag and N. Kushmerick. Boosted wrapper induction. In *AAAI/IAAI*, pages 577–583, 2000.

[77] D. Freitag and A. K. McCallum. Information extraction with hmms and shrinkage. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999.

[78] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In P. Vitányi, editor, *Proceedings of the 2nd European Conference on Computational Learning Theory*, volume 904 of *LNAI*, pages 23–37, Heidelberg, Mar. 1995. Springer.

[79] N. Fuhr. Xml information extraction and information retrieval, 2002.

[80] N. Fuhr and K. Großjohann. Xirql: A query language for information retrieval in xml documents. In *Proceedings of SIGIR*, pages 172–180, 2001.

[81] C. Gardent and B. Webber. Describing discourse semantics. In *Proceedings of the 4th TAG+ Workshop*, Philadelphia, 1998.

[82] M. Garofalakis, A. Gionis, R. Rastogi, S. Seshadri, and K. Shim. Xtract: A system for extracting document type descriptors from xml documents. In *Proc. ACM SIGMOD'2000*, pages 165 – 176, 2000.

[83] R. Gilleron, S. Tison, and M. Tommasi. Solving systems of set constraints using tree automata. In *10th Annual Symposium on Theoretical Aspects of Computer Science*, volume 665 of *Lecture Notes in Computer Science*, pages 505–514. Springer Verlag, 1993.

[84] R. Gilleron, S. Tison, and M. Tommasi. Solving systems of set constraints with negated subset relationships. In *Proceedings of the* $34^{th}$ *Symp. on Foundations of Computer Science*, pages 372–380, 1993. Full version in the LIFL Tech. Rep. IT-247.

[85] R. Gilleron, S. Tison, and M. Tommasi. Some new decidability results on positive and negative set constraints. In J.-P. Jouannaud, editor, *Proceedings, First International Conference on Constraints in Computational Logics*, volume 845 of *Lecture Notes in Computer Science*, pages 336–351. Springer Verlag, 1994.

[86] R. Gilleron, S. Tison, and M. Tommasi. Set constraints and automata. *Information and Control*, 149:1 – 41, 1999.

[87] E. Gold. Language identification in the limit. *Inform. Control*, 10:447–474, 1967.

[88] E. Gold. Complexity of automaton identification from given data. *Inform. Control*, 37:302–320, 1978.

[89] S. A. Goldman and S. S. Kwek. On learning unions of pattern languages and tree patterns in the mistake bound model. *Theorical Computer Science*, 288(2):237 – 254, 2002.

[90] P. B. Golgher, A. S. da Silva, A. H. F. Laender, and B. Ribeiro-Neto. Bootstrapping for example-based data extraction. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 371–378. ACM Press, 2001.

[91] G. Gottlob and C. Koch. Monadic queries over tree-structured data. In *Proceedings of the* $17^{th}$ *LICS*, Lecture Notes in Computer Science, Copenhagen, 2002. Springer Verlag.

[92] J. Goubault-Larrecq and K. N. Verma. Alternating two-way AC-tree automata. Technical Report LSV-02-1, Cachan, France, 2002.

[93] J. Hammer, H. García-Molina, S. Nestorov, R. Yerneni, M. Breunig, and V. Vassalos. Template-based wrappers in the TSIMMIS system. In *In Proceedings of ACM SIGMOD Conference on Management of Data*, pages 532–535, 1997.

[94] D. Hermjakob and E. H. Hovy. Knowledge-based question answering. In *Proceedings of SCI-02 conference*, 2002.

[95] T. Hofmann. Text categorization with labeled and unlabeled data: A generative model approach. In *Working Notes for NIPS 99 Workshop on Using Unlabeled Data for Supervised Learning*, 1999.

[96] H. Hosoya and B. Pierce. Xduce: A typed XML processing language. In *Proceedings of Third International Workshop on the Web and Databases*. (WebDB2000), 2000.

[97] C.-N. Hsu and M.-T. Dung. Generating finite-state transducers for semi-structured data extraction from the web. *Information Systems*, 23(8):521 – 538, 1998.

[98] T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of ICML-99, 16th International Conference on Machine Learning*, pages 200–209, 1999.

[99] T. Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*. Kluwer Academic Publishers, 2002.

[100] M. Kanazawa. *Learnable Classes of Categorial Grammars*. The European Association for Logic, Language and Information. CLSI Publications, 1998.

[101] R. M. Kaplan and J. Bresnan. Lexical-functional grammar: A formal system for grammatical representation. In J. Bresnan, editor, *The Mental Representation of Grammatical Relations*, pages 173–281. MIT Press, Cambridge, MA, 1982.

[102] M. Kay. Functional grammar. In C. Chiarello et al., editor, *Proceedings of the* $5^{th}$ *Annual Meeting of the Berkeley Linguistics Society*, pages 142–158, 1979.

[103] C. Kirchner, editor. *Proceedings. Fifth International Conference on Rewriting Techniques and Applications*, volume 690 of *Lecture Notes in Computer Science*, 1993.

[104] C. A. Knoblock, S. Minton, J. L. Ambite, P. J. Modi, N. Ashish, I. Muslea, A. G. Philpot, and S. Tejada. Modeling web sources for information integration. In *Proc. Fifteenth National Conference on Artificial Intelligence*, 1998.

[105] A. Koller, J. Niehren, and K. Striegnitz. Relaxing underspecified semantic representations for reinterpretation. *Grammars*, 3(2/3):217–241, 2000. Special issue of MOL'99.

[106] A. Koller, J. Niehren, and S. Thater. Bridging the gap between underspecification formalisms: Hole semantics as dominance constraints. In *Proceedings of the 11th Conference of the European Chapter of the Association of Computational Linguistics*, Apr. 2003. Available at `http://www.ps.uni-sb.de/Papers`.

[107] A. Koller, J. Niehren, and R. Treinen. Dominance constraints: Algorithms and complexity. In M. Moortgat, editor, *Proceedings of the Third International Conference on Logical Aspects of Computational Linguistics (Dec. 1998, Grenoble, France)*, volume 2014 of *Lecture Note in Artificial Intelligence*, pages 106–125, Heidelberg, 2001. Springer.

[108] R. Kosala, M. Bruynooghe, J. V. den Bussche, and H. Blockeel. Information extraction from web documents based on local unranked tree automaton inference. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003)*, 2003.

[109] R. Kosala, J. V. den Bussche, M. Bruynooghe, and H. Blockeel. Information extraction in structured documents using tree automata induction. In *Proc. of the 6th International Conference Principles of Data Mining and Knowledge Discovery (PKDD-2002)*, pages 299 – 310, 2002.

[110] N. Kushmerick. Finite-state approaches to web information extraction. In *Proc. 3rd Summer Convention on Information Extraction*, 2002.

[111] N. Kushmerick, D. S. Weld, and R. B. Doorenbos. Wrapper induction for information extraction. In *Intl. Joint Conference on Artificial Intelligence (IJCAI)*, pages 729–737, 1997.

[112] C. C. T. Kwok, O. Etzioni, and D. S. Weld. Scaling question answering to the web. In *Proceedings of WWW10*, 2001.

[113] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35 – 56, 1990.

[114] A. Lemay. *De l'apport des langages résiduels en inférence grammaticale de langages réguliers*. PhD thesis, LIFL, 2002.

[115] F. Letouzey, F. Denis, and R. Gilleron. Learning from positive and unlabeled examples. In *ALT'00, Eleventh International Conference on Algorithmic Learning Theory*, Lecture Notes in Artificial Intelligence, pages 71 – 85. Springer Verlag, 2000.

[116] C. Li, R. Yerneni, V. Vassalos, H. Garcia-Molina, Y. Papakonstantinou, J. Ullman, and M. Valiveti. Capability based mediation in TSIMMIS. In *Proceedings of the ACM SIGMOD Conference*, pages 564–566, 1998.

[117] L. Liu, C. Pu, and W. Han. XWRAP: An XML-enabled wrapper construction system for web information sources. In *ICDE*, pages 611–621, 2000.

[118] L. Liu, C. Pu, W. Tang, D. Buttler, J. Biggs, T. Zhou, P. Benninghoff, W. Han, and F. Yu. CQ: a personalized update monitoring toolkit. In *In Proceedings of ACM SIGMOD Conference*, pages 547–549, 1998.

[119] R. Milner, J. Parrow, and J. Walker. A calculus of mobile processes, I and II. *Information and Computation*, 100(1):1–40,41–77, 1992.

[120] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[121] T. Mitchell. Machine learning and data mining. *Communications of the ACM*, 42(11), 1999.

[122] M. Müller and J. Niehren. Ordering constraints over feature trees expressed in second-order monadic logic. *Information and Computation*, 159(1/2):22–58, 2000. Special issue of RTA'98.

[123] M. Müller, J. Niehren, and A. Podelski. Ordering constraints over feature trees. *Constraints, an International Journal*, 5(1–2):7–42, Jan. 2000. Special issue of CP'97.

[124] M. Müller, J. Niehren, and R. Treinen. The first-order theory of ordering constraints over feature trees. *Discrete Mathematics and Theoretical Computer Science*, 4(2):193–234, 2001.

[125] E. Moriya. On two way tree automata. *Information Processing Letters*, 50:117–121, 1994.

[126] Message understanding conference, 1995.

[127] M. Murata, D. Lee, and M. Mani. Taxonomy of XML schema languages using formal language theory. In *Extreme Markup Languages*, Montreal, Canada, 2001.

[128] R. Muskens. Talking about Trees and Truth-conditions. *Journal of Logic, Language and Information*, 10(4):417–455, 2001.

[129] I. Muslea, S. Minton, and C. Knoblock. Stalker: Learning extraction rules for semistructured, web-based information sources. In *Procceeding of AAAI-98 workshop on AI and Information Extraction*, 1998.

[130] I. Muslea, S. Minton, and C. A. Knoblock. Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems*, 4(1/2):93–114, 2001.

[131] A. Neumann. *Parsing and Querying XML Documents in SML*. PhD thesis, University of Trier, Germany, 1999.

[132] A. Neumann and H. Seidl. Locating matches of tree patterns in forests. In *Foundations of Software Technology and Theoretical Computer Science*, pages 134–145, 1998.

[133] F. Neven. Automata theory for xml researchers. Sigmod Record, Vol 31, number 3.

[134] F. Neven. Automata, xml and logic. In *Proceedings of CSL*, pages 2–26, 2002.

[135] F. Neven and T. Schwentick. Automata- and logic-based pattern languages for tree-structured data. In *Semantics in Databases*, pages 160–178, 2001.

[136] F. Neven and T. Schwentick. Query automata on finite trees. *Theorical Computer Science*, 275(633–674), 2002.

[137] F. Neven, T. Schwentick, and V. Vianu. Towards regular languages over infinite alphabets. In *Proceedings of MFCS*, volume 2136 of *Lecture Notes in Computer Science*, page 560. Springer Verlag, 2001.

[138] J. Niehren and A. Koller. Dominance constraints in context unification. In M. Moortgat, editor, *Proceedings of the 3rd International Conference on Logical Aspects of Computational Linguistics (Dec. 1998, Grenoble, France)*, volume 2014 of *Lecture Note in Artificial Intelligence*, pages 199–218, Heidelberg, 2001. Springer.

[139] J. Niehren and A. Podelski. Feature automata and recognizable sets of feature trees. In *Proceedings TAPSOFT'93*, volume 668 of *Lecture Notes in Computer Science*, pages 356–375, 1993.

[140] J. Niehren, A. Podelski, and R. Treinen. Equational and membership constraints for infinite trees. In Kirchner [103].

[141] J. Niehren, R. Treinen, and S. Tison. On rewrite constraints and context unification. *Information Processing Letters*, 74(1-2):35–40, Apr. 2000.

[142] J. Niehren and M. Villaret. Parallelism and tree regular constraints. In *9th International Conference on Logic for Programming Artificial Intelligence and Reasoning*, Lecture Notes in Artificial Intelligence. Springer, Oct. 2002.

[143] K. Nigam and R. Ghani. Analyzing the applicability and effectiveness of co-training. In *Proceedings of CIKM-00, Ninth International Conference on Information and Knowledge Management*, pages 86–93, 2000.

[144] K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

[145] H. Ohsaki. Beyond regularity: Equational tree automata for associative and commutative theories. In *Proceedings of 15th International Conference of the European Association for Computer Science Logic (CSL2001)*, volume 2142 of *LNCS*, pages 539–553, Paris (France), September 2001.

[146] H. Ohsaki and T. Takai. Decidability and closure properties of equational tree languages. In *Proceedings of 13th International Conference on Rewriting Techniques and Applications (RTA2002)*, volume 2378 of *Lecture Notes in Computer Science*, pages 114–128, 2002.

[147] J. Oncina and P. Garcia. Inferring regular languages in polynomial update time. In *Pattern Recognition and Image Analysis*, pages 49–61, 1992.

[148] D. Ravichandran and E. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the ACL conference*, 2002.

[149] R. Reischuk and T. Zeugmann. Analyzing the average-case behavior of conjunctive learning algorithms. Technical Report DOI-TR-153, Department of Informatics, Kyushu University, Kyushu, Japan, 1998.

[150] J. Rogers and K. Vijay-Shanker. Obtaining trees from their descriptions: An application to tree-adjoining grammars. *Computational Intelligence*, 10:401–421, 1994.

[151] A. Sahuguet and F. Azavant. Building intelligent web applications using lightweight wrappers. *Data Knowledge Engineering*, 36(3):283–316, 2001.

[152] Y. Sakakibara. Learning context-free grammars from structural data in polynomial time. *Theorical Computer Science*, 76:223 – 242, 1990.

[153] Y. Sakakibara. Efficient learning of context-free grammars from positive structural examples. *Information and Computation*, 97(1):23–60, Mar. 1992.

[154] R. E. Schapire. The boosting approach to machine learning: An overview. In *Proc. MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[155] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[156] R. F. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT-98)*, pages 80–91, New York, July 24–26 1998. ACM Press.

[157] L. Segoufin and V. Vianu. Validating streaming xml documents. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 53–64. ACM Press, 2002.

[158] H. Seidl, T. Schwentick, and A. Muscholl. Numerical document queries. In *Proceedings of the Symposium on Principles Of Database Systems*, pages 155–166, 2003.

[159] J. Siméon and P. Wadler. The essence of xml. In *Proceedings of the ACM Symposium on Principles of Programming Languages*. The ACM Press, 2003.

[160] G. Smolka. Feature constraint logics for unification grammars. *Journal of Logic Programming*, 12:51–87, 1992.

[161] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233–272, 1999.

[162] M. Takahashi. Generalizations of regular sets and their application to a study of context-free languages. *Information and Control*, 27:1–36, 1975.

[163] J.-M. Talbot, P. Devienne, and S. Tison. Generalized definite set constraints. *Constraints, an International Journal*, 5:25–39, 2000.

[164] I. Tellier. Meaning helps learning syntax. In *Proceedings of ICGI'98 International Colloquium on Grammatical Inference*, volume 1433 of *Lecture Notes in Artificial Intelligence*, pages 25–36. Springer Verlag, 1998.

[165] I. Tellier. Syntactico-semantic learning of categorical grammars. In *Proceedings of NeMLaP3/CoNLL98 ACL Workshop on Paradigms and Grounding in Language Learning*, pages 311–314, 1998.

[166] I. Tellier. Towards a semantic-based theory of language learning. In *12th Amsterdam Colloquium*, pages 217–222, 1999.

[167] I. Tellier. Semantic-driven emergence of syntax : the principle of compositionality upside-down. In *Proceedings of The Evolution of Language*, pages 220–224, 2000.

[168] J. W. Thatcher. Characterizing derivation trees of context-free grammars through a generalization of automata theory. *Journal of Comput. and Syst. Sci.*, 1:317–322, 1967.

[169] J. W. Thatcher. A further generalization of finite automata and an application to a decision problem. *American Mathematical Society*, 534(Notices 14), 1967. Abstract.

[170] J. W. Thatcher and J. B. Wright. Generalized finite automata with an application to a decision problem of second-order logic. *Mathematical System Theory*, 2:57–82, 1968.

[171] M. Tommasi. *Automates et contraintes ensemblistes*. PhD thesis, Laboratoire d'Informatique Fondamentale de Lille, 1994.

[172] F. Torre. Intégration des biais de langage à l'algorithme générer-et-tester - contributions à l'apprentissage disjonctif. Thèse de Doctorat, Laboratoire de Recherche en Informatique Université Paris-Sud France, jan 2000.

[173] C. J. Van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979.

[174] K. N. Verma. Two-way equational tree automata for ac-like theories: Decidability and closure properties. In *Proceedings of 14th International Conference on Rewriting Techniques and Applications*, volume 2706 of *Lecture Notes in Computer Science*, pages 180–196, 2003.

[175] XML Query web pages. `http://www.w3.org/XML/Query`.

[176] D. Yarowski. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings thirty-third meeting of the ACL*, pages 189 – 196, 1995.

[177] D. Zhang and W. S. Lee. Web based pattern mining and matching approach to question answering. In *Proceedings of TREC-11*, 2002.

[178] S. D. Zilio and D. Lugiez. Xml schema, tree logic and sheaves automata. In R. Nieuwenhuis, editor, *Proc. of RTA - Rewriting Techniques and Applications*, volume 2706 of *Lecture Notes in Computer Science*, pages 246–263. Springer Verlag, 2003.