UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

THICK 2D RELATIONS FOR DOCUMENT UNDERSTANDING

Marco Aiello and Arnold M.W. Smeulders

2002

Technical Report # DIT-02-0063

Thick 2D Relations for Document Understanding

Marco Aiello *,1 Arnold M.W. Smeulders

Intelligent Sensory Information Systems, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands

Abstract

We use a propositional language of qualitative rectangle relations to detect the reading order from document images. To this end, we define the notion of a document encoding rule and we analyze possible formalisms to express document encoding rules such as LaTeX and SGML. Document encoding rules expressed in the propositional language of rectangles are used to build a reading order detector for document images. In order to achieve robustness and avoid brittleness when applying the system to real life document images, the notion of a thick boundary interpretation for a qualitative relation is introduced. The framework is tested on a collection of heterogeneous document images showing recall rates up to 89%.

Key words: document image analysis, document understanding, spatial reasoning, bidimensional Allen relations, constraint satisfaction: applications

1 Introduction

When Dave placed his own drawing in front of the 'eye' of HAL—in 2001: A Space Odyssey—HAL showed to have correctly comprehended and interpreted the sketch. "That's Dr. Hunter, isn't it?" (Rosenfeld, 1997). But what would have happened if Dave used the first page of a newspaper in front of the eye and started discussing its contents? Considering HAL a system capable of AI, we expect HAL to recognize the document as a newspaper, to understand how to extract information and to understand its contents. Finally, we expect Dave

* Corresponding author.

1 Presently at Department of Information and Telecommunication Technologies, University of Trento, Via Sommarive, 14, 38050 Trento, Italy

and HAL to begin a conversation on the contents of the document. In short, HAL has to be able to perform *document image analysis*.

Document image analysis is the set of techniques to recover syntactic and semantic information from images of documents, prominently scanned versions of paper documents. An excellent survey of document image analysis is provided in (Nagy, 2000) where, by going through 99 articles that appeared in the IEEE's Transactions on Pattern Analysis and Machine Intelligence (PAMI), Nagy reconstructs the history and state of the art of document image analysis. Research in document images analysis is useful and studied in connection with document reproduction, digital libraries, information retrieval, office automation, and text-to-speech.

One may have different goals when performing document image analysis. For instance, one may be in interested in the reconstruction of the reading order of a document from its image. One way to achieve this is by performing the following intermediate steps. First, one identifies the basic components of the document, the so-called *document objects*. Second, one identifies the logical function of the document objects within the document (e.g., title, page number, caption). This is called *logical labeling*. Last, one infers the order in which the user is to read the document objects. This phase is called the *reading order detection*. In the process, one moves from basic geometric information of the document composition, the *layout structure*, to semantic information, the *logical structure*. document objects and their spatial arrangement are prototypical examples of elements of the layout structure, while the reading order is an instance of the logical structure.

In Figure 1, we illustrate possible flows of information in document image analysis. The first row represents the flow from the document image to its reading order. The following row represents the flow from the image to the identification of the document class. Discovering to which scientific publication belongs a given document image is an example of document classification. One should interpret the arrows in the figure as possible choices. It is perfectly normal to move from one row to another, or to stop the analysis at the layout structure level. For example, systems for mail delivery do not need to perform any document classification, or reading order detection.

The first document image analysis systems were built to process documents of a specific class, e.g., forms for telegraph input. One of the recent trends is to build systems as flexible as possible, capable of treating the widest variety of documents. This has led to categorize the knowledge used in a document image analysis system into: class specific and general knowledge (e.g. Cesarini et al., 1999). In addition, such knowledge can be explicitly available or implicitly hard-coded in the system.

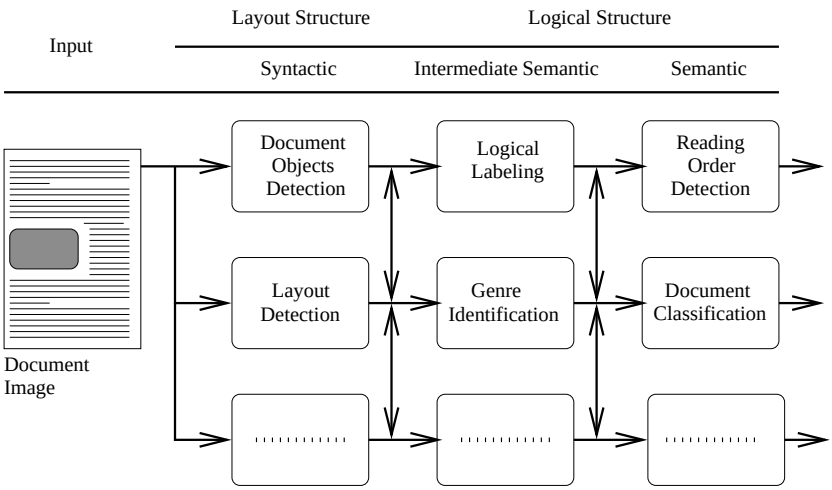


Fig. 1. Various tasks in document image analysis and understanding. Left to right, from input data towards semantic content.

Lee et al. (2000) present a system to analyze technical journals of one kind (PAMI) based on explicit knowledge of the specific journal. The goal is that of region segmentation and identification (logical labeling). The knowledge is formalized in "IF-THEN" rules applied directly to part of the document image and "IF-THEN" meta rules. Though the idea of encoding the class specific knowledge of a document is promising, it is not clear whether the proposed approach is scalable and flexible. Given the specific form of the IF-THEN rules, the impression is that the system is not suited for the analysis of documents different from PAMI. Experimental results show good performance in the task of logical labeling, especially in the detection of formulas and drawings embedded in the main text.

There are a number of problems related to the rule based approaches found in the literature. The most prominent is the high specificity of the rules. The specificity makes it hard or impossible to extend such systems to documents of a class different from the one for which the system was originally designed. Another problem is the lack of proof of correctness or termination. Recent rule-based approaches for layout and logical structure detection are presented in (Klink and Kieninger, 2001; Lee et al., 2000; Niyogi and Srihari, 1996) while an older one is (Tsujimoto and Asada, 1992).

Given the difficulty in designing appropriate rules for the analysis of documents, approaches based on learning are interesting. The document classification components of the WISDOM++ system (Altamura et al., 2001) are based on first-order learning algorithms (Esposito et al., 2000). Another advantage of such systems is their flexibility compared to the non-learning based systems. By training the system on a different class of documents with similar layout, it should be possible to reuse the same architecture. On the negative side, the rules learned are not intuitive. More often than not, these rules are

impossible to modularize for further use on different document classes.

An important aspect of a document image analysis system working at the logical structure level is the representation of the information extracted from the document. The key here is a modularity and standardization of the representation. Markup languages are a good example of representation means with such qualities. The system presented in (Worring and Smeulders, 1999) uses HTML as its final output form, while Altamura et al. (2001) use XML. More abstract representations are labeled and weighted graphs. These have been used in various systems such as, for instance, the ones presented in (Li and Ng, 1999; Cesarini et al., 1998; Walischewski, 1997).

As we are investigating practical applications of spatial reasoning formalisms, it is relevant to review approaches using these kind of formalisms. In particular, we consider bidimensional extensions of Allen's interval relations, that is, rectangular relations. To the best of our knowledge, bidimensional Allen relations have been used in document image analysis in three cases (Klink et al., 2000; Singh et al., 1999; Walischewski, 1997). In all these approaches, bidimensional Allen relations are used as geometric features descriptors, at times as labels for graphs and at other times as layout relations among document objects. Thus, the use of Allen relations is relegated to feature comparison and it is not used for performing any other kind of reasoning.

We present a methodology based on inference with bidimensional qualitative spatial relations for logical structure detection of document images. In particular, the methodology addresses the issue of detecting the reading order in documents from an heterogeneous collection without using any document specific knowledge.

The methodology is implemented in a prototype system named `SpaRe` (`Spa`tial `Re`asoning component) part of a larger architecture for logical structure detection in a broad class of documents. In the next section, we give an overview of the architecture. In Section 3, we describe the methodology based on the concept of document encoding rule and of thick boundary interpretation of bidimensional Allen relations. Section 4 is dedicated to the experimental results and their discussion. Directions for future work and a discussion of the methodology are presented in Section 5.

2 A logical structure detection architecture

In (Aiello et al., 2002), the authors present a logical structure detection architecture. Departing from a pre-processed document image the goal of such an architecture is that of logically labeling the document objects and subsequently

identify the reading order. The system uses general document knowledge only, hence, it is applicable to documents of different classes.

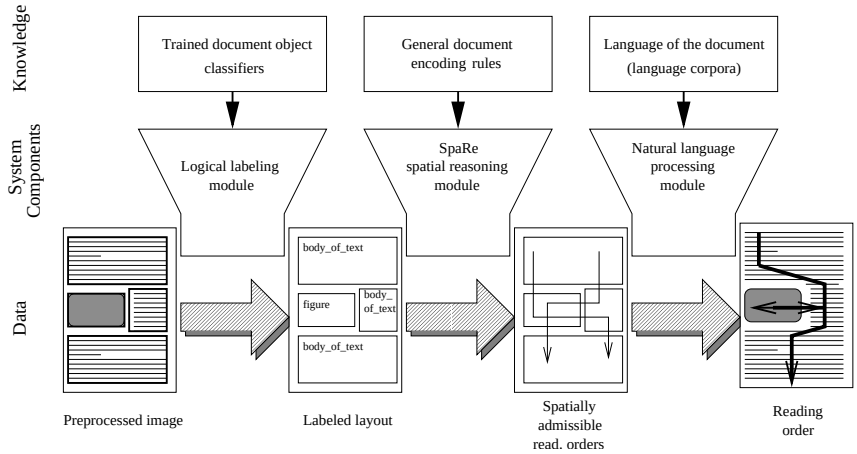


Fig. 2. The flow of knowledge and data in the logical structure detection architecture presented in (Aiello et al., 2002).

Referring to Figure 2, one has a glimpse of the architecture presented in (Aiello et al., 2002). The input is a pre-processed document image in which the document objects have been segmented, local textual content recognized and font information identified. The original document can be of any class as long as it is acceptable that document objects are represented by rectangles. Overlapping document objects are accepted by the system.

There are three modules: a logical labeler, a spatial reasoning reading order detector, and a natural language processing 'disambiguator'. Logical labeling on the pre-processed image is achieved via pre-trained classifiers.

The spatial reasoning module starts from the logically labeled layout of the document and, using general document encoding rules, it outputs a number of reading orders. The module is the subject of the remainder of the paper.

The natural language processing module starts from the spatially admissible reading orders and the textual content of each one of the textual document objects. It uses this information to prune the set of spatially admissible reading orders of those which are linguistically not acceptable. This is performed by applying a combination of statistical methods and shallow parsing techniques. The statistical tools are trained on a large corpora of text. The training corpora is based on (Hersh et al., 1994) and (Baayen et al., 1995) which are independent from the document classes analyzed. Details of this module are presented in (Todoran et al., 2001b).

The output of the system is a reading order for the input document image. To be more precise, the output is a list of reading orders for the document ranked in order of linguistic plausibility (a probability is assigned to each

reading order). Experimental results on two different collections of documents for each module and for the whole system have been presented in (Aiello et al., 2000; Todoran et al., 2001b; Aiello et al., 2002).

3 Methodology

We focus on the spatial reasoning module of the architecture presented in the previous section. Figure 3 is a zoom-in of the spatial reasoning component in Figure 2 highlighting details. First, the generic document knowledge in the form of document encoding rules may have different origins. Second, the spatial reasoning module `SpaRe`, is actually composed of two sub-modules. The first one, which performs inference on the spatial relations of the layout and on the document encoding rules, is based on constraint satisfaction techniques. The second one is a module to sort graphs, that is, directed transitive cyclic ones. In the following sections, we analyze each of these items.

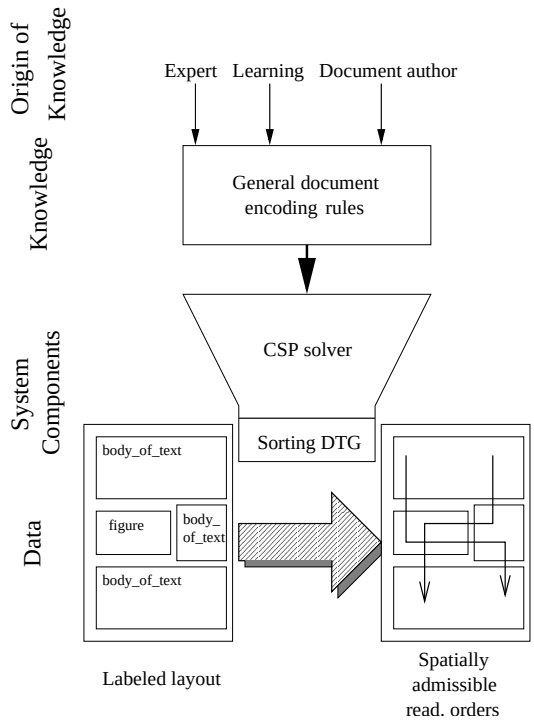


Fig. 3. The flow of knowledge and data in the spatial reasoning module `SpaRe`. The document encoding rules originate from an expert, or from previous learning or are given directly by the document author. The module itself is composed of a constraint satisfaction problem solver and a handler for directed transitive cyclic graphs.

3.1 Document encoding rules

A *document encoding rule* is a principle followed by the author of a document to convey an intent of the author by layout details. document encoding rules can be one of two types: general or class specific. Document encoding rules can be expressed in a informal or in a formal manner. Informal rules are proposed in natural language or by sketch. Examples are found in books such as (Reynold, 1979). Examples of generic and specific, and formal and informal rules are presented in Figure 4.

	general	class specific
informal	"the caption neighbors its figure"	"the caption starts with the word "Fig." with font size 12pt, green text, and it is centered"
formal	$\forall f \in$figure $\exists c \in$caption: neighbors(f, c)	$\forall c \in$caption$\rightarrow$ text_starts$(c,$ "Fig. "$)\wedge$ font_color$(c,$ "Green "$)\wedge$ point$(c, 12)\wedge$centered(c)

Fig. 4. Examples of generic and specific, and formal and informal rules. The formal rules are expressed in a first-order like language for documents whose semantics should be self-evident.

Let us consider a number of formal ways to express document encoding rules.

LaTeX is a compiled markup language. Typically, there is a number of source files with the main marked-up text (the `.tex` files), a number of style definition files (`.sty`, `.cls`) and a compiler. The document encoding rules can reside as macros in the `.tex` file, but the most common solution is that document encoding rules reside inside the style files. Consider the figure environment in the class file for generating transactions for the ACM. [2]

[2] M. Aiello. (2001). `http://www.acm.org/pubs/submissions/latex_style/acmtrans2m.cls`. The class file currently in use at ACM, an extension of the `acmtrans2e.cls` version.

```
\newcounter{figure}
\def\thefigure{\@arabic\c@figure}
\def\fps@figure{tbp}
\def\ftype@figure{1}
\def\ext@figure{lof}
\def\fnum@figure{Fig.\ \thefigure}
\def\figure{\let\normalsize\footnotesize \normalsize
      \@float{figure}}\let\endfigure\end@float
\@namedef{figure*}{\@dblfloat{figure}}
\@namedef{endfigure*}{\end@dblfloat}
```

The above definition, among other things,[3] defines the figure as belonging to a float environment (Goossens et al., 1994) whose default major features are: a float occupies the top of a page; a float does not have to appear where it is declared in the source; a float should not occupy more than 70% of the page otherwise it is moved after the first `\clearpage` instruction; if a caption is present it cannot be split across pages. The ACM transactions style file provides further class specific definitions for displaying the caption which overwrite the corresponding LaTeX definitions.

```
\long\def\@makecaption#1#2{\vskip 1pc
        \setbox\@tempboxa\hbox{#1.\hskip 1em\relax #2}
        \ifdim \wd\@tempboxa >\hsize #1. #2\par \else
        \hbox to\hsize{\hfil\box\@tempboxa\hfil}
        \fi}
\def\nocaption{\refstepcounter\@captype \par
      \vskip 1pc \hbox to\hsize{\hfil \footnotesize
      Figure \thefigure\hfil}}
```

The second example of a document rule in LaTeX places the word "Figure" the figure counter immediately below the picture, placing a vertical space of `1pc` units (i.e., 12pt). The size of such text is set to the value of `\footnotesize`.

More abstractly, the document encoding rule for a figure says that a figure is left to float in the main text, its preferred position is on top of a page and the caption is placed immediately below. The figure and caption always appear in the above/below spatial relation on one page.

WYSIWYG are computer systems in which the input of the user corresponds almost exactly to the final layout of the document (WYSIWYG stands for 'what you see is what you get'). A prototypical example is Microsoft's Word. In these sort of systems, it is hard to distinguish between the syntactic and the semantic portion of document encoding rules, as they are hidden in the implementation. The only control that the user has over the formal document encoding rules is through the functionalities provided by an interface allowing the user to change rule parameters.

In a common way of employing the WYSIWYG-style there are few strict

[3] See (Knuth, 1984) for details over the syntax and semantics of TeX.

document encoding rules, while the user still enforces elements of style. In a typical way of doing, captions may be put underneath a figure and also typically be indented (apart from the one place where the user forgot to implement that).

However, when observing the text one could learn document encoding rules, for example, that captions are always below the figure and immediately following it provided there is one. In that case, one would require rules which can express topological relationships with some form of tolerance as the user will implement notions like alignment and marking with a limited precision. In addition, one would require rules which express topographic relationships as they can be implemented in the freedom to move around on the 2D-screen where the WYSIWYG-program runs, implying that the caption is always close to the figure. Finally, to address the inconsistencies of ad hoc rule implementation and the lack of discipline to enforce them would require rules with a less than strict character.

SGML languages are a family of interpreted markup languages, whose best known members are HTML and XML. The eXtensible Markup Language, XML for short, achieves a clear separation between content (the `.{XML}` file), syntactic document encoding rules (`.css`, `.xsl`, `.dtd`) and semantics of the document encoding rules (the browser's interpretation of the document encoding rules). For instance, the document encoding rule for a caption like <CAPTION> A figure </CAPTION> could be the following:

- *(syntax):* inside a `.css` file
  CAPTION
  {dispaly: block; font-size: 12pt; color: #000000; text-align: center}
- *(semantics):* the browser will display the text "A figure" in one block of text, in black color, using the default font, using the font size 12pt, and center it.

To the same degree SGML as WYSIWYG offers the possibility to move around the images of the document objects and hence implement document encoding rules by habit rather than by a priori rules. As the user has no visual feedback, the factual encoding rules are more informal than in the WYSIWYG paradigm. Hence, here are needed topological and topographical rule sets to describe the power of SGML but even more forgiving than in the WYSIWYG style.

Abstract formal languages can also serve as document encoding languages, for instance, first-order logic. The syntax and semantics are the usual ones for first-order logic, taking special care in giving adequate semantics to spatial relations and predicates.

A final example of a general document encoding rule stated informally in natural language is the following:

"in the Western culture, documents are usually read top-bottom and left-right." (1)

One can immediately spot a problem of stating rules in natural language, that is, ambiguity. In fact, we do not know if one should interpret the "and" as commutative or not. Should one first go top-bottom and *then* left-right? Or, should one apply any of the two interchangeably? It is not possible to say from the rule merely stated in natural language.

In the next section, we define an abstract propositional formal language to express qualitative spatial relations among document objects to formally express document encoding rules.

3.2 Relations adequate for documents

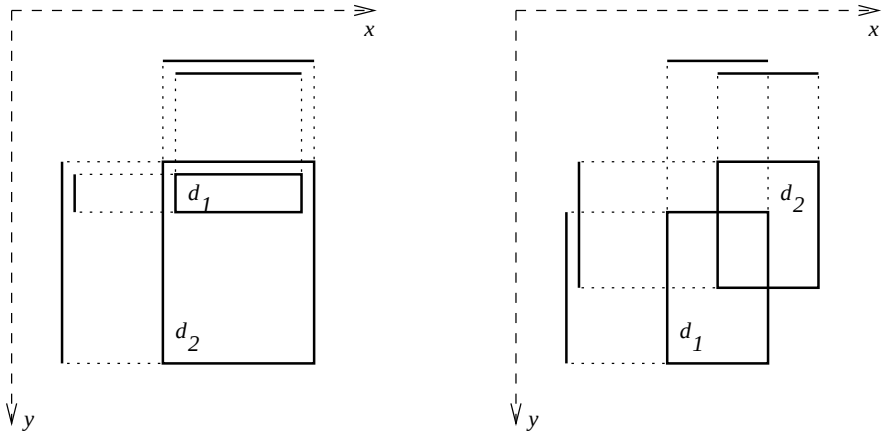
Considering relations adequate for documents and their components, requires a preliminary formalization step. This consists of regarding a document as a formal model. At this level of abstraction a document is a tuple $\langle D, R, l\rangle$ of document objects D, a binary relation R, and a labeling function l. Each document object $d \in D$ consists of the coordinates of its bounding box (defined as the smallest rectangle containing all elements of that object)

$$D = \{d \mid d = \langle id, x_1, y_1, x_2, y_2\rangle\}$$

where id is an identifier of the document object and (x_1, y_1) (x_2, y_2) represent the upper-left corner and the lower-right corner of the bounding box of the document object. In addition, we consider the logical labeling information. Given a set of labels L, logical labeling is a function l, typically injective, from document objects to labels:

$$l: \ D \rightarrow L$$

In the following, we consider an instance of such a model where the set of relations R is the set of bidimensional Allen relations and where the set of labels L is {title, body_of_text, figure, caption, footer, header, page_number, graphics}. We shall refer to this model as a *spatial [bidimensional Allen] model.* Bidimensional Allen relations consist of 13×13 relations: the product of Allen's 13 interval relations (Allen, 1983; van Benthem, 1983) on two orthogonal axes. (Consider an inverted coordinate system for each document with origin (0,0) in the left-upper corner. The x axis spans horizontally increasing to the right, while the y axis spans vertically towards the bottom.) Each relation $r \in A$ is a tuple of Allen interval relations of the form: $precedes$, $meets$, $overlaps$, $starts$, $during$, $finishes$, $equals$, and $precedes\_i$, $meets\_i$, $overlaps\_i$, $starts\_i$, $during\_i$, $finishes\_i$. We shall refer to the set of Allen bidimensional relations simply as A and to the propositional language over bidimensional Allen relations as $\mathcal{L}$ the remainder of the paper. Since Allen relations are jointly exhaustive and pairwise disjoint, so is A. This implies that given any two document objects there is one and only one A relation holding among them.



(a) (b)

Fig. 5. (a) The document object d_1 is `Part` of d_2, as the projection of d_1 on both axes is *during* the projection of d_2; (b) The document object d_2 `Overlap`s with d_2, as the projection on x of d_1 *overlaps* that of d_2 and on y it *overlaps_i* that of d_2.

Document objects are represented by their bounding boxes and the relative position of these objects plays a key role in the interpretation of the meaning of the document. For example, if a document object is above another one it is likely that it should be read before. If a document object is contained in another one, it is likely that the contained one is a 'part' of the containing one, for instance the title of a remark inside a frame. document objects can be also overlapping. This last feature is more common when the document has different colors and colored text runs over pictures, logos and drawings.

All relations of the examples above are expressible in terms of $\mathcal{L}$. For instance, 'being part of' is

$$\begin{aligned}\texttt{Part}(d_1, d_2) \text{ iff } &(during\_x(d_1, d_2) \vee starts\_x(d_1, d_2) \vee finishes\_x(d_1, d_2)) \wedge \\ &(during\_y(d_1, d_2) \vee starts\_y(d_1, d_2) \vee finishes\_y(d_1, d_2)) \quad (2)\end{aligned}$$

To analyze the expressive power of $\mathcal{L}$, we encode the basic RCC5 (Randell et al., 1992) relations:

- $\texttt{Part}^{-1}(d_1, d_2) = \texttt{Part}(d_2, d_1)$,
- $\texttt{Equal}(d_1, d_2) = equal\_x(d_1, d_2) \wedge equal\_y(d_1, d_2)$,
- $\texttt{Disconnected}(d_1, d_2) = precedes\_x(d_1, d_2) \vee precedes\_i\_x(d_1, d_2) \vee$ $precedes\_y(d_1, d_2) \vee precedes\_i\_y(d_1, d_2)$,
- $\texttt{Overlap}(d_1, d_2) = \neg\texttt{Part}(d_1, d_2) \wedge \neg\texttt{Part}^{-1}(d_1, d_2) \wedge$ $\neg\texttt{Equal}(d_1, d_2) \wedge \neg\texttt{Disconnected}(d_1, d_2) \wedge$ $\neg\texttt{ExternalConnection}(d_1, d_2)$.

Similarly, one can encode RCC8 in $\mathcal{L}$. Examples of document objects satisfying the part and the overlap relations are presented in Figure 5.

Restricting attention to RCC relations one looses a feature of $\mathcal{L}$ of great importance, namely, its ordering expressivity with respect to the axes. Take for instance the `Disconnected` relation. There are various ways in which two document objects can satisfy this relation. If either $precedes\_x(d_1, d_2) \wedge equal\_y\_x(d_1, d_2)$ or $precedes\_i\_x(d_1, d_2) \wedge equal\_y\_x(d_1, d_2)$ holds, then it is true that the RCC8 predicate $\texttt{Disconnected}(d_1, d_2)$ holds, but the two situations are most different. In the first case, d_1 is to the left of d_2, in the second case it is to the right. In other words, in the first case it is likely that d_1 is to be read before than d_2 in the document, while in the second case d_2 is to be read before d_1. This is one of the key features that we exploit in using $\mathcal{L}$ to define document encoding rules.

Consider again the example of the relation between a figure and its caption in the LaTeX ACM transactions class file. This spatial relation is $\mathcal{L}$ definable:

$$(during\_x(\text{figure}, \text{caption}) \vee equals\_x(\text{figure}, \text{caption})) \wedge precedes\_y(\text{figure}, \text{caption})$$

The spatial relation between the word "Figure" and the figure counter is also $\mathcal{L}$ definable:

$$meets\_x(\text{“Figure ”}, \text{figure\_counter}) \wedge$$
$$equals\_y(\text{“Figure ”}, \text{figure\_counter}) \vee during\_i\_y(\text{“Figure ”}, \text{figure\_counter})$$

Other features of the LaTeX definitions are not $\mathcal{L}$ definable: trivially, all font and textual features. But also size and distance features are not $\mathcal{L}$ definable, e.g., the fact that the white space between a figure and a caption is of a fixed amount (`1pc`).

3.2.1 Document encoding rules with $\mathcal{L}$

The language $\mathcal{L}$ is adequate to express mereotopological and ordering relations among rectangles. Here, we show how to use this power to express formal unambiguous document encoding rules.

Take the informal document encoding rule (1) expressed in natural language. Consider the layout of a document as presented in Figure 6.a, where the numbering of the document objects is provided counterclockwise. After having read the document object 2, to which one should the reader move? Only having the layout and not the content of the text there is not a unique choice. One would either move to the block of text 4 or to block 3. In the first case, one has followed the left-right rule, in the latter the top-bottom rule. No one would have proposed to move to block 1, this because it is in violation of the top-bottom rule.

The top-bottom, left-right document rules are expressible in the language $\mathcal{L}$

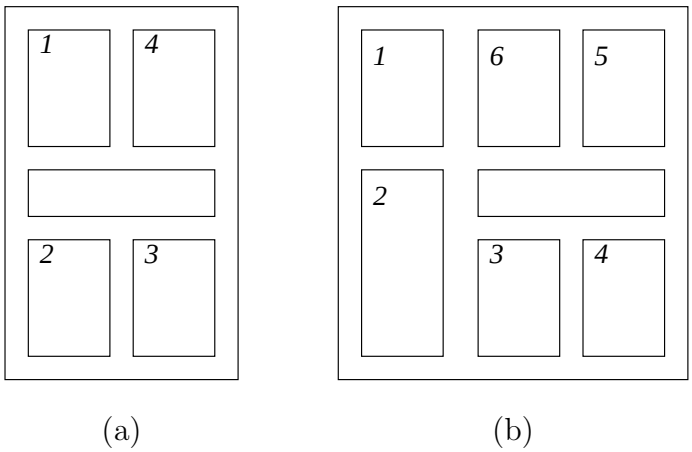


Fig. 6. Layouts of documents considering text objects only.

by:

$$\begin{aligned}\text{before\_in\_reading}(d_1, d_2) \text{ iff } & \mathit{precedes\_x}(d_1, d_2) \vee \mathit{meets\_x}(d_1, d_2) \vee \\ & \mathit{overlaps\_x}(d_1, d_2) \vee \mathit{precedes\_y}(d_1, d_2) \vee \\ & \mathit{meets\_y}(d_1, d_2) \vee \mathit{overlaps\_y}(d_1, d_2) \qquad (3)\end{aligned}$$

The equation reads "the document object d_1 is 'before in the reading order' of the document object d_2 if the a Boolean combination of basic $\mathcal{L}$ relations are satisfied." The rule (3) is the formal counterpart to (1). Though the generality of (3) is also its weakness. Too many document objects satisfy it, calling for the design of rules balancing between being more restrictive and being general. Consider the layout proposed in Figure 6.b. It is hard to judge if one would follow the reading $1, 2, 6, 3, 5, 4$ or the reading $1, 6, 5, 2, 3, 4$, but the reading $1, 6, 2, 3, 5, 4$ surely seems odd. Without knowing the content of the document, we are inclined to consistently apply a column-wise or row-wise rule. Therefore, a candidate for a general and yet more restrictive rule in comparison with (3) is a *column-wise* document rule. In this case, one *first* goes top-bottom, *then* left-right. A rule to encode this behavior is again expressible with $\mathcal{L}$. It has the following form:

before_in_reading$^{\text{col}}(d_1, d_2)$ iff

$$
\begin{aligned}
& precedes\_x(d_1, d_2) \vee meets\_x(d_1, d_2) \vee \\
& \\
& (overlaps\_x(d_1, d_2) \wedge \\
& (precedes\_y(d_1, d_2) \vee meets\_y(d_1, d_2) \vee overlaps\_y(d_1, d_2))) \vee \\
& \\
& ((precedes\_y(d_1, d_2) \vee meets\_y(d_1, d_2) \vee overlaps\_y(d_1, d_2)) \wedge \\
& (precedes\_x(d_1, d_2) \vee meets\_x(d_1, d_2) \vee overlaps\_x(d_1, d_2) \vee \\
& starts\_x(d_1, d_2) \vee finishes\_i\_x(d_1, d_2) \vee equals\_x(d_1, d_2) \vee \\
& during\_x(d_1, d_2) \vee during\_i\_x(d_1, d_2) \vee finishes\_x(d_1, d_2) \vee \\
& starts\_i\_x(d_1, d_2) \vee overlaps\_i\_x(d_1, d_2))) \qquad (4)
\end{aligned}
$$

An analogous *row-wise* rule is obtained by inverting the axes in (4).

3.2.2 Thick boundary interpretation

The direct application of systems based on Allen or similar relations results in brittle systems. This is because Allen relations rely on the precise identification of a boundary of the interval. This means that some relations never occur in practical situations. One goes directly from *precedes* to *overlaps* and from *overlaps* to *during* without ever identifying an instance of *meets*, *starts*, or *finishes*. To solve this drawback of Allen-like relations, we provide a less brittle interpretation of the relations.

Instead of considering two interval extremes to be equal when they have the same coordinates, we consider them equal if they are closer than a fixed threshold distance T. This can be seen as if the bounding boxes of the document objects have a *thick boundary*. We name the set of 13 Allen's relations thus interpreted *thick boundary rectangle relations*.

The thickness of the boundary is assumed identical for all objects in the document. It is fixed with respect to the page size. The optimal value is found through experimentation. There is a constraint on the T with respect to the size of the smallest document object: it should not exceed half the size of the shortest side of all bounding boxes. Referring to Figure 7, one sees how the A relations with their thick interpretation are more tolerant in the establishment of a relation between two intervals. For example, interval a meets interval b not only if $x_2^a = x_1^b$, but also if $x_1^b - T \leq x_2^a \leq x_1^b + T$. With the thick boundary interpretation, Allen's relation maintain the jointly exhaustive and pairwise disjoint property (cf. Aiello et al., 2002, for a proof).

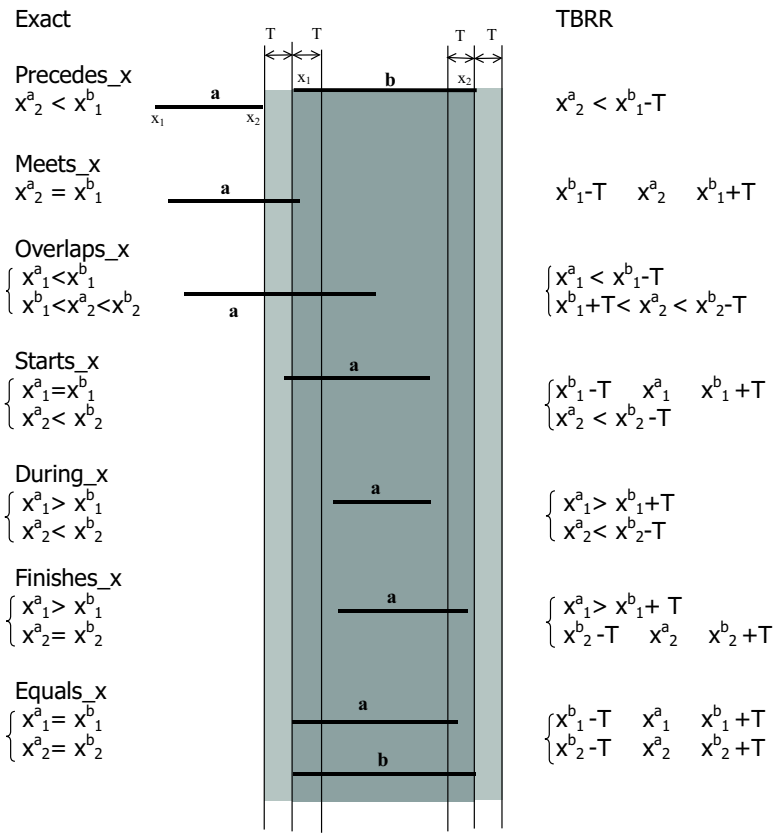


Fig. 7. The thick boundary interpretation of Allen's relations. The interval b is considered fixed and the threshold T is highlighted on its extreme points. The interval a varies in all 13 possible positions. On the left, the equation of the standard interpretation of Allen's relations. On the right, the thick boundary interpretation.

3.3 Inference

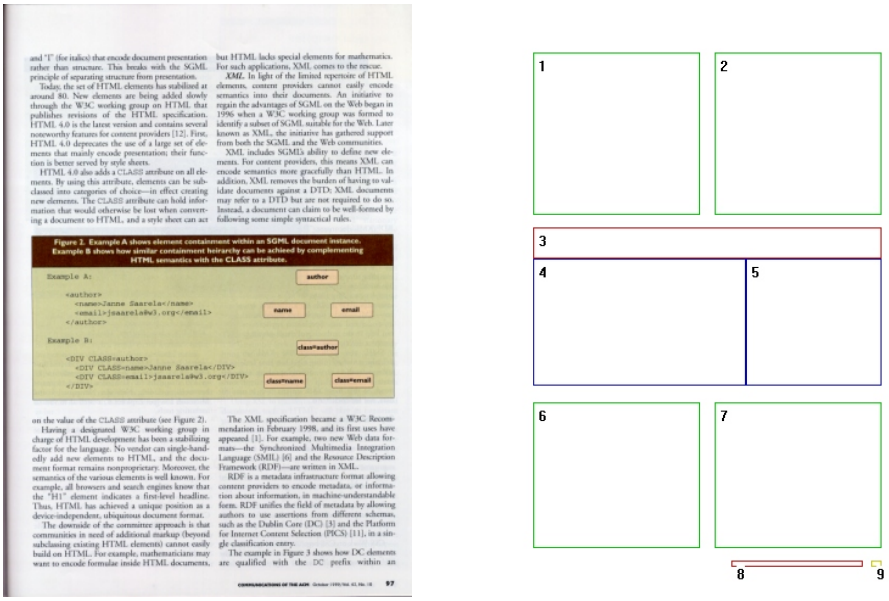
Equipped with a qualitative spatial language for document objects $\mathcal{L}$, with document encoding rules and the layout and logical labeling information, we are now in the position to perform inference in order to achieve 'understanding' of a document. Following is the definition of document understanding in this context.

First, we define the notion of an admissible transition between document objects. Given a pair of document objects d_1 and d_2, a document model $\langle D, R, l\rangle$ and a set of document encoding rules S, we say that (d_1, d_2) is an *admissible transition* with respect to R iff the bidimensional Allen relation $(d_1, d_2) \in R$ is consistent with S.

A *spatially admissible reading order* with respect to a document model $\langle D, R, l\rangle$ and a set of document encoding rules S is a total ordering of document objects in D with respect to the admissible transitions.

The *understanding* of the document with respect to a document model $\langle D, R, l \rangle$ and a set of document encoding rules S is the set of spatially admissible reading orders.

Following the above definitions, we see that inference is performed by two following steps. The first one is a constraint satisfaction step in which instances of bidimensional Allen relations are matched against document encoding rules expressed in $\mathcal{L}$. The second one is a graph sorting procedure similar to topological sorting.



(a) (b)

Fig. 8. A page from the Communications of the Association for Computing Machinery and a possible layout segmentation of it.

Consider the image from the magazine Communications of the Association for Computing Machinery presented in Figure 8.a. A possible segmentation of its layout (Figure 8.b) is formally represented by

```
[1, body\_of\_text, [13, 23, 93, 101],  Times, 11, 0, 16]
[2, body\_of\_text, [100, 23, 180, 101],  Times, 11, 0, 16]
[3, caption, [13, 107, 180, 122],  Arial, 11, 0, 16]
[4, graphics, [13, 122, 115, 183],  Courier , 11, 16, 0]
[5, figure, [115, 122, 180, 183],  None , 11, 0, 16]
[6, body\_of\_text, [13, 191, 93, 261],  Times, 11, 0, 16]
[7, body\_of\_text, [100, 191, 180, 261],  Times, 11, 0, 16]
[8, footer, [108, 267, 171, 270],  Arial, 7, 0, 16]
[9, page\_number, [175, 267, 180, 270],  Arial, 12, 0, 16]
```

where each element of the list represents one document object together with its layout and logical labeling information. The first element is a unique iden-

tifier, the second is the logical label, the third is the upper-left corner and the bottom-right corner of the bounding box, the fourth is the font of the text (if applicable), then the size of the font, the color of the font, and the last element is the color of the background.

Consider using the general document encoding rule (3). For all pairs of document objects labeled by "body_of_text", we consider their bidimensional Allen relation. Then we input these together with (3) into a constraint satisfaction solver. Obtaining the following set of admissible transitions

```
[1, 2], [1, 6], [1, 7], [2, 6], [2, 7], [6, 2], [6, 7]
```

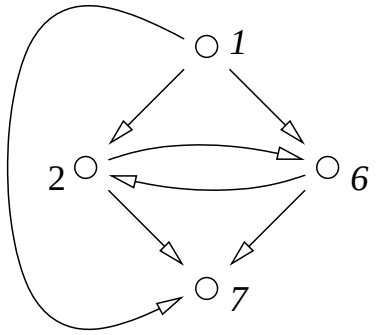


Fig. 9. The graph of spatially admissible transitions for the body_of_text document objects of the document in Figure 8.

One can view this as a directed graph of spatially admissible transitions, Figure 9. There are two possible complete total orderings of this graph. They are

```
[1, 6, 2, 7]        [1, 2, 6, 7]
```

Following the above definition, the two spatially admissible reading orders constitute the 'understanding' of the document in Figure 8.b with respect to the set of document encoding rules {(3)}. Once the set of spatially admissible transitions is identified, the task it that of totally sorting the graph. The algorithm to perform the sorting of directed transitive cyclic graphs is presented in Appendix A.

3.4 Complexity

The number of textual document objects that can be present in a page can vary greatly. It may go from a few in a simple one column page to more than 20 in complex multi-column pages and, in extreme cases such as big-format newspapers, to over 50. This generates a concern about the complexity of the methodology proposed here.

It turns out that the methodology has polynomial complexity in the number of document objects present in a page. Let us informally sketch here the reason for this.

The methodology is composed of two separate phases. First, all pairs of document objects are tested to see whether the document encoding rules are satisfied. This is a constraint satisfaction task involving boolean combinations of bidimensional Allen relations. Balbiani et al. (1998) have shown that this task has polynomial complexity. The constraint satisfaction is performed $n(n-1)$ times, where n is the number of document objects. Thus, the complexity of the first phase is still polynomial. Second, the set of all spatially admissible relations created in the first phase is sorted. This is performed using the algorithm presented in Appendix A having worst case complexity of $O(n^2)$. The complexity can be lowered to $O(n+m)$, where m stands for the number of edges in the graph of spatially admissible relations, but given the proliferation of edges this is not convenient in practice. In short, the overall complexity of the proposed methodology is polynomial.

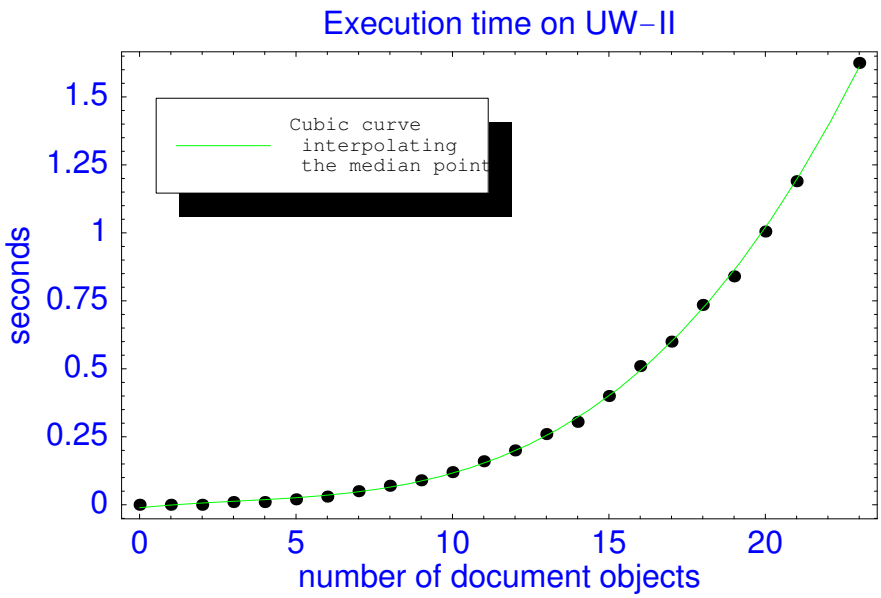


Fig. 10. Execution time in seconds with respect to the number of document objects in the pages from the UW-II dataset.

Additional evidence for the polynomial complexity comes from the experimentation with a prototype based on the methodology. The prototype was tested on the 624 pages of the University of Washington dataset UW-II (Phillips and Haralick, 1997). In Figure 10, the execution time of the prototype with respect to the number of document objects found in a page is shown. For the set of all documents with a given number of document objects the median execution time is represented by a bullet. The curve displayed on the graphic is the interpolating curve of the points. It is a cubic curve with small coefficients

$(-0.011 + 0.011n - 0.002n^2 + 0.0002n^3)$.

4 Evaluation

The methodology proposed has been implemented in a prototype system: `SpaRe`. The core of `SpaRe` is implemented in the declarative programming language Eclipse,[4] making use of the finite domain constraint satisfaction libraries.

To test `SpaRe`, we used the Media Team Data-Base (MTDB) from the University of Oulu, (Sauvola and Kauniskangas, 2000). The data set consists of scanned documents of various types: technical journals, newspapers, magazines, and one-page commercials. Elements from the data set are presented in Figure 11. We only used the documents in English, resulting in a data set of 34 documents having 171 pages. The MTDB data set has a ground truth at the document object level. Every document object has a layout label and a logical label. The reading orders are part of the ground truth. Of the 171 pages, 133 have a unique reading order, 32 have two independent reading orders, 5 have three, and 1 has four. We considered the layout information from the ground truth as the input to our system. As there is no ground truth for textual content and font information, we used the TextBridge OCR package from ScanSoft[5] to extract these.

For evaluation purposes, the documents in the data set were split into three main groups, based on their complexity:

- *trivial* documents containing up to 3 textual document objects;
- *regular* documents containing between 4 and 8 textual document objects;
- *complex* documents containing more than 8 textual document objects;

Out of 171 document pages, 98 are of type *trivial*, 66 of type *regular* and 7 are of type *complex*.

The goal of the experimentation was to evaluate whether `SpaRe` is effective in the detection of the reading order given the layout information. As subtasks, we were interested in evaluating the performance with different document encoding rules and with different values of the threshold for the thick boundary interpretation of bidimensional Allen relations.

The experiments consisted of three cases. In the first case, we have used the layout and labeling information from the ground truth and the general doc-

[4] `http://www-icparc.doc.ic.ac.uk/eclipse`.
[5] TextBridge SDK 4.5, `http://www.scansoft.com`.

(a) (b) (c) (d)

Fig. 11. Sample images from the MTDB data set.

ument encoding rule (3), denoted as *General Rule on Ground Truth data*. In the second case, we have used the layout and labeling information from the ground truth and the column and row-wise document encoding rules (4), denoted as *Column/Row Rules on Ground Truth data*. In the last case, we have used the layout and labeling information from an existing logical labeler (see Section 2) and the column and row-wise document encoding rules (4), denoted as *Column/Row Rules on the logical labeler data*. For each one of these we have varied the threshold of the thick boundary interpretation from 0 to 400 dots.

4.1 Criteria

To evaluate `SpaRe`, we use precision and recall (Baeza-Yates and Ribeiro-Neto, 1999). The set of reading orders detected (D) is compared to the ground truth. For 38 documents, the ground truth defines independent reading orders on non-intersecting subsets of the textual objects within the same document. In these cases, the reading orders are composed by one main sequence of document objects and one or two blocks to be read independently; e.g., a page containing a frame with independent text. To account for this portion of documents with multiple reading orders (20% of the whole data set), we consider a reading order correct if it is identical to at least one permutation of the independent reading orders as defined in the ground truth.

We refer to the set of permutations of the ground truth as the set of correct reading orders (C). Then, the precision and recall are defined as follows:

$$p = \frac{\mid \mathrm{D} \cap \mathrm{C} \mid}{\mid \mathrm{D} \mid} \qquad r = \frac{\mid \mathrm{D} \cap \mathrm{C} \mid}{\mid \mathrm{C} \mid} \tag{5}$$

The values lie between 0 and 1 inclusive, where 0 indicates the worst possible performance and 1 the best possible one. Because there is only one reading order, the recall can only be 1 if the correct reading is among the ones detected, or 0 if it is not. This makes the recall less informative of the overall behavior

of the system.

4.2 Results

We have evaluated the results in terms of the average precision and recall defined in Equation 5.

General Rule on Ground Truth data. We have used the general document encoding rule (3) on the ground truth layout and logical labels of the MTDB documents. The values of average precision with respect to increasing values of the threshold are shown in Figure 12.

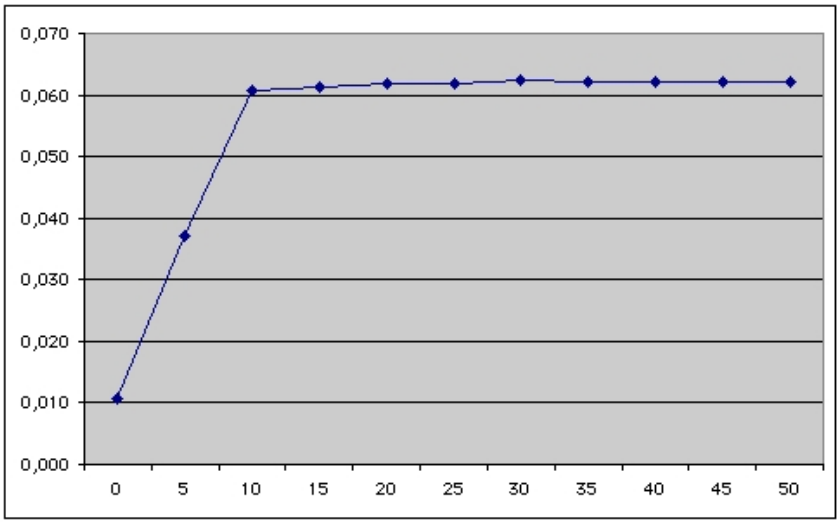


Fig. 12. Average precision for increasing threshold values (between 0 and 50) using the general rule on the ground truth of the MTDB data set. The maximum value is for the threshold value of 30.

The average precision and recall of the system for the entire MTDB data set for the threshold value of 15 are:

Document group	*Number of Documents*	SpaRe p	SpaRe r
trivial	98	0.96	0.99
regular	66	0.31	0.97
complex	7	0.003	1.00
average	171	0.06	0.98

SpaRe detected 2714 reading orders for the 171 document pages in the data set. In the case of a very rich and complex document, 2157 reading orders were detected. For other four documents, 140, 50, 37 and 15 reading orders

were detected. For the remaining collection the average of reading orders detected was of 1.74. In two cases, none of the reading orders as detected were correct.

Column/Row rule on Ground Truth data. We have used together the column and row-wise document encoding rules on the ground truth layout and logical labels of the MTDB documents. The values of average precision with respect to increasing values of the threshold are shown in Figure 13. The maximum value of precision is for the threshold value of 15.

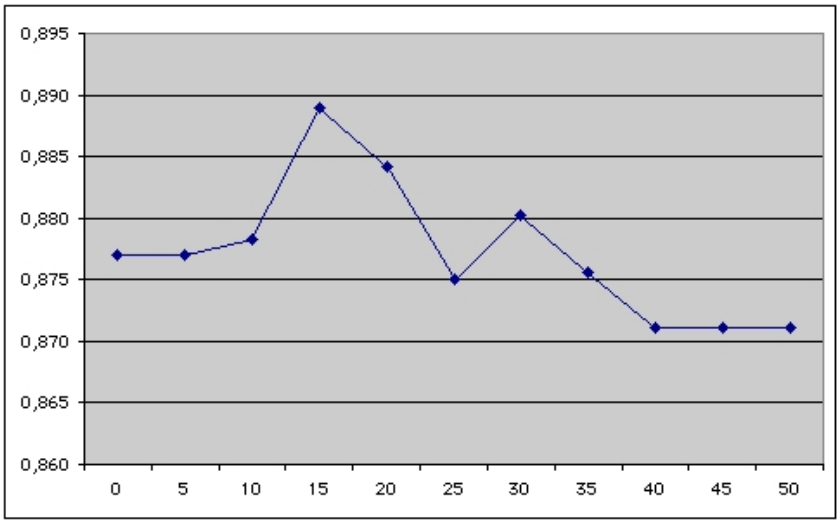


Fig. 13. Average precision for increasing threshold values (between 0 and 50) using the column/row rule on the ground truth of the MTDB data set.

The average precision and recall of the system for the entire MTDB data set for the threshold value of 15 are:

Document group	*Number of Documents*	SpaRe p	SpaRe r
trivial	98	0.97	0.99
regular	66	0.79	0.97
complex	7	0.88	1.00
average	171	0.89	0.98

SpaRe detected 190 reading orders for the 171 document pages in the data set. For 16 documents 2 reading orders were detected, including the correct one. In one case, none of the two reading orders as detected were correct. For one document, 4 possible reading orders were detected and none of them was correct. For the rest of 154 documents, SpaRe detected one reading order only and in one case this was not correct.

In the case of a two column scientific article composed of 6 textual document objects, SpaRe detected 4 reading orders. These were all wrong because

a short subtitle ("Acknowledgments") was too close to a white space in the neighboring column and was considered the title of the neighboring row in a row-wise reading. This row-wise connection was possible in four different ways, all incorrect. In case of a first page of an article in a magazine composed of 3 textual document objects, the title was on the left of the main text and centered vertically. In a reading order, the title was considered by `SpaRe` to be a subtitle of one of the two main bodies of text. It was placed incorrectly in the center of the reading order instead of on top of it. For one document composed of 4 textual document objects organized in one column with two subtitles and poorly typeset, `SpaRe` wrongly detected the reading order. The reason is that the subtitles were almost embedded in the main text and in *overlap* relation in the x axes instead of meet. The problem disappears when increasing the threshold value above 25 points.

The column-wise document rule has as one of its conditions that two blocks meet on the x axis. But with the boundary's thickness set to 0, this never occurs in the data set. On the other hand, allowing thickness, the *meet* relation holds among some neighboring document objects.

Column/Row on the logical labeler data. We have used the column and row-wise document encoding rules on the output of a logical labeling system on the MTDB documents. The values of average precision with respect to increasing values of the threshold are shown in Figure 14. The maximum value of precision is for the threshold value of 15.

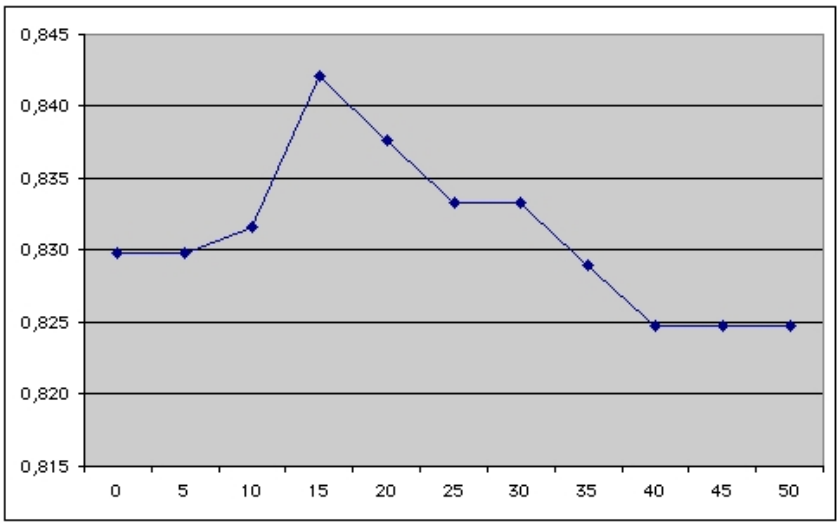


Fig. 14. Average precision for increasing threshold values (between 0 and 50) using the column/row rule on the data from the logical labeler.

The average precision and recall of the system for the entire MTDB data set for the threshold value of 15 are:

Document group	*Number of Documents*	SpaRe p	SpaRe r
trivial	98	0.92	0.94
regular	66	0.74	0.92
complex	7	0.86	1.00
average	171	0.84	0.94

SpaRe detected 192 reading orders for the 171 document pages in the data set. For 18 documents 2 reading orders were detected where the ground truth indicates only one. For one document, 4 possible reading orders were detected and none of them was correct. For the rest of 152 documents, SpaRe detected one reading order only. For 11 documents the correct reading order was not detected by SpaRe. In particular, for the simple documents 2 extra reading orders were detected and the number of wrongly understood documents was of 6. For the regular documents, the number of wrong detections was 5. For the 7 complex documents, there were no errors.

All additional misdetections of the reading order using the logical labeler data in place of the ground truth data are due to the misclassification of title objects. They are confused with footers, captions or rulers. The misclassification in the logical labeler data propagates to SpaRe. Eight additional documents are interpreted erroneously.

4.3 Discussion of the results

Varying the threshold in the thick boundary interpretation of Allen bidimensional relations does influence the overall performance considerably. In Figure 15, we compare the values of precision and recall for the three experimental cases increasing the threshold from 0 (no thickness) to 400 points. We notice that the precision increases considerably when the threshold goes from 0 to 5-10 points. Then it stabilizes showing minor variation over a wide range of thicknesses.

Moving the thickness from 0 to the maximum values corrects the situations in which boundary detection is not ideal. The reason for the stabilization of the precision between 15 and 100 points can bee interpreted as follows. In a document, document objects need not be found perfectly aligned. As far as the variation is small, the document layout is still intelligible. The acceptable variation depends on the specific document. For example, in a multicolumn document without overlapping frames, it is necessary to allow a small variation because the elements of a column will never be perfectly aligned; on the other hand, the variation should not go beyond half of the size of the white space

between two adjacent columns otherwise columns will be confused.

Letting the thickness grow much beyond 100, makes the precision fall down as the thickness becomes too big with respect to the average document block size. The document objects become 'blurred' entities and *overlap* becomes the most frequent relation. Performance degrades rapidly.

Considering the maximum values in Figure 12, Figure 13, and Figure 14, we notice that the maximum value is different for different rules.

The recall is stable and has always a high score between 0.9 and 1.0. This makes this measure of little interest in the presented experimentation. The reason for this high values resides in the fact that only one reading order is considered for the documents.

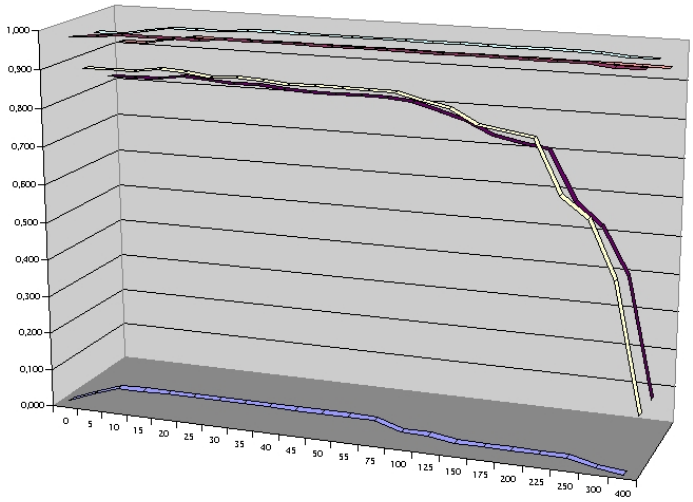


Fig. 15. Comparing precision and recall for the three experimental cases with respect to increasing threshold (from 0 to 400). From foreground to background, the recall for the general rule on ground truth data, the recall for column/row rules on ground truth data, the recall for column/row rules on the logical labeler data, the precision for the general rule on ground truth data, the precision for column/row rules on ground truth data, and the precision for column/row rules on the logical labeler data.

From the comparison of the use of the column and row-wise rules on the ground truth and on the logical labeler data (with threshold set to 15), one notices a small degradation of the overall performance. On the whole collection this means an appreciable decrease in performance, but not a total brake-down of the approach, as the precision goes from 0.89 to 0.84 and the recall from 0.98 to 0.94.

Considering the use of the general and the column and row-wise document encoding rules, one notices a big difference with respect to precision. The

problem with the general document encoding rule is its generality. It looses almost none of the correct readings of a document, but it finds too many. For instance, for a three column document with an image in the central column composed of 14 textual document objects, the general rule gives 2714 admissible reading orders while using the column-wise rule one gets only the correct one. When performing the experiment with the column and row-wise rules, we appreciate the sharp increase in precision, while the recall remains unmodified. This means that the rules are less general to detect less reading orders, but are not too specific to degrade the performance. Even on a heterogeneous collection of documents such as the MTDB, the column and row-wise rules have high values of recall and, most notably, precision. It is safe to conclude that the general rule is of no interest when compared with the column and row-wise rules.

The average execution time of `SpaRe` is appreciably fast. On a standard Sparc 300 Mhz machine, it takes about 28 seconds of wall clock time to process the whole data set. The median execution value for a document is of 10 milliseconds. The execution time increases more than linearly with the number of document objects. Therefore, there is a practical upper bound to the complexity and richness of document components that can be analyzed.

5 Conclusions

We have shown the feasibility, and efficacy, of applying a symbolic approach to logical structure detection in the context of document image analysis and understanding. The approach is based on a spatial language of rectangles and basic mereotopological rectangle relations (bidimensional Allen relations). Inference is achieved via constraint satisfaction techniques.

We have shown a bidimensional Allen based language to have appropriate expressive power for the task of document understanding. Though, what the language misses is a notion of neighborhood or some other kind of weak metric expressivity. Considering the 11% of the documents understood erroneously using the column and row-wise rules on the ground truth, one may argue that the correct order would have been captured by using a rule preferring neighboring text objects. Something not expressible in bidimensional Allen. In (Aiello et al., 2002), the first steps in this directions are takes by using Voronoi diagrams.

The logical labeler adds 4% of misclassified reading orders. Little can be modified in `SpaRe` to overcome these failures. When logical labels do not correspond with the actual logical function of the objects, any symbolic approach shows brittleness.

Two notable features of the presented symbolic approach are its flexibility and modularity. `SpaRe` is flexible enough to treat a wide variety of documents, including scientific articles, newspapers, magazines and commercial hand-outs, in a single run.

To increase the number of document classes handled, future work includes an extension to explicitly deal with independent reading orders. Independent reading orders are the case of complex documents, such as newspapers where pieces of text independent of one another coexist on the same sheet. The foreseeable key point of such an extension lies in the identification of appropriate document rules.

In conclusion, we do not know if HAL was equipped with a symbolic document image analysis system or with one based on different technologies. The only thing we know is that whenever a HAL-like machine will be available we expect it to read and understand the contents of any printed document brought to its attention.

References

Aiello, M., Monz, C., and Todoran, L. (2000). Combining linguistic and spatial information for document analysis. In Mariani, J. and Harman, D., editors, *Proc. of RIAO'2000 Content-Based Multimedia Information Access*, pages 266–275. CID.

Aiello, M., Monz, C., Todoran, L., and Worring, M. (2002). Document Understanding for a Broad Class of Documents. *International Journal on Document Analysis and Recognition*. Accepted with revisions. An earlier version appeared as (Todoran et al., 2001a).

Allen, J. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26:832–843.

Altamura, O., Esposito, F., and Malerba, D. (2001). Transforming paper documents into XML format with WISDOM++. *International Journal of Document Analysis and Recognition*, 4(1):2–17.

Arlazarov, V., Dinic, E., Kronrod, M., and Faradjev, I. (1970). On economical finding of transitive closure of a graph. *Soviet Math. Dokl.*, 11:1270–1272.

Baayen, R., Piepenbrock, R., and Gulikers, L. (1995). The CELEX lexical database (release 2). Distributed by the Linguistic Data Consortium, University of Pennsylvania.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley.

Balbiani, P., Condotta, J., and Fariñas del Cerro, L. (1998). A model for reasoning about bidimensional temporal relations. In Cohn, A. G., Schubert, L., and Shapiro, S., editors, *Proc. of the International Conference on Prin-*

ciples of Knowledge Representation and Reasoning (KR'98), pages 124–130. Morgan Kaufmann.

Cesarini, F., Francesconi, E., Gori, M., and Soda, G. (1999). A two level knowledge approach for understanding documents of a multi-class domain. In Hull et al. (1999), pages 135–138.

Cesarini, F., Gori, M., Marinai, S., and Soda, G. (1998). INFORMys: A flexible invoice-like form-reader system. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(7):730–746.

Esposito, F., Malerba, D., and Lisi, F. (2000). Machine learning for intelligent processing of printed documents. *Journal of Intelligent Information Systems*, 14(2/3):175–198.

Goossens, M., F., M., and Samarin, A. (1994). *The LaTeX Companion*. Addison-Wesley.

Hersh, W., Buckley, C., Leone, T., and Hickam, D. (1994). OHSUMED: an interactive retrieval evaluation and new large test collection for research. In *Proc. of ACM-SIGIR conference on Research and development in information retrieval*, pages 192–201. Springer.

Hull, J., Lee, S., and Tombre, K., editors (1999). ICDAR, IEEE.

Klink, S., Dengel, A., and Kieninger, T. (2000). Document structure analysis based on layout and textual features. In Murshed, N. and Amin, A., editors, *Proc. of International Workshop on Document Analysis Systems, DAS2000*, pages 99–111. IAPR.

Klink, S. and Kieninger, T. (2001). Rule-based document structure understanding with a fuzzy combination of layout and textual features. *International Journal of Document Analysis and Recognition*, 4(1):18–26.

Knuth, D. (1984). *The TeXbook*. Addison–Wesley.

Knuth, D. E. (1968). *The Art of Computer Programming*. Addison Wesley. Third edition 1998.

Knuth, D. E. and Szwarcfiter, J. L. (1974). A structured program to generate all topological sorting arrangements. *Information Processing Letters*, 2(6):153–157.

Lee, K., Choy, Y., and Cho, S. (2000). Geometric structure analsysis of document images: A knowledge approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(11):1224–1240.

Li, X. and Ng, P. (1999). A document classification an extraction system with learning ability. In Hull et al. (1999), pages 197–200.

Munro, I. (1971). Efficient determination of the transitive closure of a directed graph. *Information Processing Letters*, 1(2):56–58.

Nagy, G. (2000). Twenty Years of Document Image Analysis in PAMI. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(1):38–62.

Niyogi, D. and Srihari, S. (1996). Using domain knowledge to derive logical structure of documents. In Alferov, Z., Gulyaev, I., and Pape, D., editors, *Proc. Document Recognition and Retrieval III.*, pages 114–125. SPIE.

Phillips, I. and Haralick, R. (1997). Uw-ii english/japanese document image database. CD-Rom.

Randell, D., Cui, Z., and Cohn, A. G. (1992). A Spatial Logic Based on Regions and Connection. In *Proc. of Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'92)*, pages 165–176. San Mateo.

Reynold, L. (1979). *The Thames and Hudson Manual of Typography.* Thames & Hudson.

Rosenfeld, A. (1997). Eyes for Computers: How HAL Could "See". In Stork, D., editor, *HAL's Legacy*, pages 210–235. MIT Press.

Sauvola, J. and Kauniskangas, H. (2000). MediaTeam Document Database II. CD-ROM collection of document images, University of Oulu, Finland. http://www.mediateam.oulu.fi/MTDB/index.html.

Schiirmann, J., editor (1997). ICDAR, IEEE.

Singh, R., Lahoti, A., and Mukerjee, A. (1999). Interval-algebra based block layout analysis and document template generation. In *"Workshop on Document Layout Interpretation and its Applications"*. Event connected with Hull et al. (1999).

Toda, S. (1990). On the complexity of topological sorting. *Information Processing Letters*, 35(5):229–233.

Todoran, L., Aiello, M., Monz, C., and Worring, M. (2001a). Document Understanding for a Broad Class of Documents. Technical Report ISIS-TR-2001-15, University of Amsterdam.

Todoran, L., Aiello, M., Monz, C., and Worring, M. (2001b). Logical structure detection for heterogeneous document classes. In *Document Recognition and Retrieval VIII*, pages 99–110. SPIE.

Tsujimoto, S. and Asada, H. (1992). Major components of a complete text reading system. *Proc. of the IEEE*, 80(7):1133–1149.

van Benthem, J. (1983). *The Logic of Time*, volume 156 of *Synthese Library.* Reidel, Dordrecht. [Revised and expanded, Kluwer, 1991].

Walischewski, H. (1997). Automatic knowledge acquisition for spatial document interpretation. In Schiirmann (1997), pages 243–247.

Warshall, S. (1962). A Theorem on Boolean Matrices. *Journal of the ACM*, 9(1):11–12.

Worring, M. and Smeulders, A. (1999). Content based internet access to paper documents. *International Journal of Document Analysis and Recognition*, 1(4):209–220.

A Sorting transitive directed graphs

We extend the notion of *topological sorting* a directed acyclic graph (Knuth, 1968; Knuth and Szwarcfiter, 1974). Instead of a directed 'acyclic' graph, we sort a directed 'cyclic' graph whose edge relation is transitively closed. We call the latter *directed transitive cyclic graph.* More formally, a directed transitive cyclic graph is a graph $G = \langle V, E \rangle$ such that if $(i, j) \in E$ and $(j, k) \in E$, then

$(i, k) \in E$. In what follows, we assume that there are n vertices $|V| = n$ and m edges $|E| = m$. The problem of sorting a directed transitive graph G consists of creating sequences of nodes of the graph such that for any pair of nodes u and v in G appearing in any sequence, then (u, v) must be an edge of G.

Algorithms to perform topological sorting of directed acyclic graphs work iterating the following procedure until all nodes have been visited. First, a node v with no predecessors

$$\forall u \neq v \; \neg\exists (u, v) \in E$$

is identified. The node v is placed in the output. Then, all the edges (v, u) such that $\forall u \neq v \; (v, u) \in E$ are removed from the graph. In other words, the set of edges E of the graph is replaced by its subset $E/\{(v, u) \in E\}$ without the edges departing from the node v. If the original graph is acyclic, then the algorithm outputs a topological sorting of the input graph, otherwise the output is incorrect. The complexity of this sort of algorithms is $O(m + n)$. Notice that the algorithm does not return any clue on the incorrectness of the output in the case the input graph is cyclic. This is rather natural when considering the complexity of topological sorting and that of identifying cycles in directed graphs. It is well known that the latter is in NL-hard (see, for instance, (Toda, 1990)).

The algorithm for sorting transitive cyclic directed graphs takes as input a connected graph $G = \langle V, E \rangle$ and outputs a sequence of nodes $v_1 \cdot v_2 \cdot v_3 \cdot \ldots \cdot v_n$ such that:

(1) for all i: $v_i \in V$,
(2) $|v_1 \cdot v_2 \cdot v_3 \cdot \ldots \cdot v_n| = |V|$,
(3) for all $i \neq j$: $v_i \neq v_j$,
(4) if $i < j$: $(v_i, v_j) \in E$.

One starts by removing all self-loops $(v, v) \in E$ to setup the graph. Then the main cycle of the algorithm begins by considering all the nodes and counting the number of edges departing from each one, also known as *degree* of the node: $deg(v) = |\{(v, w) \in E | \; w \in V\}|$. Then one chooses a node with the highest degree, which has to be the same as the number of nodes of the graph minus one. In other words, the node is related — is 'before' — all other nodes of the graph. As we allow for cycles, there can be more than one node satisfying this condition. Once a node with maximal degree has been chosen, we remove it from the graph together with all the edges connected to it, both outgoing and incoming, and repeat the procedure on the remaining subgraph.

Consider the simple example in Figure A.1. The input graph is $G = \langle \{1, 2, 6, 7\}$ $\{(1, 2), (1, 6), (1, 7), (2, 6), (2, 7), (6, 2), (6, 7)\} \rangle$, it is easy to check that it meets the input conditions. The first step of the algorithm is to create a list of nodes

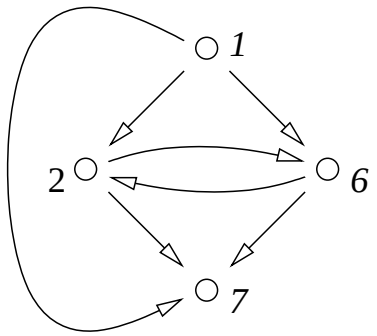


Fig. A.1. A simple directed transitive cyclic graph.

and their occurrences: $L = \{(1,3), (2,2), (6,2), (7,0)\}$. The node 1 is selected as first node of the output, as its degree is $3 = |V| - 1$. The list L is then updated to $L = \{(2,2), (6,2), (7,0)\}$. Two choices are possible at the following iteration: either 2 or 6. Suppose the first item is chosen, then L becomes $\{(6,1), (7,0)\}$. Finally, the output is updated with 6 and 7, respectively, yielding the final output of $\{1,2,6,7\}$ (also $\{1,6,2,7\}$ is a correct solution, and it can be computed by backtracking to the point in which v_2 was chosen in place of v_3).

Let us now proceed with a more precise definition of the algorithm. The preliminary step of the algorithm consists of the construction of a list L of pairs (v, o), where o is the degree of v, i.e., $o = deg(v)$. In pseudo-code, we have:

```
fail ← false;
for all v such that (v,v) ∈ E;
   E ← E/(v,v);
while(|V| > 0 and (not fail))
     sort L in descending order of occurrences
     % let (v*,l) be the first element of L
     if (l ≠ |V| − 1)
        fail ← true;
     else
        output ← output + v*;
        V ← V/v*;
        for all w such that (v,w) ∈ E or (w,v) ∈ E;
           E ← (E/(v*,w)) ⋂ (E/(w,v*));
        update(L);
```

Given that sorting a set of up to n values, each of which is an integer in the interval [0,n-1] can be performed with a bucket sort in $O(n)$, one can

conclude that the complexity of the proposed algorithm is in the $O(n^2)$ class. [6] If the algorithm terminates with `fail` set to false, then a correct sorting of the original directed transitive graph graph G will be found in the variable `output`. If no check is performed on the input graph, nothing can be said in case the algorithm returns true for the variable `fail`. On the other hand, if the input graph is tested to be transitively closed, then `fail` set to true indicates that no sorting for the input graph G exists. Algorithms to transitively close a graph can be found in the literature (Warshall, 1962; Munro, 1971; Arlazarov et al., 1970), and are also relatively inexpensive: $O(n^3)$, $O(n^{2.376})$, and $O(\frac{n^3}{\ln(n)})$, respectively.

[6] It is possible to devise an algorithm for directed transitive graph sorting with lower asymptotic complexity, though this is beyond the scope of the presented material. The steps of such an algorithm consist of: 1) finding strongly connected components of the graph, which is in $O(n+m)$; 2) consider the graph of the strongly connected components; 3) topologically sort the new graph. This algorithm has $O(n+m)$ complexity where n is the number of nodes and m the number of edges.