Wrapping PDF Documents Exploiting Uncertain Knowledge

Sergio Flesca[1], Salvatore Garruzzo[2], Elio Masciari[3], and Andrea Tagarelli[1]

[1] DEIS, University of Calabria
{flesca, tagarelli}@deis.unical.it
[2] DIMET, University of Reggio Calabria
salvatore.garruzzo@unirc.it
[3] ICAR-CNR – Institute of Italian National Research Council
masciari@icar.cnr.it

Abstract. The PDF format represents the de facto standard for print-oriented documents. In this paper we address the problem of wrapping PDF documents, which raises new challenges in the information extraction field. The proposal is based on a novel bottom-up wrapping approach to extract information tokens and integrate them into groups related according to the logical structure of a document. A PDF wrapper is defined by specifying a set of group type definitions which impose a target structure to token groups containing the required information. Due to the intrinsic uncertainty on the structure and presentation of PDF documents, we devise constraints on token groupings as fuzzy logic conditions. We define a formal semantics for PDF wrappers and propose an algorithm for wrapper evaluation working in polynomial time with respect to the size of a PDF document.

1 Introduction

In the context of Information Extraction, *wrapping* is the process of extracting data containing information pertinent to a specific application domain, and organizing such data into a machine-readable format. Traditional wrapping refers to the Web environment; Web wrapping systems (e.g. [1, 2, 3, 4, 5, 6]) exploit markup tags in HTML pages to generate delimiter-based extraction rules [7].

However, HTML and XML are not the only formats to spread and exchange textual information for the purpose of making it accessible to companies and private users. A further and related kind of textual document refers to *print-oriented* formats, whose Acrobat PDF [8] is the de facto standard. A PDF document is described by a *PDF content stream*, which contains a sequence of graphical and textual objects located at precise positions inside the document pages. Such intrinsic print-oriented nature of PDF documents arises many issues that make the information extraction task particularly difficult.

As a motivating application scenario, consider PDF documents describing company balance sheets, like that of Fig. 1. Automatically extracting information from balance sheets is extremely useful to build data warehouses of financial data. Balance sheets are statements of the total assets and liabilities of

E. Dubois and K. Pohl (Eds.): CAiSE 2006, LNCS 4001, pp. 175–189, 2006.

Balance Sheet for XYZ Manufacturing Company
as of Dec 31 200

(All Figures in USD)

Assets	200	Liabilities and Owners' Equity	200
Current Assets	5,000,000	*Current Liabilities*	
Cash	500,000	Accounts Payable	4,000,000
T-Bills	1,000,000	Dividend Payable	2,000,000
Accounts Receivable	7,000,000	Taxes Payable	3,000,000
Total Current Assets	**13,500,000**	**Total Current Liabilities**	**9,000,000**
Inventory		*Long-term Liabilities*	
Raw Materials	825,000	Long-term Bank Loan	5,000,000
WIP	750,000	**Total Liabilities**	**14,000,000**
Finished Goods	1,200,000	Owners' Equity	
Total Inventory	**2,775,000**	Capital	20,000,000
Long-term assets		Retained Earnings	28,275,000
Land	30,000,000	**Total Net Worth**	**48,275,000**
Machinery	20,000,000		
Depreciation (machinery)	-5,000,000		
Intangible Assets			
Patents	1,000,000		
Total Long-term Assets	**46,000,000**		
Total Assets	**62,275,000**	**Total Liabilities + Net Worth**	**62,275,000**

Fig. 1. Excerpt of a sample balance sheet

an organization, at a particular date, and are usually available as PDF documents. Each company may encode balance data using different presentation styles (e.g. two-column or four-column layout, different instructions for text formatting etc.). The subjectivity in the layout structure that characterize even thematically similar balance sheets leads to "uncertainty" in the specification of the syntactic extraction rules.

To date, the problem of wrapping PDF documents has not been studied at all, in spite of its applicability to a wide variety of scenarios. No existing wrapper generation system is designed for print-oriented documents. At a first sight, it is evident that while most information extraction approaches can be extended to deal with the characteristics of the PDF format, the main hypothesis enabling most wrapping approaches is lacking: even if documents are yet automatically produced using data coming from company databases, each company can use a different program to encode data in a document and, consequently, the resulting layouts can be different.

In this paper we address the problem of extracting information from PDF documents by focusing on their spatial and content features. We propose a novel bottom-up wrapping approach which considers the complex schema of the information to be extracted and exploits logical fuzzy rule-based conditions on the extracted information. The combined use of bottom-up extraction and fuzzy conditions enables effectively handling uncertainty on the comprehension of the layout structure of PDF documents.

Section 2 introduces basic notions for dealing with PDF documents as sets of spatial tokens, and provides background on spatial relations and fuzzy set theory. Section 3 describes a novel framework for wrapping PDF documents and a semantics for PDF wrappers. Section 4 addresses the PDF wrapper evaluation issue and describes an algorithm for extracting a maximal token group from a source PDF

document, which works in polynomial time with respect to the size (number of tokens) of the source document. Section 5 contains concluding remarks.

2 Preliminaries

2.1 Fuzzy Sets

The sharp nature of classic set theory may often lead to scenarios in which the exact assignment of an object to a set is hardly obtained or unfeasible. *Fuzzy set theory* [9] takes into account the uncertainty due to subjective factors in data by introducing a smooth measure to state the place and role in the objects' class assignment. Given a set U, a fuzzy set A is defined by means of its *membership function* $\mu_A : U \mapsto [0..1]$, such that, for any element $x \in U$, the membership value $\mu_A(x)$ is defined as: $\mu_A(x) = 0$ if x does not belong to A, $\mu_A(x) = 1$ if x belongs to A, whereas $0 < \mu_A(x) < 1$ if x partially belongs to A.

A fuzzy atom is a formula $p(t_1, \ldots, t_n)$, where $t_1, \ldots, t_n$ are terms and p is a predicate symbol. Fuzzy predicates can be regarded as fuzzy sets of tuples, that is $\mu(p(t_1, \ldots, t_n)) = \mu_p(t_1, \ldots, t_n)$, where μ_p is the membership function of predicate p. Given an atom a, the truth value $\mu(a)$ ranges in $[0..1]$, and a fuzzy fact is an expression of the form $a \leftarrow \mu(a)$. In our setting, we mainly use built-in predicates, where the truth values of the ground atoms are pre-assigned. Truth value of conjunction and disjunction of atoms can be straightforwardly defined by means of *aggregation operators* [10]. Formally, given two fuzzy atoms $p(t_1, \ldots, t_n)$ and $q(t'_1, \ldots, t'_k)$, we have:

- $\mu(p(t_1, \ldots, t_n) \wedge q(t'_1, \ldots, t'_k)) = min(\mu(p(t_1, \ldots, t_n)), \mu(q(t'_1, \ldots, t'_k)))$,
- $\mu(p(t_1, \ldots, t_n) \vee q(t'_1, \ldots, t'_k)) = max(\mu(p(t_1, \ldots, t_n)), \mu(q(t'_1, \ldots, t'_k)))$,
- $\mu(\neg p(t_1, \ldots, t_n)) = 1 - \mu(p(t_1, \ldots, t_n))$.

As we shall explain in the following, fuzzy formulas enable modelling uncertainty in the wrapping process.

2.2 Spatial Documents

We refer to the concept of *token* as the basic element of a PDF document. A token is an atomic object (i.e. a textual element or an image), which is totally contained within a document page. The graphical representation of a token on the page layout takes up a certain room delimited by the token bounding box. We assume the presence of an alphabet Γ of token values.

Definition 1 (Document token). *A* document token *is a tuple* $\langle v, p, inf_x, inf_y, sup_x, sup_y \rangle$*, where* $v \in \Gamma$*,* p *is a page number,* inf_x *and* inf_y *(resp.,* sup_x *and* sup_y*) are the coordinates (pixels) of the top-left corner (resp., bottom-right corner) of the token bounding box. A* spatial document *is a set of document tokens.*

Reasoning about tokens and their relationships lies on the capability of defining suitable predicates to check syntactic as well as semantic properties, and to characterize the spatial properties between pairs of tokens.

Spatial predicates allow for capturing relationships between locations of tokens. We denote with $\texttt{spatialrelation}(t_1, t_2)$ a type of predicate that holds if there exists a specific spatial relation between tokens t_1 and t_2. More precisely, spatial predicates include *cardinal direction* predicates, namely *east*, *west*, *north*, *northeast*, *northwest*, *south*, *southeast*, *southwest*, and *topological* predicates such as *precedes* and $follows$.

Cardinal direction predicates are defined as fuzzy predicates, whose truth values depend on spatial relationships between tokens. For instance, the truth value of an atom $north(t_1, t_2)$, defined on tokens t_1 and t_2, depends on the amplitude of the angle formed by a line connecting the center of t_1 to the center of t_2 and the vertical axis of the document. By contrast, topological predicates are more simple since we assume they admit only true or false as truth value.

Content predicates are defined on the content of tokens. Some useful content predicates are listed below:

- $containsStr(t, s)$: holds if string s is contained in the text of token t;
- $isNumber(t)$: holds if token t represents a number;
- $value(t, s)$: holds if string s is the text value of token t;
- $regExp(t, e)$: holds if the text of token t matches a regular expression e;
- $concept(t, c)$: measures the relevance of token t with respect to an ontology concept c.

In the following section, we shall give evidence that spatial and content predicates are the basis for setting constraints on the construction of *token groups*.

3 Wrapping PDF Documents

3.1 PDF Extracted Information Model

In order to extract desired information, individual tokens are hierarchically organized into groups. A *token group* collects logically related tokens. For instance, in Fig. 1, tokens appearing in a line of a balance sheet are to be grouped together to compose a balance item. Balance items of type *Current Assets, Cash, T-bills, Accounts Receivable*, and *Total Current Assets* are then grouped together to compose the current asset group and so on. We assume the presence of a finite alphabet $\mathcal{T}$ of group types.

Definition 2 (Token group). *A* token group *is a pair* $\langle \tau, \gamma \rangle$*, where* $\tau \in \mathcal{T}$ *is a group type and* γ *is either a sequence* $\gamma = [\langle \tau_1, \gamma_1 \rangle, \ldots, \langle \tau_n, \gamma_n \rangle]$ *of token groups, or a single token.*

Given a token group $g = \langle \tau, \gamma \rangle$, we denote with $children(g)$ the set of the groups appearing in γ, and with $subgroups(g)$ the set of the groups that either appear in γ or are recursively contained in $subgroups(g')$ such that $g' \in \gamma$.

As stated in Definition 2, each token group is associated with a group type, and may consist of more subgroups each having a specific type. Thus, a compound token group is characterized in terms of the group types corresponding to its subgroups.

Definition 3 (Group content type). *Given a token group* $g = \langle \tau, \gamma \rangle$*, the* content type $cnt(g)$ *of* g *is either* $\tau_1, \ldots, \tau_n$ *if* $\gamma = [\langle \tau_1, \gamma_1 \rangle, \ldots, \langle \tau_n, \gamma_n \rangle]$*, or symbol* $\epsilon \notin \mathcal{T}$ *if* γ *is a single token.*

However, not all the token groups are well-suited to be considered as a result of the extraction task. In particular, we only consider "non-overlapping" groups, that is any group must not contain two identical subgroups. This must be true not only for the children of a group but also for all its descendants. For this purpose, the notion of *well-formed group* is next given.

Definition 4 (Well-formed group). *A token group* $g = \langle \tau, \gamma \rangle$ *is said to be* well-formed *if and only if there not exist two groups* $g', g'' \in subgroups(g)$*, with* $g' \notin subgroups(g'')$ *and* $g'' \notin subgroups(g')$*, such that* $children(g') \cap children(g'') \neq \emptyset$*.*

Proposition 1. *Let* $g = \langle \tau, \gamma \rangle$ *be a token group. If* g *is well-formed, each* $g' \in \gamma$ *is well-formed.*

Let $g' = \langle \tau', \gamma' \rangle$ and $g'' = \langle \tau'', \gamma'' \rangle$ be two token groups, where $\gamma' = [g'_0, \ldots, g'_n]$ and $\gamma'' = [g''_0, \ldots, g''_m]$. We say that g' *contains* g'' ($g' \supseteq g''$) if and only if:

- $g' = g''$, or
- $n = m$ and, for each $i \in [0..n]$, $g'_i \supseteq g''_i$, or
- $n > m$ and there exists a sequence of indexes $i_0, \ldots, i_m$ such that, for each $j \in [0..m]$, $i_j < i_{j+1}$ and $g'_{i_j} \supseteq g''_j$.

3.2 PDF Wrappers

In order to define a PDF wrapper we have to specify how to create a group starting from previously recognized subgroups, and which conditions these subgroups must satisfy. However, the print-oriented nature of PDF documents makes it impossible to specify tight conditions. To overcome this limitation we possibly exploit fuzzy conditions on token groupings.

Spatial and content predicates, introduced in Section 2.2 for characterizing properties and relationships between tokens, can be easily extended for token groups. The underlying idea is that a content predicate holds for a group if it holds for all the token within the group. Analogously, a spatial relationship between two groups holds if it holds for all the pairs of tokens within the groups. For instance, given two groups g_1 and g_2, relation $north(g_1, g_2)$ can be formalized as follows: $north(g_1, g_2) \rightarrow \forall t_i \in \gamma_1, t_j \in \gamma_2,\ north(t_i, t_j)$. This guarantees the *monotonicity* property for cardinal direction predicates. Extension of topological predicates to token groups can be made straightforwardly, since such predicates are not defined by means of fuzzy constraints.

A fuzzy constraint is a disjunction of conjunctions of content and spatial predicates. Formally, given a set of variables $\mathcal{V}$, a *fuzzy constraint* $c(\mathcal{V})$ is a formula $\bigvee_{i=0}^{n} c_i(\mathcal{V}_i)$, where each $c_i(\mathcal{V}_i)$ is a conjunction of group atoms $\bigwedge_{j=0}^{k_i} a_{i,j}(\mathcal{V}_{i,j})$ such that $\mathcal{V}_{i,j}$ is the set of all variables appearing in $a_{i,j}$, $\mathcal{V}_i = \bigcup_{j=0}^{k_i} \mathcal{V}_{i,j}$, and $\mathcal{V} = \bigcup_{i=0}^{n} \mathcal{V}_i$.

Let $\mathcal{V}$ and G be a set of variables and a set of groups, respectively, and let θ be a set of pairs x/g such that $x \in \mathcal{V}$ and $g \in G$. θ is a *group variable substitution* if and only if it does not contain two pairs x/g' and x/g'' such that $g' \neq g''$. Moreover, given a conjunction $c(\mathcal{V})$ of group atoms, a group variable substitution θ is said *ground* for $c(\mathcal{V})$ if and only if it contains a pair x/g for each $x \in \mathcal{V}$.

As is usual in standard logic programming, the application of a substitution θ to a variable x returns a group g if $x/g \in \theta$, or x itself otherwise. The result of the application of a substitution θ to an atom $a(x_0, \ldots, x_l)$, denoted with $\theta \circ a(x_0, \ldots, x_l)$, is an atom $a(\theta x_0, \ldots, \theta x_l)$. The application of a substitution to conjunction and disjunction of atoms is defined accordingly.

Proposition 2. *Let c be a conjunction of group atoms, and θ, θ' be ground group variable substitutions for c. If, for each $x/g \in \theta$, there is a pair $x/g' \in \theta'$ such that $g \subseteq g'$ then $\mu(\theta \circ c) \geq \mu(\theta' \circ c)$.*

Group Type Definitions. The specification of group content model requires the conditions, which selected subgroups must satisfy, are defined with respect to the type of a group.

Given an alphabet $\mathcal{T}$ of group types and an alphabet $\mathcal{V}$ of variables, an *annotated element name* is defined as a pair $\tau : x$, such that $\tau \in \mathcal{T}$ and $x \in \mathcal{V}$. An *annotated content model* on $(\mathcal{T}, \mathcal{V})$ is either a *one-unambiguous regular expression* [11] on annotated element names, or a pair $\#TOKEN : x$, where $\#TOKEN$ is the token content model and denotes a simple textual token.[1]

Given a group $g = \langle \tau, \gamma \rangle$ and an annotated content model e, the content type $cnt(g)$ is *valid* for e if:

- $cnt(g) = \epsilon$ and $e = \#TOKEN : x$, or
- $cnt(g) = \tau_1, \ldots, \tau_n \in L(e)$, where $L(e)$ denotes the language generated by e.

Given a group $g = \langle \tau, \gamma \rangle$ and an annotated content model e such that $cnt(g)$ is valid for e, the variable binding of γ with respect to e, denoted with $e(g)$, is a set of pairs $\{g_1/x_1, \ldots, g_n/x_n\}$ such that x_i is the variable implicitly associated to g_i parsing $\gamma = [g_1, \ldots, g_n]$ with e. Moreover, we denote with $\Theta(e, g)$ the set of all the subsets of $e(g)$ that are group variable substitutions. Notice that, in general $e(g)$ is not a group variable substitution, since a variable can be associated to multiple subgroups.

Definition 5 (Group type definition). *Let τ be a group type, e be an annotated content model on $(\mathcal{T}, \mathcal{V})$, and c be a fuzzy constraint on e. A* group type definition *is a tuple $\langle \tau, e, c \rangle$.*

[1] The "one-unambiguous" property for a regular expression allows for determining uniquely which position of a symbol in the expression should match a symbol in an input word, without looking beyond that symbol in the input word. For this reason, it is worth emphasizing that there is only one way by which a string of group types can be "parsed" using an annotated content model, thus an annotated content model implicitly associates a variable x_i to each subgroup $\langle \tau_i, \gamma_i \rangle$ in γ.

Roughly speaking, a group type definition is a complete specification of the content model, together with an additional fuzzy constraint. Moreover, a group type definition $\langle \tau, e, c \rangle$ refers to the type τ. In the following, we characterize the validity of a token group of type τ with respect to a group type definition referring to τ.

Definition 6 (Group validity). *Let $g = \langle \tau, \gamma \rangle$ be a token group and $gtd = \langle \tau', e, c \rangle$ be a group type definition. We say that g is* valid *with respect to gtd if $\tau = \tau'$ and $cnt(g)$ is valid for e.*

The syntactic validity of groups being extracted is not affected by the application of fuzzy constraints. The truth value of a token group can be computed by combining the truth values of its subgroups with the truth values of the fuzzy constraints according to the rules defined in Section 2.1. However, we have to define how a constraint c is grounded by the application of a group variable substitution θ. It may happen that applying θ to c the resulting formula is not ground, i.e. $\theta \circ c$ still contains some variables. In this case, the semantics of constraint intuitively requires that the conjunction in which a variable still appears will not be considered. The following example explains the above intuition.

Example 1. In Fig. 1, consider a group g whose description is as follows:

group	***type*** (τ)	***content*** (γ)
g	`Assets`	$[g_1, g_2]$
g_1	`Current_Assets`	t_1
g_2	`Intangible_Assets`	t_2

where g_1, g_2 are token groups and t_1, t_2 are tokens. Group g is associated with the group type definition $gtd = \langle \tau, e, c \rangle$ such that $\tau =$ `Assets`, $e =$ `Current_Assets` : CA, (`Inventory` : I|`Intangible_Assets` : IA), and $c = follows(CA, I)$. By parsing g with e, we have $e(g) = \{g_1/CA, g_2/IA\}$, which equals the set $\Theta(e, g)$ since it is a group variable substitution. By applying the substitution to c we obtain the non-ground constraint $follows(g_1, I)$. Thus, the truth value of g is provided by the minimum between the truth values of its subgroups (i.e. g_1, g_2).

Given a fuzzy constraint $c = c_0 \vee \ldots \vee c_n$ and a group variable substitution θ, the grounded version of $\theta \circ c$ is defined as $ground(\theta \circ c) = ground(\theta \circ c_0) \vee \ldots \vee ground(\theta \circ c_n)$, where $ground(\theta \circ c_i)$ is $\theta \circ c_i$ if $\theta \circ c_i$ is ground, $false$ otherwise.

Given a group $g = \langle \tau, \gamma \rangle$ which is valid for a group type definition $gtd = \langle \tau', e, c \rangle$, the truth value of g with respect to gtd is given by the truth value of the formula

$$\mu_{gtd}(g) = \mu \left(\bigwedge_{g' \in \gamma} \mu(g') \quad \wedge \quad ground(\theta \circ c), \forall \theta \in \Theta(e, g) \right),$$

where for each possible variable binding we compute its truth value with respect to all ground constraints $\theta \circ c$. Once computed the truth value of each conjunct, the overall value is obtained by using the *min* operator (cf. Section 2.1). Notice

that, the truth value of the overall formula propagates starting from the token value. Thus, token truth values bound the overall value of the formula and, as usual in classic logic, if any conjunct evaluates to 0 the overall formula will evaluate to 0 too.

PDF Wrapper Specification and Semantics. Faced with the above definitions, we are now ready to provide the notion of *PDF wrapper*.

Definition 7 (PDF wrapper). *A* PDF wrapper *is a tuple* $W = \langle \tau, G \rangle$*, where* $\tau \in \mathcal{T}$ *is the root group type and* $G = \{\langle \tau_0, e_0, c_0 \rangle, \ldots, \langle \tau_n, e_n, c_n \rangle\}$ *is a set of group type definitions such that* $\tau_i \neq \tau_j$*, for each* $i, j \in [0..n],\ i \neq j$*.*

Let $W = \langle \tau, G \rangle$ be a PDF wrapper and $\langle \tau_1, e_1, c_1 \rangle$, $\langle \tau_2, e_2, c_2 \rangle \in G$ be two group type definitions. We say that τ_1 *depends on* τ_2 if τ_2 appears in e_1, or there exists a group type definition $\langle \tau_3, e_3, c_3 \rangle$ such that τ_1 depends on τ_3 and τ_3 depends on τ_2. A PDF wrapper is said to be *non-recursive* if it does not contain two group types τ_1, τ_2 such that τ_1 depends on τ_2. In the following we consider only non-recursive PDF wrappers.

Given a wrapper $W = \langle \tau_0, G \rangle$ and a group type τ, we denote with $W(\tau)$ the group type definition of τ in G.

Definition 8 (Valid group). *Let* $W = \langle \tau_0, G \rangle$ *be a PDF wrapper and doc be a PDF document. A group* $g = \langle \tau, \gamma \rangle$ *on doc is* valid *for* W *if and only if:*

1. *g is well-formed, and*
2. $\tau = \tau_0$*, and*
3. *g is valid with respect to* $W(\tau)$*, and*
4. *for each subgroup* $g' = \langle \tau', \gamma' \rangle$ *of g,* g' *is valid with respect to* $W(\tau')$*.*

The set of all groups over doc valid for W *is denoted as* $\mathcal{G}(W, doc)$*.*

Broadly speaking, a *valid group* is essentially a well-formed group of tokens, which conforms to the schema defined by a wrapper. The reliability of the extracted data that are contained in a group g is substantially fuzzily measured by $\mu_{gtd}(g)$. However, since the truth value of a group depends both on its associated *gtd* and the truth values of its subgroups, in the following we denote with $\mu_W(g)$ the truth value of a group according to the definition of a wrapper W.

Given a PDF wrapper W and a document *doc*, there can be several different groups that are valid with respect to W. However, not all such groups are desirable as results of the evaluation of W on *doc*. The key-idea is to consider only groups that are "maximal", i.e. groups whose truth value is equal to or greater than a predefined threshold, and which are not contained inside other valid groups that meet the truth value requirement as well.

Definition 9 (Maximal group). *Let* $W = \langle \tau_0, G \rangle$ *be a PDF wrapper,* t *be a truth value threshold, and doc be a PDF document. A group* $g \in \mathcal{G}(W, doc)$ *is said to be* maximal *with respect to* t *if and only if:*

– $\mu_W(g) \geq t$, *and*
– *there not exists* $g' \in \mathcal{G}(W, doc)$ *such that* $g' \supseteq g$ *and* $\mu_W(g') \geq t$.

The set of all maximal groups with respect to t *is denoted as* $\mathcal{M}_t(W, doc)$.

Clearly, it is possible to consider only the maximal groups having the greatest truth value. However, this strategy may be computationally expensive; thus, we rather prefer to adopt a greedy approach for searching a maximal group with the greatest truth value.

4 PDF Wrapper Evaluation

In the proposed approach, the objective of evaluating a wrapper for a PDF document is to compute a maximal token group from that PDF document. We consider a restricted form of wrappers, called $\star$*-free wrappers*, which do not contain optionals or repeated subgroups in group definitions.

Definition 10 ($\star$-free wrapper). *Let* $W = \langle \tau_0, G \rangle$ *be a PDF wrapper.* W *is said to be* $\star$-free *if and only if, for each group type definition* $\langle \tau, e, c \rangle \in G$, e *does not contain* $*$ *or* ?.

Given a wrapper W, we denote with $W^\star$ the wrapper obtained by replacing each group type definition $\langle \tau, e, c \rangle$ in W with $\langle \tau, e^\star, c \rangle$, where $e^\star$ is the annotated content model obtained from e by removing each occurrence of symbols $*$ and ?. This operation allows us to consider the "kernel" of the content model defining W: indeed, as can be easily observed, any expression in $L(e^\star)$ is also in $L(e)$, that is $L(e^\star) \subseteq L(e)$.

The search for maximal groups works by repeatedly trying to add new subgroups to existing groups. This is achieved by expanding optional parts of group definition which have not been previously considered.

Definition 11 (Group expansion). *Let* $W = \langle \tau_0, G \rangle$ *be a PDF wrapper,* doc *be a PDF document,* $g = \langle \tau, \gamma \rangle$ *and* $g' = \langle \tau, \gamma' \rangle$ *be two groups from* doc, *and* $G(\tau)$ *be* $\langle \tau, e, c \rangle$. *We say that* g' *is an* expansion *of* g *if and only if* $g \subseteq g'$ *and each pair* $g_i/x_i \in e(g)$ *also belongs to* $e(g')$.

Lemma 1. *Let* W *be a PDF wrapper,* doc *be a PDF document, and* g, g' *two groups from* doc. *If* g' *is an expansion of* g *then* $\mu_W(g) \geq \mu_W(g')$.

Lemma 2. *Let* W *be a PDF wrapper,* doc *be a PDF document, and* t *be a truth value threshold. For each group* g *in* $\mathcal{M}_t(W, doc)$, *there exists a group* g' *in* $\mathcal{M}_t(W^\star, doc)$ *such that* g *is an expansion of* g'.

Lemma 3. *Let* W *be a PDF wrapper,* doc *be a PDF document, and* t *be a truth value threshold.* $\mathcal{M}_t(W, doc)$ *is empty if and only if* $\mathcal{M}_t(W^\star, doc)$ *is empty.*

Theorem 1. *Let* W *be a PDF wrapper,* doc *be a PDF document, and* t *be a truth value threshold. Checking if* $\mathcal{M}_t(W, doc)$ *is empty can be done in polynomial time with respect to the number of tokens in* doc.

Theorem 2. *Let W be a PDF wrapper, doc be a PDF document, and t be a truth value threshold. Checking whether a group g is in $\mathcal{M}_t(W, doc)$ is feasible in polynomial time with respect to the number of tokens in doc.*

4.1 A Fuzzy Algorithm for Extracting Maximal Token Groups

In this section we describe a PDF wrapper evaluation algorithm designed to extract a maximal token group from a PDF document (Fig. 2). Given a PDF document *doc* and a wrapper W for it, the maximal token group can be extracted according to the content models specified in W and to a predefined truth value threshold.

Initially, all the elementary token groups (i.e. groups of type *#TOKEN*) are extracted from the source document *doc*. These groups are simply computed by selecting all the tokens that satisfy the truth value threshold with respect to the associated constraints. Then, the $\star$-free wrapper $W^\star$ is applied to *doc* to extract all the $\star$-free token groups; among these, the token group with the maximum truth value is chosen and recursively "expanded" while the group being constructed satisfies the desired truth value threshold.

Expanding a token group consists substantially in adding some subgroups to its content, that is re-defining its original annotated content model including the content of other groups. For this purpose, a *normal form* for annotated content models is exploited.

Definition 12 (Normal form annotated content model). *An annotated content model $e = exp_1, \ldots, exp_n$ is in* disjunction-free normal form *if and only if, for each $i \in [1..n]$, exp_i is either an annotated element name $\tau : x$ or an expression of the form exp?, or exp∗, where exp is an annotated content model.*

Given a group g and an annotated content model e in disjunction-free normal form, g is *matched* by e if and only if $cnt(g)$ is valid for $e^\star$. In order to compute the expansion of a group g, the annotated content model e in disjunction-free normal form that matches g is considered. Let $e = exp_1, \ldots, exp_n$ be an annotated content model in disjunction-free normal form. When parsing g with e, each $g_i = \langle \tau_i, \gamma_i \rangle$, such that $\gamma_i \in cnt(g)$, is associated to one subexpression of e which corresponds to a group type; then, a subexpression of the form $exp?$ or $exp*$ is chosen, and a group sequence $g'_1, \ldots, g'_k$ valid for exp_i is found.

As an example, consider a group $g = \langle \tau, [\langle a, \gamma_1 \rangle, \langle b, \gamma_2 \rangle, \langle c, \gamma_3 \rangle] \rangle$ and an annotated content model in disjunction-free normal form $e = (a : x1, b : x2, (b : x3 \mid d : x4)*, c : x5)$ which matches g. Parsing g with e will result in assigning $x1$ to $\langle a, \gamma_1 \rangle$, $x2$ to $\langle b, \gamma_2 \rangle$, and $x5$ to $\langle c, \gamma_3 \rangle$. g can be expanded by selecting a new group of type b or d, and computing the groups $g' = \langle \tau, [\langle a, \gamma_1 \rangle, \langle b, \gamma_2 \rangle, \langle b, \gamma'_4 \rangle, \langle c, \gamma_3 \rangle] \rangle$ or $g'' = \langle \tau, [\langle a, \gamma_1 \rangle, \langle b, \gamma_2 \rangle, \langle d, \gamma''_4 \rangle \langle c, \gamma_3 \rangle] \rangle$, respectively. Besides groups g' and g'', two annotated content models in disjunction-free normal form that match g' and g'', respectively, can be derived and further used to find new group expansions. For example, with respect to group g', the annotated content model $e' = (a : x1, b : x2, b : x3, (b : x3 | d : x4)*, c : x5)$, which is in disjunction-free normal form, can be derived.

Input:
A PDF document *doc*; A PDF wrapper $W = \langle \tau_0, G \rangle$;
A truth value threshold t.
Output:
A maximal token group g.
Method:
$dbGroups := extractElementaryTokenGroups(doc, W, t)$;
$G := buildStarFreeGroups(W, dbGroups, t)$;
$g := selectMaxTruthValueGroup(G)$;
do
/ computes a group g' from g by expanding g itself or one of its subgroups */*
$g' :=$ `expand`$(W, g, dbGroups, t)$;
if $(g' \neq null)$ **then**
$g := g'$;
while $(g' \neq null)$;
return g;

Function `expand`$(W, g, dbGroups, t) : g'$;
Method:
$S := \emptyset$; $g' := null$;
for each g'' in $descendantOrSelf(g)$ **do**
$Exp :=$ `findExpansion`$(W, g'', dbGroups, t)$;
for each $gg \in Exp$ **do**
$gg' := replace(g, g'', gg)$;
if gg' is valid for $G(\tau_0)$ and $\mu_W(gg') \geq t$ and gg' is well-formed **then**
$S := S \cup \{gg'\}$;
if $(S \neq \emptyset)$ **then**
$g' := selectMaxTruthValueGroup(S)$;
return g';

Function `findExpansion`$(W, g, dbGroups, t) : Exp$;
Method:
let $g = \langle \tau, \gamma, e \rangle$, with $e = exp_1, \ldots, exp_n$;
$Exp := \emptyset$;
for each exp_i of the form $exp?$ or $exp*$ **do**
$S := instantiate(exp_i)$;
for each $ex \in S$ **do**
$SG := selectGroups(ex^\star, dbGroups)$;
for each $sg \in SG$ **do**
let $ex' = exp_1, \ldots, exp_{i-1}, ex, exp_{i+1}, \ldots, exp_n$;
if $\langle \tau, compose(ex', \gamma, sg) \rangle$ is valid for $G(\tau)$ and
$\mu_W(\langle \tau, compose(ex', \gamma, sg) \rangle \geq t$ and
$\langle \tau, compose(ex', \gamma, sg) \rangle$ is well-formed **then**
$Exp := Exp \cup \langle \tau, compose(ex', \gamma, sg), ex' \rangle$;
return Exp;

Fig. 2. The TokenGroupExtractor algorithm

Let g be a group to be expanded, and $e = exp_1, \ldots, exp_n$ be its matched annotated content model in disjunction-free normal form. The expansion of g can be computed by *instantiating* each expression $exp_i \in e$, that is by computing a set of annotated content models (in disjunction-free normal form) derived from exp_i and such that they do not parse the empty string. We define a function *instantiate* as follows:

- $instantiate(exp*) = instantiate(exp), exp*$;
- $instantiate(exp?) = instantiate(exp)$;
- $instantiate(exp_1|exp_2) = instantiate(exp_1) \cup instantiate(exp_2)$;
- $instantiate(exp_1, exp_2) = instantiate(exp_1), instantiate(exp_2)$;
- $instantiate(\tau : x) = \tau : x$.

The concatenation of two sets of annotated content models E_1, E_2 is the set of annotated content models $\{e_1, e_2 \mid e_1 \in E_1 \wedge e_2 \in E_2\}$. We say that an expression of the form $exp?$ or $exp*$ is an *expandable* expression.

Let $g = \langle \tau, \gamma \rangle$ be a group and $e = exp_1, \ldots, exp_n$ be the annotated content model in disjunction-free normal form that matches g. Let exp_i be an expandable expression, ex_i be an expression in $instantiate(exp_i)$, γ be $[g_1, \ldots, g_n]$, and $g'_1, \ldots, g'_k$ be a sequence of groups matching ex_i. The sequence $[g_1, \ldots, g_j, g'_1, \ldots, g'_k, g_{j+1}, \ldots, g_n]$ matching $ex = exp_1, \ldots, exp_{i-1}, ex_i, exp_{i+1}, \ldots, exp_n$, is denoted as $compose(ex, \gamma, [g'_1, \ldots, g'_k])$.

We are now able to gain an insight into the TokenGroupExtractor algorithm of Fig. 2. We assume that a database *dbGroups* is used to store extracted *annotated groups*. An annotated group is a triplet of the form $g = \langle \tau, \gamma, e \rangle$, where $\langle \tau, \gamma \rangle$ is a group and e the annotated content model in disjunction-free normal form that matches $\langle \tau, \gamma \rangle$.

Once elementary token groups have been extracted and stored into *dbGroups*, function *buildStarFreeGroups* first normalizes the input wrapper by producing, for each $gtd \in G$, group type definitions in disjunction-free normal form; then, following the order derived from the dependence relation between group types, for each normalized group type definition $\langle \tau, e, c \rangle$ it computes $\star$-free groups by invoking $selectGroups(e^\star, dbGroups)$, selects those that are well-formed and satisfy the desired truth value threshold, annotates them with the content model e, and finally adds the computed groups into *dbGroups*.

Function `expand` tries to expand a group g and returns an expanded group if possible, null otherwise. Among all possible ways of expanding g, function `expand` chooses the one exhibiting the highest truth value. Given three groups g, g', and gg, function *replace* returns a new group obtained by replacing g' with gg inside g if g' is a subgroup of g, or returns gg otherwise. Moreover, function $descendantOrSelf$, given a group g, yields the set of all the subgroups of g and g itself. Finally, function $selectMaxTruthValueGroup$, given a set of groups S, returns the group with the maximum truth value in S.

Function `findExpansion` computes a set of all the possible expansions of an annotated group with respect to the wrapper, the group database, and the truth value threshold. Function *selectGroups*, given a $\star$-free annotated content model $e = \tau_1 : x_1, \ldots, \tau_n : x_n$ (in disjunction-free normal form) and *dbGroups*, returns

the set of group sequences of the form $\{g_1, \ldots, g_n \mid \forall i\ g_i \in \pi_{\tau_i}(dbGroups)\}$, where $\pi_{\tau_i}(dbGroups)$ is the set $\{g \mid g \in dbGroups \wedge type(g) = \tau_i\}$.

Theorem 3. *Let W be a PDF wrapper, doc be a PDF document, and t be a truth value threshold. If $\mathcal{M}_t(W, doc)$ is not empty, the* TokenGroupExtractor *algorithm computes a maximal group g in $\mathcal{M}_t(W, doc)$ in polynomial time with respect to the number of tokens in doc.*

5 A Case Study: Wrapping Balance Sheets

To assess the effectiveness of the PDF wrapping framework, we considered a collection of balance sheets made publicly available from Italian companies; it is highly heterogeneous due to the variety of formatting styles used to report the balance assets and liabilities. For the sake of brevity, we describe a significant example of information extraction from a page of the test balance sheet shown in Fig. 3.

Suppose we would like to extract items each containing a balance voice (i.e. the item label) and two currency values referring to different fiscal years. Table 1 summarizes details about the specification of a wrapper suitable to the example balance sheet.

The maximal group to be extracted is of type `item_collection`. This group is composed of token groups `item`, which in turn consist of triplets of type (`balance_voice`, `amount`, `amount`). Each group `item` is constrained by a conjunction of two cardinal direction predicates. Once a group `item` has been built,

STATO PATRIMONIALE - ATTIVO	31/12/2003	31/12/2002
B) IMMOBILIZZAZIONI		
I) IMMOBILIZZAZIONI IMMATERIALI		
1) Costi di impianto e di ampliamento	10.739	73.792
5) Avviamento	433.824	495.799
I TOTALE IMMOBILIZZAZIONI IMMATERIALI	444.563	569.591
II) IMMOBILIZZAZIONI MATERIALI		
2) Impianti e macchinari		
a) impianti e macchinari	399.839	336.282
b) f.a.impianti e macchinari	169.253-	105.762-
2 TOTALE Impianti e macchinari	230.586	230.520
3) Attrezzature industriali e commerciali		
a) attrezzature industriali e commerciali	63.045	61.845
b) f.a.attrezzature industriali e commerciali	47.446-	29.561-
3 TOTALE Attrezzature industriali e commerciali	15.599	32.284
4) Altri beni		
a) altri beni	19.693	18.703
b) f.a.altri beni	11.094-	6.621-
4 TOTALE Altri beni	8.599	12.082
II TOTALE IMMOBILIZZAZIONI MATERIALI	254.784	274.886

Bilancio di esercizio al 31/12/2003 Pagina 1

Fig. 3. Sample page of the test Italian company's balance sheet

Table 1. A wrapper for the balance sheet of Fig. 3

$W = \langle \tau_0, G \rangle$	$\tau_0 =$ `item_collection` $G = \langle gtd_1, gtd_2, gtd_3, gtd_4 \rangle$
$gtd_1 = \langle \tau_1, e_1, c_1 \rangle$	$\tau_1 = \tau_0$ $e_1 = (\texttt{item} : IT)*$ $c_1 = True()$
$gtd_2 = \langle \tau_2, e_2, c_2 \rangle$	$\tau_2 = item$ $e_2 = \texttt{balance\_voice} : BV, \texttt{amount} : N_1, \texttt{amount} : N_2$ $c_2 = west(BV, N_1) \wedge west(N_1, N_2)$
$gtd_3 = \langle \tau_3, e_3, c_3 \rangle$	$\tau_3 =$ `balance_voice` $e_3 = \sharp TOKEN : X_1$ $c_3 = concept(X_1, balance\_voice\_object)$
$gtd_4 = \langle \tau_4, e_4, c_4 \rangle$	$\tau_4 =$ `amount` $e_4 = \sharp TOKEN : X_2$ $c_4 = isNumber(X_2)$

```
<?xml version="1.0" encoding="UTF-8"?>
<item_collection>
   <item>
      <balance_voice>
         <value>1) Costi di impianto
                  e di ampliamento</value>
         <page>1</page>
         <inf_x>45.7799</inf_x>
         <inf_y>372.8574</inf_y>
         <sup_x>210.92456</sup_x>
         <sup_y>383.8794</sup_y>
      </balance_voice>
      <amount>
         <value>10.739</value>
         <page>1</page>
         <inf_x>423.2689</inf_x>
         <inf_y>374.8614</inf_y>
         <sup_x>476.07428</sup_x>
         <sup_y>383.8794</sup_y>
      </amount>
      <amount>
         <value>73.792</value>
         <page>1</page>
         <inf_x>505.3864</inf_x>
         <inf_y>374.8614</inf_y>
         <sup_x>558.19183</sup_x>
         <sup_y>383.8794</sup_y>
      </amount>
   </item>
   <item>
      <balance_voice>
         <value>5) Avviamento</value>
         <page>1</page>
         <inf_x>45.7799</inf_x>
         <inf_y>395.8974</inf_y>
         <sup_x>118.79464</sup_x>
         <sup_y>406.9194</sup_y>
      </balance_voice>
      <amount>
         <value>433.824</value>
         <page>1</page>
         <inf_x>420.4875</inf_x>
         <inf_y>397.9014</inf_y>
         <sup_x>475.847</sup_x>
         <sup_y>406.9194</sup_y>
      </amount>
      <amount>
         <value>495.799</value>
         <page>1</page>
         <inf_x>502.6391</inf_x>
         <inf_y>397.9014</inf_y>
         <sup_x>557.9986</sup_x>
         <sup_y>406.9194</sup_y>
      </amount>
   </item>
   . . .
   . . .
</item_collection>
```

Fig. 4. XML document extracted from the balance sheet of Fig. 3

it is added to the content of the group of type `item_collection`, without being subject to a real constraint (i.e. constraint c_1 always holds).

Another noteworthy remark concerns c_3. This constraint checks whether a token associated to type `balance_voice` really represents a balance voice. To accomplish this, a predicate *concept* evaluates the membership of a given token as an instance of class *balance_voice_object*, according to some functions which

compute at what degree a string (token value) conceptually matches a class property (e.g. a domain-specific cue phrase). Figure 4 shows an XML fragment representing the maximal token group of type `item_collection` extracted by evaluating the wrapper of Table 1, with a truth value threshold set to 0.8.

6 Conclusion

We have presented a novel PDF wrapping framework based on a bottom-up approach, in which the extraction task consists in grouping together document tokens. A wrapper is defined by specifying the content for each type of token group. Annotated content models and fuzzy constraints as the basic elements of group type definitions. Fuzzy constraints are used to impose spatial and logical conditions on the content of each group. We have defined a declarative semantics of PDF wrappers and provided a polynomial time algorithm for extracting maximal groups. We have given evidence that fuzzy constraints are well-suited to capture subjective factors that brand the authorship in logically structuring information into a PDF document.

A system prototype is in advanced phase of development[2] and is currently being applied for extracting information from balance sheets.

References

1. Ashish, N., Knoblock, C.A.: Wrapper Generation for Semistructured Internet Sources. ACM SIGMOD Record **26**(4) (1997) 8–15
2. Baumgartner, R., Flesca, S., Gottlob, G.: Visual Web Information Extraction with Lixto. In: Proc. VLDB '01 Conf. (2001) 119–128
3. Crescenzi, V., Mecca, G., Merialdo, P.: RoadRunner: Towards automatic data extraction from large Web sites. In: Proc. VLDB '01 Conf. (2001) 109–118
4. Freitag, D.: Machine Learning for Information Extraction in Informal Domains. Machine Learning **39**(2–3) (2000) 233–272
5. Muslea, I., Minton, S., Knoblock, C.: Hierarchical Wrapper Induction for Semistructured Information Sources. Autonomous Agents and Multi-Agent Systems **4**(1/2) (2001) 93–114
6. Soderland, S.: Learning Information Extraction Rules for Semistructured and Free Text. Machine Learning **34**(1–3) (1999) 233–272
7. Laender, A., Ribeiro-Neto, B., da Silva, A., Teixeira, J.: A Brief Survey of Web Data Extraction Tools. ACM SIGMOD Record **31**(2) (2002) 84–93
8. Adobe Systems Incorporated: PDF Reference, 5th edition: Adobe Portable Document Format version 1.6. Available at `http://partners.adobe.com/public/developer/pdf` (2004)
9. Zadeh, L.: Fuzzy Sets. Information and Control **8** (1965) 338–353
10. Wygralak, M.: Fuzzy Cardinals based on the Generalized Equality of Fuzzy Subsets. Fuzzy Sets & Systems **18** (1986) 143–158
11. Bruggemann-Klein, A., Wood, D.: One-Unambiguous Regular Languages. Information and Computation **142**(2) (1998) 182–206

[2] `http://www.deis.unical.it/tagarelli/pdf-wrapping`