A LOGIC-BASED APPROACH TO SEMANTIC INFORMATION EXTRACTION

Massimo Ruffolo
Exeura s.r.l. - ICAR-CNR
University of Calabria, 87036 Rende (CS), Italy

Marco Manna
Department of Matematics
University of Calabria, 87036 Rende (CS), Italy

Keywords: Information Extraction, Knowledge Representation, Logic Programming, Two-Dimensional Grammars, Knowledge Management.

Abstract: Recognizing and extracting meaningful information from unstructured documents, taking into account their semantics, is an important problem in the field of information and knowledge management. In this paper we describe a novel logic-based approach to semantic information extraction, from both HTML pages and flat text documents, implemented in the HιLεX system. The approach is founded on a new two-dimensional representation of documents, and heavily exploits DLP^+ - an extension of disjunctive logic programming for ontology representation and reasoning, which has been recently implemented on top of the DLV system. Ontologies, representing the semantics of information to be extracted, are encoded in DLP^+, while the extraction patterns are expressed using regular expressions and an ad hoc two-dimensional grammar. The execution of DLP^+ reasoning modules, encoding the HιLεX grammar expressions, yields the actual extraction of information from the input document. Unlike previous systems, which are merely syntactic, HιLεX combines both semantic and syntactic knowledge for a powerful information extraction.

1 INTRODUCTION

Existing systems for storing unstructured information such as document repositories, digital libraries, and Web sites, consist mainly of a huge amount of HTML pages or flat text documents, organized according to syntactic, semantic and presentation rules, recognizable only by human readers. Such repositories tend to be practically useless both for the vastness of the information they hold and the lack of machine readability. Moreover, they are unable to manage the actual knowledge that the information sources convey.

Recognizing and extracting relevant information automatically from these rapidly changing sources, according to their semantics, is an important problem in the information and knowledge management area.

In the recent literature a number of approaches for information extraction from unstructured documents have been proposed. An overview of the large body of existing literature and systems is given in (Eikvil, 1999; Feldman et al., 2002; Kuhlins and Tredwell, 2003; Laender et al., 2002; Rosenfeld et al., 2004). The currently developed systems are purely syntactic, and they are not aware of the semantics of the information they are able to extract.

In this work we present a logic-based approach, implemented in the HιLεX system, which combines both syntactic and semantic knowledge for a powerful and expressive information extraction from unstructured documents. Logic-based approaches to the information extraction problem are not new (Baumgartner et al., 2001a; Baumgartner et al., 2001b), however, the approach we propose is original. Its novelty is due to:

- The two-dimensional representation of an unstructured document. A document is viewed as a cartesian plan composed by a set of nested rectangular regions called *portions*. Each portion, univocally identified through the cartesian coordinates of two opposite vertices, contains a piece of the input document (*element*) annotated into an ontology.
- The exploitation of a logic-based knowledge representation language called DLP^+, extending DLP (Gelfond and Lifschitz, 1991) with object-oriented features, including classes, (multiple) inheritance, complex objects, types, which is well-suited for representation and powerful reasoning on ontologies. This language is supported by the DLV^+ system (Ricca et al., 2005), implemented on top of

Ruffolo M. and Manna M. (2006).
A LOGIC-BASED APPROACH TO SEMANTIC INFORMATION EXTRACTION.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - AIDSS*, pages 115-123
DOI: 10.5220/0002458601150123

DLV (Eiter et al., 2000; Eiter et al., 1997; Faber and Pfeifer, 1996; Leone et al., 2004).

- The use of an ontology, encoded in DLP^+, describing the domain of the input document. A concept of the domain is represented by a DLP^+ class; each class instance is a *pattern* representing a possible way of writing the concept and is used to recognize and annotate an element contained in a portion.
- The employment of a new grammar, named HιLεX grammar, for specifying the (above mentioned) patterns. HιLεX grammar extends regular expressions for the representation of two-dimensional patterns (like tables, item lists, etc.), which often occur in web pages and textual tabular data. The patterns are specified through DLP^+ rules, whose execution yields the *semantic information extraction*, by associating (the part of the document embraced by) each portion to an element of the domain ontology.

It is worthwhile noting that, besides the domain ontologies, HιLεX system uses also a *core ontology*, containing (patterns for the extraction of) general linguistic elements (like, e.g., date, time, numbers, email, words, etc.); presentation elements (like, e.g., font colors, font styles, background colors, etc.); structural elements (like, e.g., table cell, item lists, paragraphs, etc.) which are not bounded to a specific domain but occur generally.
The advantages of the HιLεX system over other existing approaches are mainly the following:

- The extraction of information according to their semantics and not only on the basis of their syntactic structure (as in the previous approaches).
- The possibility to extract information in the same way from documents in different formats. The same extraction pattern can be used to extract data from both flat text and HTML documents. Importantly, this is not obtained by a preliminary HTML-to-text translation; but it comes automatically thanks to higher abstraction due to the view of the input document as a set of logical portions.
- The possibility to obtain a "semantic" classification of the input documents, which is much more accurate and meaningful than the syntactic classifications provided by existing systems (mainly based on counting the number of occurrences of some keywords), and opens the door to many relevant applications (e.g., emails classification and filtering, skills classification from curricula, extraction of relevant information from medical records, etc.).

Distinctive features of the novel semantic approach to information extraction implemented in the HιLεX system, summarized above, allows a better digital contents management and fruition in different application field such as: e-health, e-entertainment, e-commerce, e-government, e-business.

Figure 1: Financial Yahoo Page.

The remainder of this work is organized as a by example explanation of the proposed approach. In particular: section 2 shows the two-dimensional document representation idea; section 3 describes the DLP^+ knowledge representation language and how ontologies are used to represent the semantics of information to be extracted and to give a *logic two-dimensional representation* of unstructured documents; section 4 describes the syntax and the semantics of the two-dimensional pattern specification grammar and the logic-based pattern recognition method exploiting it; finally, section 5 shows the architecture of the HιLεX system.

2 TWO-DIMENSIONAL REPRESENTATION OF UNSTRUCTURED DOCUMENTS

The two-dimensional representation of an unstructured document is the main notion, which the semantic information extraction approach, presented in this work, is based on. This notion is founded on the idea that an unstructured document can be considered as a cartesian plan composed by a set of nested rectangular regions called *portions*. Each region, univocally identified through the cartesian coordinates of two opposite vertices, contains a piece of the input document including an *element* of the information to be extracted. Information elements, organized according to syntactic, presentation and semantic rules of a language recognizable by a human reader, can be *simple* or *complex*. *simple* elements are characters, table cells, words (classified using its part-of-speech tag recognized using natural language techniques); *complex* elements are phrases, item lists, tables, paragraphs, text boxes obtained as composition of other

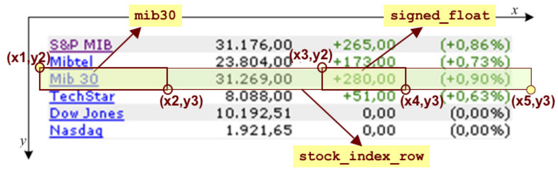

Figure 2: Example of `portions`.

simple or complex elements.

To better explain the idea of portion consider the web page depicted in Figure 1 (obtained from the Italian Yahoo financial portal) containing information about the stock exchange market. Suppose we would like to acquire, from this page, the table containing the stock index values and their variation (surrounded by a smooth etched box in Figure 1). A two-dimensional representation of data contained in the highlighted document region we are interested on (Figure 2), can be obtained by drawing on it an hypothetical cartesian plan. Each element of the table can be identified, in that plan, by suitable rectangular regions (*portions*).

For instance, in Figure 2, the stock index name "Mib 30" is a simple element which is contained in the portion identified by [(x_1, y_2),(x_2, y_3)]. In the same way, the signed float number representing the absolute variation of the "Mib 30" is contained in the portion [(x_3, y_2),(x_4, y_3)]. Since portions can be nested, the portion containing the complex element representing the concept of "stock index row" can be identified by the points [(x_1, y_2),(x_5, y_3)] and so on.

3 REPRESENTING KNOWLEDGE

The semantic information extraction approach implemented in the HιLεX system is based on the DLP^+ (Ricca et al., 2005) ontology representation language.

DLP^+ is a powerful logic-based language which extends Disjunctive Logic Programming (DLP) (Eiter et al., 2000) by object-oriented features. In particular, the language includes, besides the concept of relations, the object-oriented notions of classes, objects (class instances), object-identity, complex-objects, (multiple) inheritance, and the concept of modular programming by means of reasoning modules. This makes DLP^+ a complete ontology representation language supporting sophisticated reasoning capabilities.

Moreover, the DLP^+ ontology representation language is implemented on the DLV^+ system, a cross-platform development environment for knowledge modeling and advanced knowledge-based reasoning. The DLV^+ system (Ricca et al., 2005) permits to easily develop real world complex applications and allows to perform advanced reasoning tasks in a user friendly visual environment. DLV^+ seamlessly integrates the DLV (Eiter et al., 2000) system exploiting the power of a stable and efficient ASP solver (for further background on DLV and DLP^+see (Ricca et al., 2005; Eiter et al., 2000)).

In the HιLεX system the DLP^+ language is heavily exploited for the formal representation of the semantics of information to be extracted (employing suitable ontologies). Furthermore, DLP^+ allows the encoding of the *logic two-dimensional representation* of unstructured documents. Finally, DLP^+ reasoning modules (which are specialized DLP^+ logic programs) are exploited for the implementation of the logic-based pattern recognition method allowing the actual semantic information extraction.

More in detail, the elements of information to be extracted are modeled by using the DLP^+ class *element* which is defined as follows:

```
class element (type: expression_type,
   expression: string, label: string).
```

The three attributes have the following meaning:

- `expression`: holds a string representing the pattern specified by regular expressions or by the HιLεX two-dimensional grammar (described in detail in the following section), according to the `type` property. Patterns contained in these attributes are used to recognize the elements in a document.
- `type`: defines the type of the expression (i.e. `regexp_type`, `hilex_type`).
- `label`: contains a description of the element in natural language.

As pointed out in section 2, elements are located inside rectangular region of the input document called `portions`. Document portions and the enclosed elements are represented in DLP^+ by using the class *point* and the relation *portion*

```
class point (x: integer, y: integer).
relation portion (p: point, q: point, elem:
   element).
```

Each instance of the relation `portion` represents the relative rectangular document region. It relates the two points identifying the region, expressed as instances of the class `point`, and an ontology element, expressed as instance of the class `element`. The set of instances of the `portion` relation constitute the *logic two-dimensional representation* of an unstructured document.[1]

[1]This DLP^+ encoding allows to exploit the two-dimensional document representation on which the semantic information extraction approach proposed in this paper is based on.

The element class is the common root of two kind of ontologies, the `core ontology` and the `domain ontologies`. Every pattern encoding information to be extracted is represented by an instance of a class belonging to these ontologies.

In the following the structure of core and domain ontologies are described in details.

3.1 The Core Ontology

The core ontology is composed of three parts. The first part represents general simple elements describing a language (like, e.g., alphabet symbols, lemmas, Part-of-Speech, regular forms such as date, e-mail, etc.). The second part represents elements describing presentation styles (like, e.g., font types, font styles, font colors, background colors, etc.). The third part represents structural elements describing tabular and textual structures (e.g. table cells, table columns, table rows, paragraphs, item lists, texture images, text lines, etc.). The core ontology is organized in the class hierarchy shown below:

```
class linguistic_element isa {element}.
    class character isa {linguistic_element}.
        class number_character isa {character}.
        ...
    class regular_form isa {linguistic_element}.
        class float_number isa {regular_form}.
        ...
    class italian_lexical_element isa
        {linguistic_element}.
    class english_lexical_element isa
        {linguistic_element}.
         class english_lemma isa
             {english_lexical_element}.
        ...
    class spanish_lexical_element isa
        {linguistic_element}.
    ...
class presentation_element isa {element}.
    class font_type isa
        {presentation_element}.
    ...
class structural_element isa {element}.
    class table_cell isa
        {structural_element}.
    class separator isa
        {structural_element}.
    ...
```

Examples of instances of the `float_number` class are:

```
unsigned_float_number: float_number (type: regexp_type,
    expression:"(\d{1,3}(?>.\d{3})*,\d+)",
    label: "RegExp for unsigned float number").

signed_float_number: float_number (type: regexp_type,
    expression:"([+-]\s*\d{1,3}(?>.\d{3})*,\d+)",
    label: "RegExp for signed float number").

percentage: float_number (type: regexp_type,
    expression:"(\(?(?>[+-])?(?>(?>100(?>,0+)?)|
                (?:\d{1,2}(?>,\d+)?))%\)?)",
    label: "RegExp for percentage").
```

When in a document the regular expression characterizing a particular kind of float number is recognized, a document portion is generated and annotated w.r.t. the corresponding class instance.

3.2 Domain Ontologies

Domain ontologies contain simple and complex elements of a specific knowledge domain. The distinction between core and domain ontologies allows to describe knowledge in a modular way. When a user need to extract data from a document regarding a specific domain, he can use only the corresponding domain ontology. The modularization improve the extraction process in terms of precision and overall performances. Referring to the example of previous section, elements representing concepts related to the stock index market domain can be organized as follows:

```
class stock_market_domain isa {element}.
    class stock_index isa
        {stock_market_domain,
        linguistic_element}.
    class stock_index_cell isa
        {stock_market_domain,
        structural_element}
    class stock_index_row isa
        {stock_market_domain,
        structural_element}.
    class stock_index_table isa
        {stock_market_domain,
        structural_element}.
    class index_value isa
        {stock_market_domain, regular_form}.
```

Examples of instances of the `stock_index` class are:

```
mibtel: stock_index (type: regexp_type,
    expression: ``Mibtel'').
mib30: stock_index (type: regexp_type,
    expression: ``Mib30'').
dowJones: stock_index (type: regexp_type,
    expression: ``Dow Jones'').
```

When a regular expression characterizing a stock index is recognized in a document, a portion is generated and annotated w.r.t. the corresponding class instance.

4 A TWO-DIMESIONAL GRAMMAR FOR EXTRACTION PATTERNS SPECIFICATION

The internal representation of extraction patterns, in the HɩLεX system, is obtained by means of a two-dimensional grammar, founded on picture languages

(Chang, 1970; Giammarresi and Restivo, 1997), and allowing the definition of very expressive target patterns. Each pattern represents a two-dimensional composition of portions annotated w.r.t. the elements defined in the ontology. The syntax of the HiLεX two-dimensional grammar is presented in the following.

```
NEW_ELEMENT → GENERALIZATION | RECURRENCE | CHAIN |
TABLE
GENERALIZATION → GEN1 | GEN2 | GEN3
GEN1 → generalizationOf (arg: ARG1)
GEN2 → orContain_generalizationOf (arg: ARG1,
  inArg: ARG1, condition: CND)
GEN3 → andContain_generalizationOf (arg: ARG1,
  inArg: ARG1, condition: CND)
CND → coincident | notCoincident | null
RECURRENCE → recurrenceOf (arg: ARG3,
  range: RANGE, dir: DIR)
CHAIN → CHAIN1 (arg: ARG2, dir: DIR, sep: SEP)
CHAIN1 → sequenceOf | permutationOf
TABLE → TAB1 (arg: ARG2, range: RANGE,
  dir: DIR, sep: SEP)
TAB1 → sequenceTableOf | permutationTableOf
ARG1 → ARG2 | ARG3
ARG2 → [ LIST ]
ARG3 → BASE_ELEM
LIST → ARG3 , ARG3 LIST1
LIST1 → , ARG3 LIST1 | ε
RANGE → < NUM , NUM > | NUM | + | *
DIR → vertical | horizontal | both
SEP → ARG3 | null
```

According to the HiLεX grammar, a portion annotated w.r.t. a NEW_ELEMENT can be obtained by applying the composition language constructs to portions annotated w.r.t. basic ontology elements (BASE_ELEM). The semantics of each construct, together with some examples of usage, are presented in the following section.

GENERALIZATION: A portion annotated to basic ontology element (BASE_ELEM) can be re-annotated to the new ontology element (NEW_ELEMENT), by using the `generalizationOf` operator. The effect of this operator is a semantic rewriting generalizing the portion annotation.

Example 1 Consider the HTML document presented in section 2 whose elements are properly modelled in the core and domain ontologies. Let `unsigned_float_number` be an instance of the `float_number` class defined in the core ontology. A portion annotated as `unsigned_float_number` can be re-annotated as a `absolute_index_value` by using the following expression:

```
absolute_index_value: index_value (type:hilex_type,
    expression:"generalizationOf (
        arg: unsigned_float_number)",
    label:"Absolute value of a stock index" ).
```

The HiLεX grammar constructs `orContain_generalizationOf` and `andContain_generalizationOf` allow to define new annotations of existing portion on the basis of the semantics of contained portions. The generalization operators exploit the spatial (strict) containments of portions.

RECURRENCE: A portion annotated w.r.t. a NEW_ELEMENT, obtained by means of the `recurrenceOf` operator, consists in the concatenation, along a given direction, of a fixed number of portions annotated w.r.t. the same BASE_ELEM.

Example 2 Using the HiLεX `recurrenceOf` construct, a separator between two elements, contained in a document, can be defined as an instance of the `separator` class, constituted by a *null portion* (i.e. a portion without annotation having overlapped vertex along a coordinate) or the concatenation, in the horizontal direction, of an undefined number of portions annotated w.r.t. the `blank_char` element, defined as an instance of the core ontology `character` class.

```
sep_01: separator (type: hilex_type,
    expression : ``recurrenceOf (
        arg: blank_char,
        range: *, dir: horizontal)'',
    label: "Blank characters separator").
```

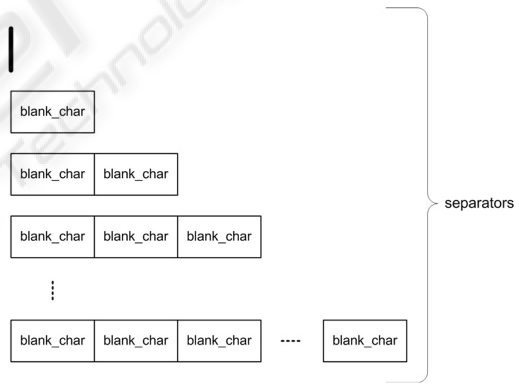

Figure 3: Example of `recurrence`.

CHAIN: A portion annotated w.r.t. a NEW_ELEMENT by using the `sequenceOf` and `permutationOf` operators, constitutes a chain of portions annotated w.r.t. BASE_ELEMs. In particular, a portion obtained by the application of the `sequenceOf` operator is a concatenation of at least two portions annotated w.r.t. BASE_ELEMs in a given direction and a fixed order, whereas, a portion obtained by using the `permutationOf` operator is a concatenation of at least two portions annotated w.r.t. BASE_ELEMs in a given direction, without an established order.

Example 3 A table row containing stock index variations can be represented using the HiLεX construct `sequenceOf` in the following way:

```
stock_index_row_01: stock_index_row( type:hilex_type,
    expression:"sequenceOf( arg: [stock_index,
        absolute_index_value, absolute_index_variation,
        percentage_index_variation],
        dir:horizontal, sep:sep_01 )",
    label:"Row containing stock index variations" ).
```

The figure 4 shows the portion annotated w.r.t an instance of the `the stock_index_row` class. It is constituted by an ordered sequence, in the horizontal direction, of portions annotated w.r.t. instances of the `stock_index` class, and the `unsigned_float`, `signed_float` and `percentage` instances of the `float_number` class. Between each couple of portions could be present a portion annotated w.r.t the element `sep_01`, an instance of the `separator` class, defined in the example 2. This expression considers only the semantics of the portions and their spatial positioning. Any reference to the document structure is required to recognize the concept of `stock_index_row`.

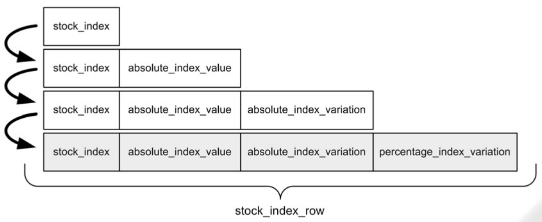

Figure 4: Example of `chain`.

TABLE: A portion annotated w.r.t. a `NEW_ELEMENT` can be defined by using the `sequenceTableOf` or `permutationTableOf` HιLεX operators, as a table of portions annotated w.r.t. <u>BASE_ELEM</u>s.
A portion, obtained from the `sequenceTableOf` operator, is composed by portions having a fixed composition along a direction, repeated a certain number of times along the other direction, whereas, a portion obtained from the `permutationTableOf` operator is composed by portions having an unordered composition along a direction, repeated with the same structure a fixed number of times along the other direction. This construct allows to recognize table in both HTML and text documents. In fact, portions provide an abstract representation of unstructured documents independent from the document format.

Example 4 The figure 5 depicts a portion annotated w.r.t. an instance of the `stock_index_table` class obtained by using the `sequenceTableOf` HιLεX grammar construct as shown in the following:

```
stock_index_table_01:stock_index_table( type: hilex_type,
    expression:"sequenceTableOf( arg: [stock_index,
        adsolute_index_value, absolute_index_variation,
        percentage_index_variation],
        range:<2,5>, dir:vertical, sep:sep_01 )",
    label:"table containing stock_index_row" ).
```

The instance `stock_index_table_01` represents a table of stock index variations composed

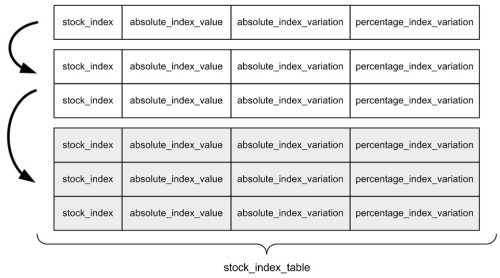

Figure 5: Example of `table`.

by a vertical sequence of at least 2 and at most 5 rows. Each row is a sequence of other portions annotated w.r.t. instances of the class `stock_index`, and the `unsigned_float`, `signed_float` and a `percentage` (i.e. a `stock_index_row`) instances of the `float_number` class.

4.1 Logic-Based Pattern Recognition

Extraction patterns expressed by means of the HιLεX two-dimensional grammar allow the actual semantic information extraction from unstructured documents. The pattern recognition mechanism is implemented encoding the HιLεX grammar expressions in DLP^+. In particular, each pattern is rewritten in a DLP^+ reasoning module as a set of rules exploiting the following basic operators able to manipulate points and portions.

```
relation strictFollow(p1: point, q1: point,
 elem1: element, p2: point, q2: point, elem2: element).

relation strictBelow(p1: point, q1: point,
 elem1: element, p2: point, q2: point, elem2: element).

relation minContain (p1: point, q1: point,
 elem1: element, p2: point, q2: point, elem2: element).

relation min_max_horizontalRecurrence(p: point,
 q: point, elem: element, min: integer, max: integer).

relation min_max_verticalRecurrence(p: point,
 q: point, elem: element, min: integer, max: integer).
```

The `strictFollow` operator, for example, is implemented by means of the DLP^+ rule presented in following:

```
strictFollow (P1, Q1, E1, P2, Q2, E2) :-
    portion (p:  P1, q:  Q1, elem:  E1),
    portion (p:  P2, q:  Q2, elem:  E2),
    P1:  point (y:  YP),
    Q1:  point (x:  X, y:  YQ),
    P2:  point (x:  X, y:  YP),
    Q2:  point (y:  YQ).
```

The semantics of the five basic operators is intuitively given in Figure 6.

The table containing the stock index variations, incorporated in the page presented in section 2, can be

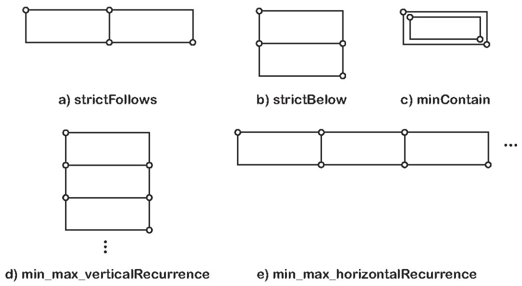

Figure 6: Basic operators.

extracted using the pattern presented in the example 4. The corresponding DLP^+ rewriting is shown below.

```
module(stock_index_table_01){
    portion(p:P1, q:Q7, elem:row_of_stock_index_table_01):-
        strictFollow(p1:P1, q1:Q1,
                     elem1:E1,
                     p2:P2, q2:Q2,
                     elem2:sep_01),
        strictFollow(p1:P2, q1:Q2,
                     elem1:sep_01,
                     p2:P3, q2:Q3,
                     elem2:absolute_index_value),
        strictFollow(p1:P3, q1:Q3,
                     elem1:absolute_index_value,
                     p2:P4, q2:Q4,
                     elem2:sep_01),
        strictFollow(p1:P4, q1:Q4,
                     elem1:sep_01,
                     p2:P5, q2:Q5,
                     elem2:absolute_index_variation),
        strictFollow(p1:P5, q1:Q5,
                     elem1:absolute_index_variation,
                     p2:P6, q2:Q6,
                     elem2:sep_01),
        strictFollow(p1:P6, q1:Q6,
                     elem1:sep_01,
                     p2:P7, q2:Q7,
                     elem2:percentage_index_variation),
        instanceOf(E1,stock_index).

    portion(p:P, q:Q, elem:stock_index_table_01):-
        min_max_VerticalRecurrence(p:P, q:Q,
            elem:stock_index_table_row_01,
            min:2, max:5).
}
```

The new portion, which structure satisfies the extraction pattern, is recognized by applying rules contained in the reasoning module shown above. These rules exploit the *logic two-dimensional representation* of unstructured document. The `row_of_stock_index_table_01` is a temporary instance of the class `stock_index_row`, having the same structure shown in the example 3. After the module execution such an instance is deleted.

The result of the extraction process is graphically shown in Figure 7. Figure 7 (a) depicts portions identified using patterns represented by regular expressions. Regular expressions are recognized by a document preprocessor based on a pattern matching mechanism. Figure 7 (b) and (c) show portions identified by the pattern recognizer exploiting the logic representation of the HıLεX grammar expressions.

Figure 7: Portions Extracted from the Yahoo Page.

It is worthwhile noting that patterns are very synthetic and expressive. Moreover, patterns are general in the sense that they are independent from the document format. This last peculiarity implies that the extraction patterns, presented above, are more robust w.r.t. variations of the page structure than extraction patterns defined in the previous approaches. For example, the table containing the stock index variations could appear wherever in the page. Furthermore, the same extraction patterns can also be used to extract information from flat text having the structure depicted in figure 8. The result of the extraction process on flat text is depicted in Figure 8 (a), (b), (c) having the same structure of Figure 7.

5 THE HıLεX SYSTEM

The architecture of the HıLεX system, implementing the semantic information extraction approach described in the previous sections, is represented in figure 9. The Knowledge Base (KB) of HıLεX stores the core and domain ontologies by means of the DLV^+ system persistency layer. The information extraction process is executed in three main steps: document pre-processing, pattern recognition, and pattern extraction. Each step is performed by a suitable architectural module.

In the first step a *Document Pre-Processor* takes in input an unstructured document and a query, containing the class instances names, representing the information that the user needs to extract. After the execution, the document preprocessor returns the two-dimensional logic document representation and a set of reasoning modules, constituting the input for the pattern recognizer. In particular, the *Document Pre-Processor* is composed of three sub-modules: *Query analyzer*, *Document Analyzer*, and *HıLεX Rewriter*. The *Query analyzer* takes in input the user query and explores the ontologies to identify the patterns to use for the extraction process. Patterns repre-

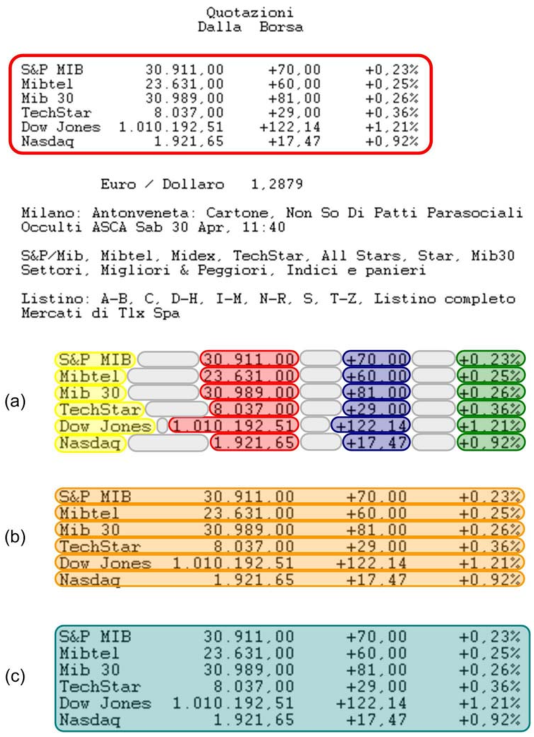

Figure 8: Flat Text Version of the Yahoo Page.

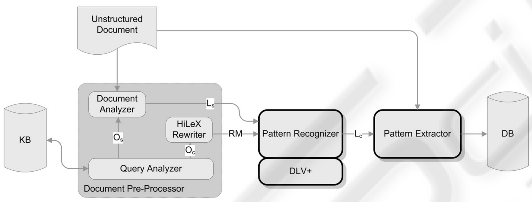

Figure 9: The Architecture of the HιLεX System.

sented through regular expressions (simple elements), together with the corresponding ontology instance names (named O_s in Figure 9) are the input of the *Document Analyzer* module. Patterns expressed using the HιLεX pattern representation grammar (complex elements) together with the corresponding ontology instance names (named O_c in Figure 9) are the input of the *HιLεX Rewriter*. The *Document Analyzer* applies pattern matching mechanisms to detect simple elements constituting the document and, for each of them, generates the relative *portion*. At the end of the analysis the *two-dimensional logic document representation* L_s is returned. The *HιLεX Rewriter* translates each pattern represented by the HιLεX two-dimensional grammar in a reasoning module containing logic rules suitable for pattern recognition. The output of the *HιLεX Rewriter* is a set of Reasoning Modules (RM) executable by the DLV^+ system. The translation is based on the operators able to manipulate portions described in Section 4.

The *HιLεX Rewriter* output (L_s) together with the *Document Analyzer* output (RM) is the input of the second step of the information extraction process, which is performed by the *Pattern Recognizer* module.

The *Pattern Recognizer* is founded on the DLV^+ system. It takes in input the logic document representation (L_s) and the set of reasoning modules (RM) containing the translation of the HιLεX patterns in terms of logic rules and recognize new complex elements. The output of this step is the *augmented logic representation* (L_c) of a unstructured document in which new document regions, containing more complex elements (e.g table having a certain structure and containing certain concepts, phrases having a particular mining, etc.), are identified exploiting the semantic knowledge represented in the ontologies. The pattern recognition is completely independent from the document format.

Finally, a *Pattern Extractor* takes in input the augmented logic representation of a document (L_c) and allows the acquisition of element instances (semantic wrapping) and/or the document classification w.r.t. the ontologies classes. Acquired instances can be stored in DLP^+ ontologies, relational and XML databases. Thus, extracted information can be used in other applications, and more powerful queries and reasoning tasks are possible on them. For example, the classification of the documents w.r.t. the ontology can be exploited for document management purpose.

6 CONCLUSIONS AND FUTURE WORKS

This work presents a novel, concrete, powerful and expressive approach to information extraction from unstructured documents. The approach, implemented in the HιLεX system, is grounded on two main ideas:

- The semantic representation of the information to extract by means of the DLP^+ ontology representation language, having solid theoretical foundations.
- The logic two-dimensional representation of documents allowing the definition of extraction patterns expressed by the HιLεX two-dimensional grammar.

Thanks to these ideas, the approach constitutes a decisive enhancement in this field. Unlike previous approach, the same extraction patterns can be used to extract information, according to their semantics, form both HTML and flat text documents. Furthermore, the HıLεX system can be used to implement a new generation of semantic wrappers. Many functions that will be available in the future "semantic web" technologies are turning into reality today with the HıLεX system.

Currently the approach is under consolidation and its theoretical foundations are under investigation and improvement. Future work will be focused on the consolidation and extension of the HıLεX two-dimensional grammar, the investigation of computational complexity issues from a theoretical point of view, the extension of the approach to pdf and other document formats, the exploitation of natural language processing techniques aimed to improve information extraction from documents with only textual contents.

7 ADDITIONAL AUTHORS

- Tina Dell'Armi. Exeura s.r.l. University of Calabria, 87036 Rende (CS), Italy *d*ellarmi@exeura.it
- Lorenzo Gallucci. Exeura s.r.l. Exeura s.r.l. University of Calabria, 87036 Rende (CS), Italy *g*allucci@exeura.it
- Nicola Leone. Department of Matematics; Exeura s.r.l. University of Calabria, 87036 Rende (CS), Italy, *l*eone@mat.unical.it
- Francesco Ricca. Department of Matematics, University of Calabria, 87036 Rende (CS), Italy, *r*icca@mat.unical.it
- Domenico Saccà. Exeura s.r.l.; DEIS; ICAR-CNR, University of Calabria, 87036 Rende (CS), Italy, sacca@unical.it

REFERENCES

Baumgartner, R., Flesca, S., and Gottlob, G. (2001a). Declarative information extraction, web crawling, and recursive wrapping with lixto. In *LPNMR '01: Proceedings of the 6th International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 21–41, London, UK. Springer-Verlag.

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

Chang, S.-K. (1970). The analysis of two-dimensional patterns using picture processing grammars. In *STOC '70: Proceedings of the second annual ACM symposium on Theory of computing*, pages 206–216, New York, NY, USA. ACM Press.

Eikvil, L. (1999). Information extraction from world wide web - a survey. Technical Report 945, Norweigan Computing Center.

Eiter, T., Faber, W., Leone, N., and Pfeifer, G. (2000). Declarative Problem-Solving Using the DLV System. In Minker, J., editor, *Logic-Based Artificial Intelligence*, pages 79–103. Kluwer Academic Publishers.

Eiter, T., Leone, N., Mateis, C., Pfeifer, G., and Scarcello, F. (1997). A deductive system for non-monotonic reasoning. In *Logic Programming and Non-monotonic Reasoning*, pages 364–375.

Faber, W. and Pfeifer, G. (since 1996). Dlv homepage.

Feldman, R., Aumann, Y., Finkelstein-Landau, M., Hurvitz, E., Regev, Y., and Yaroshevich, A. (2002). A comparative study of information extraction strategies. In Gelbukh, A. F., editor, *CICLing*, volume 2276 of *Lecture Notes in Computer Science*, pages 349–359. Springer.

Gelfond, M. and Lifschitz, V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3/4):365–386.

Giammarresi, D. and Restivo, A. (1997). Two-dimensional languages. In Salomaa, A. and Rozenberg, G., editors, *Handbook of Formal Languages*, volume 3, Beyond Words, pages 215–267. Springer-Verlag, Berlin.

Kuhlins, S. and Tredwell, R. (2003). Toolkits for generating wrappers – a survey of software toolkits for automated data extraction from web sites. In Aksit, M., Mezini, M., and Unland, R., editors, *Objects, Components, Architectures, Services, and Applications for a Networked World*, volume 2591 of *Lecture Notes in Computer Science (LNCS)*, pages 184–198, Berlin. International Conference NetObjectDays, NODe 2002, Erfurt, Germany, October 7–10, 2002, Springer.

Laender, A., Ribeiro-Neto, B., Silva, A., and Teixeira, J. (2002). A brief survey of web data extraction tools. In *SIGMOD Record*, volume 31.

Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., and Scarcello, F. (2004). The DLV System for Knowledge Representation and Reasoning.

Ricca, F., Leone, N., Dell'Armi, T., DeBonis, V., Galizia, S., and Grasso, G. (2005). A dlp system with object-oriented features. In *LPNMR '05: Proceedings of 8th International Conference on Logic Programming and Non Monotonic Reasoning*, Diamante, Italy.

Rosenfeld, B., Feldman, R., Fresko, M., Schler, J., and Aumann, Y. (2004). Teg: a hybrid approach to information extraction. In Grossman, D., Gravano, L., Zhai, C., Herzog, O., and Evans, D. A., editors, *CIKM*, pages 589–596. ACM.