Extracting Visually Presented Element Relationships from Web Documents

Radek Burget and Pavel Smrž
Faculty of Information Technology, Brno University of Technology
Bozetechova 2, 612 66 Brno, Czech Republic
{burgetr, smrz}@fit.vutbr.cz

Abstract

Many documents in the World Wide Web present structured information that consists of multiple pieces of data with certain relationships among them. Although it is usually not difficult to identify the individual data values in the document text, their relationships are often not explicitly described in the document content. They are expressed by visual presentation of the document content that is expected to be interpreted by a human reader. In this paper, we propose a formal generic model of logical relationships in a document based on an interpretation of visual presentation patterns in the documents. The model describes the visually expressed relationships between individual parts of the contents independently of the document format and the particular way of presentation. Therefore, it can be used as an appropriate document model in many information retrieval or extraction applications. We formally define the model, we introduce a method of extracting the relationships between the content parts based on the visual presentation analysis and we discuss the expected applications. We also present a new dataset consisting of programmes of conferences and other scientific events and we discuss its suitability for the task in hand. Finally, we use the dataset to evaluate results of the implemented system.

1 Introduction

The World Wide Web is traditionally viewed as a web of linked documents. A great research effort has been put to the analysis and modeling of the relationships among the individual documents [11] or even analyzing semantic document relationships in order to obtain more information about the web organization [6, 7]. From this point of view, the documents are usually regarded as atomic units with certain properies such as the individual keyword frequencies. On the other hand, a remarkably less effort

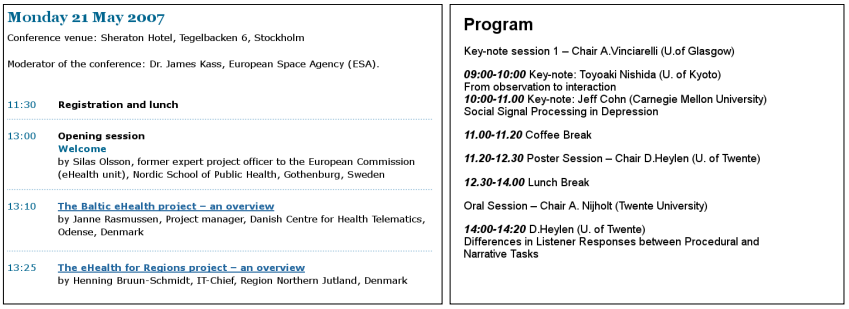

Figure 1: Different presentation styles of conference programmes

has been devoted to modeling the information structure in the documents themselves.

The WWW documents often present a structured information, that consists of multiple pieces of data of different kinds together with certain relationships among them. A typical example may be a conference programme that consists of speech titles, times, places and author names. However, the relationships are often not explicitly described in the document content. They are expressed by different, mostly visual means and the human reader is expected to interpret the visual presentation of the content appropriately in order to assign for example the appropriate author and time to a speech title.

Existing approaches to structured information identification in web documents are usually based on an analysis of a larger set of documents that follow the same presentation guidelines. Then, based on a set of sample documents, we may infer a set of rules that may be latter applied to other documents that follow the same guidelines. However, this does not solve a very frequent situation when we have a set of documents where each one comes from a different author and follows a different presentation style.

Let's consider two conference programmes presented in Fig. 1. Both documents provide an information about conference sessions, starting times, and the titles and authors of the individual presentations. However, this information is presented differently regarding the content layout, order of the individual data fields, colors and other properties and only a single exemplar of each such document is available. Moreover, the document formats may be different (for example, the HTML or PDF documents may be used).

Despite of the different formats, for a human reader, the presented relationships between the content elements remain the same and they correspond to the structure shown in Fig. 2. In this example, both documents assign some times and sessions to the individual speech titles and authors. These relationships are presented visually by different font properties, indentation and other means that allow the reader to interpret the relationships without reading the text or even without understanding the used language. We may expect that these logical relationships are similar in all the conference programmes independently on how they are actually presented. Generally, we may expect that the documents presenting data of the

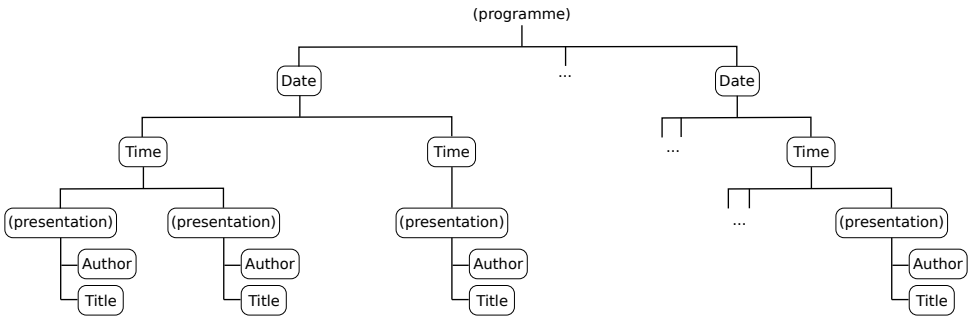

Figure 2: Expected logical structure of a conference programme

same topic will share the same logical relationships between the individual content elements although presented in different ways. To give more examples: Published articles present the relationships between the their title, authors, date of publication or even the sections and subsections. Timetables represent the relationships between the lines, places and times, etc.

In this paper, we propose a hierarchical *logical relationship model* that explicitly models the intra-document logical relationships that may be obtained by interpreting the visual presentation of the contents. This model has applications in information extraction, retrieval and other areas. Moreover, we address the problem of the automatic discovery of the logical relationships in a document. We analyze the visual presentation and content features that can be examined in order to obtain the logical relationship model. Lastly, we evaluate the proposed approach on real-world documents and we show that it can given comparable results for different document from various sources.

2 Related Work

In the area of information extraction from web documents, several authors already noted that the hierarchical relationships among the content blocks may be used for refining the identification of the particular information blocks in documents [1, 18]. However, in these methods, the content hierarchy is usually cunstructed based on some heuristics during the information extraction process and there is no explicit model defined for representing the intra-document relationships. Therefore, in this paper, we aim to create an application-independent model of the logical relationship constructed using a clearly defined algorithm.

The problem of the relationship representation and discovery in documents is closely related to the areas of document layout analysis and logical structure discovery. However, unlike our proposed logical relationship model, the logical structure discovery is usually domain-oriented as explained further.

Logical structure of electronic documents has been studied by many authors for quite a long time.

It is usually defined as a hierarchy of page segments that correspond to some visually distinguished components of its contents [5, 9, 13, 15, 17]. The logical structure should be distinguished from the *layout structure*: the layout (or geometric) structure models the relationships between the document segments based on their visual presentation; the logical structure focuses on logical relationships that are given by the expected meaning of the document segments [5, 17]. The resulting logical structure is commonly represented using hierarchical structures (trees) [5, 15] or grammars [17].

Discovering the logical structure usually includes assigning a meaning to the individual discovered components. The assigned meaning may be either a generic document section such as title, heading, footnote, list, etc. [5, 13] or a domain-specific meaning when focusing on a specific application [10, 12]. Some authors even limit the logical structure discovery to the discovery of important parts of the document without explicitly modeling their relationships; for example in [8], important parts of scholar articles are identified, which may be rather viewed as a classification task.

The discovery of the logical structure is usually based on an analysis of the page layout together with different visual features of the content. Shreve [13] notes that the logical structure reflects cultural norms of document organization and the logical relationships of document elements, and that the relationships of logical structure to physical layout are also culturally determined. Stoffel and Spretke [14] use the positions of the text lines, their indentation, spacing and font style for the logical document structure discovery. Similarly, in [5, 9], the document layout is analyzed in order to obtain the logical structure. In our older paper [1], we have also proposed a rule-based approach to HTML document code analysis for obtaining the logical structure.

Basically, the approach shared by all the mentioned approaches consists of three steps that are often analyzed together:

1. **Obtaining the physical layout** of the content by applying a kind of page segmentation or layout analysis algorithm.

2. **Transforming the layout to structure** based on pre-defined or learned rules (common document structure recognition in [5]).

3. **Interpretation of selected segments** by assigning a meaning to them based on given rules (domain dependent labeling [5]) or classification [8].

In our paper, we focus on the second step, i.e. creating a domain-independent document model that reflects the content presentation and interprets the cultural norms of the presentation as mentioned above. However, the domain knowledge is often necessary for a correct interpretation the logical relationships between the presented elements [9]. Therefore, in section 7, we also propose a way of incorporating a domain knowledge for improving the obtained logical structure.

On the other hand, unlike most of the mentioned methods, we don't claim to assign a meaning to the individual parts of the obtained structure (the third step listed above). We consider this to be one of the possible applications of the proposed model, as we mention in section 9.

3 Logical Relationship Model

When modeling the document structure, two kinds of models are usually distinguished [5, 17]: The *layout structure* (also called a *physical model*) describes the division of the document content to information blocks laid out on a page or pages. Generally, it is a hierarchical model that describes the basic blocks created for example by page headers, columns and other visually identifiable blocks in the documents. These blocks may be further divided to smaller sub-blocks. This model is domain-independent and it is created based on the page organization analysis.

On the other hand, the *logical structure* is domain-dependent: it assigns a meaning to selected parts of the document content and it represents the logical relationships between them as for example the hierarchical relationships between the semantic components such as headings, sub-headings and paragraphs in a document [15].

The purpose of the proposed Logical Relationship Model (LRM) is to provide a formally defined intermediate step between the layout structure and the logical structure. We claim that the interpretation of the visually presented relationships in the document should be separated from assigning the the meaning to the individual parts of the document.

The LRM represents the logical relationships between the individual parts of the document content as they are expressed by visual means and as they are interpreted by a human reader. In the same time, it remains domain-independent although the domain knowledge may be used additionally for refining the model as proposed in section 7. Then the LRM may be used as a general model of the document suitable for different applications.

3.1 The Layout and Logical Relationship Model

For defining the LRM, we will use the the layout (physical) model of the document as defined by many authors (e.g. [5]). Both the layout model and the LRM represent the relationships between *content elements*. With a content element we understand the smallest identifiable piece of the document content (usually text) that can be viewed as an atomic unit. For our purpose, we will define the content element as follows:

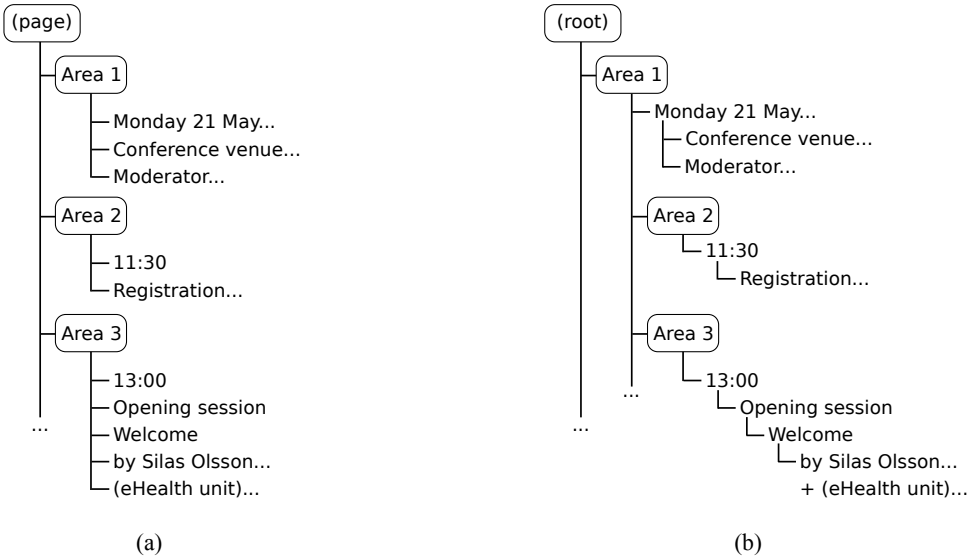

Figure 3: The layout and the logical structure of a conference programme

Definition 1 *(Content element).* Each displayed line of the text is considered to be a content element as long as it is visually consistent. For the text lines that consist of multiple parts with different visual properties, we consider the individual visually consistent parts to be separate content elements.

The difference between the layout model and the LRM is illustrated in Fig. 3. Both trees correspond to the first conference programme displayed in Fig. 1. The left tree shows the layout model. The whole page is split to several areas: a header (*Area 1*) and several items of the conference programme (*Area 2*, *Area 3*, etc.) visually separated by whitespace or other delimiters. The child nodes of each area correspond to the content elements or other areas that are placed inside the given area on the page. Further, we will call the detected visually separated areas the *content blocks*. As we can see from the example, the content blocks may be nested. The content elements then form the leaf nodes of the layout structure tree.

Definition 2 *(Content block).* A content block is a visually separated rectangular area in the page detected by the page segmentation algorithm. A content block may contain either other detected content blocks or directly the content elements placed inside of the given rectangular area of the page. Therefore, we may speak about a hierarchy of content blocks in the page.

The right tree shows the LRM. It consists of the same nodes representing the content blocks and elements. However, the parent – child relationships between the nodes correspond to their visually presented logical relationships. More specifically, the parent – child relationship in the LRM means that the child node is visually presented to be *logically subordinate* to the parent node.

Definition 3 *(Logical subordination).* Let b_1 and b_2 be two content blocks detected in the page. We

say that b_2 is logically subordinate to b_1 if the content of b_2 elaborates or concertizes the content of b_1.

The most common examples of logical subordination relationships in documents are for example title – subtitle – paragraph, term – definition, label – value, etc. In our example LRM in Fig. 3, *Area 1* is visually interpreted as the heading of the programme. Therefore, all the programme items are represented as child nodes of *Area 1* in the tree. Similarly, in each item of the programme (*Area 2* and *Area 3*), the time is used as a label introducing all the remaining information. We say that *Area 2* and *Area 3* are *logically subordinate* to *Area 1*. Similarly, in *Area 3*, we can see a deeper structure because two more content elements are present that may be interpreted as sub-labels: the session title and subtitle.

The layout structure is generally obtained as a product of page segmentation. The purpose of the segmentation is to detect the visually separated content blocks in the page and represent their nesting. The LRM is given by the expected reader's interpretation of the individual content elements. In our approach, the logical subordination relationships mentioned above are obtained from the analysis of the visual presentation only. That means, we analyze whether the content blocks are *presented to be logically subordinate* and we do not analyze their exact semantics. From this point of view, the LRM remains domain-independent.

The details of the visual presentation analysis are provided in section 4. In sections 5 and 6.3, we also provide the exact definitions of the processed layout model and the resulting LRM.

3.2 Expected Applications of the LRM

Creating a domain-independent formal model of the visually presented relationships in the document is motivated by the necessity of processing large sets of heterogeneous documents mainly for the following tasks:

Logical Structure Detection The visually presented relationships are essential for identifying the internal structure of documents, that means the headings, sub-headings, labels, etc. The generic model of these relationships may provide the information necessary for both the rule-based [5] and the machine learning-based [8] methods.

Information Extraction Explicitly described relationships in the document using LRM are suitable for extracting mainly structured data records. Then, it is possible to match the expected record structure with the structure presented in the page and to identify the content elements that correspond to the individual record fields as proposed for example in [1].

Information Retrieval The LRM may be also used for weighting the individual parts of the document during their indexing and retrieval. This may improve the retrieval results as shown for example

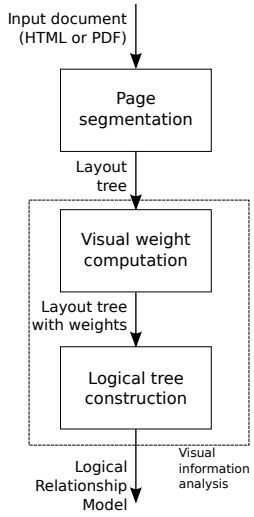

Figure 4: An overview of the LRM construction process

in [19].

We provide more details about the expected use of the LRM in the mentioned applications in section 9.

4 Overview of the Visual Analysis Approach

A general overview of our approach is shown in Fig. 4. First, the layout structure is obtained using a page segmentation algorithm and represented as a hierarchical *layout tree*. We provide the details of this process in section 5. Subsequently, this tree is further processed in several steps in order to build the LRM.

First, we assume that the logical relationships between the content elements are presented to the reader using some commonly used visual means that the reader is expected to interpret properly. These visual means include for example using different font sizes, colors or indentation. We use the visual features of the contents for assigning different weights to the individual content elements and we construct a basic *logical tree* based in these weights. The details of this process are provided in section 6.

Regarding the processed document formats, we consider HTML documents including the style information expressed by Cascading Style Sheets and PDF documents that are also frequently used on the web. These formats are supported by our implementation of the proposed method and were used for evaluation. However, the method is applicable to any document format that includes the textual information together with its visual presentation such as various popular office document formats.

5 Layout Structure Discovery using Page Segmentation

The purpose of page segmentation is to discover the individual content blocks in the page and to create the layout tree. Many segmentation algorithms have been proposed for processing the web documents as for example VIPS [3]. For our purpose we use the segmentation algorithm that we have published in [2] because it is applicable to any of the above mentioned document formats. However the further processing steps do not depend on the actual page segmentation method used.

The result of page segmentation is a tree model of the document layout structure. It can be defined as a tree:

$$L = (V_L, E_L) \tag{1}$$

where V_L is an ordered set of the tree nodes and E_L is a set of edges. The nodes in V_L correspond to the individual visual blocks detected in the page or directly to the individual content elements. E_L is a set of two-tuples that represent the visual nesting:

$$\forall v_i, v_j \in V_L : (v_i, v_j) \in E_L \Leftrightarrow v_j \text{ is directly nested in } v_i \tag{2}$$

This definition corresponds to Fig. 3(a) where the nodes correspond to the content elements (in case of leaf nodes) or content blocks (the remaining nodes). In case of more complex page layout, the tree depth is typically higher.

The segmentation usually defines an ordering of the tree nodes. For our segmentation algorithm, the node order corresponds to the order in which the individual content elements appear in the HTML or PDF code of the analyzed document (so called *document order*):

$$\forall v_i, v_j \in V_L : v_i < v_j \Leftrightarrow v_i \text{ precedes } v_j \text{ in the document code} \tag{3}$$

Based on this order, we can define the order of the child nodes of every non-leaf node from V_L.

6 Construction of the Logical Relationship Model

The aim of the document authors is to make the logical structure apparent to the reader. Therefore, we analyze the common visual means used for presenting the logical relationships between the content elements and subsequently, we use the results of this analysis for constructing the LRM.

6.1 Presentation of the Logical Structure in Documents

Logical structure of a document is presented to the reader according to existing cultural norms of document organization [13]. These norms represent the usual way of presenting the individual parts of the contents and their inter-relationships. For obtaining the basic awareness of the document structure, the following attributes of the content are important to a human reader:

Visual properties of the text The used font size, boldness, underlining or colors are often used for expressing the importance and even purpose of the individual content elements. For example, headings are usually clearly recognizable by their font size; often, there are even several levels of the headings distinguished in the document. Similarly, using a bold typeface or a different color indicates the importance of the element.

Content layout Very often, the logical relationships between the content elements are indicated by the mutual positions of the elements. The most frequent means used in this category include indentation that is commonly used for presenting the elements subordinate to another element or columns that group the related parts of the content together. Tables can be also viewed as a special case of the content layout with a defined meaning given by the relationships between the table header and the subordinate table cells.

In order to interpret the visual and layout properties, we assign weights to the individual nodes of the layout model as described in the following section. As the next step, the LRM is constructed based on these weights.

6.2 Assigning Weights to Layout Model Nodes

The visual and layout properties mentioned in the previous section allow the user to distinguish the level of logical superiority or subordination of the individual content blocks and elements. We express this level by assigning a *weight* to every node of the layout model defined in (1). A greater weight means that the corresponding content element appears to be more important in the document. For example, the main heading of a document should have the greatest weight, the sub-headings or smaller labels should have lower weights and plain document text that cannot be interpreted as a heading or label should have the lowest weight.

The weight is computed based on various visual properties of the text and the layout. Based on an analysis of real documents available on the web such as newspaper articles, conference programmes, timetables or even menus of the day, we have manually chosen a set of properties that are commonly used for indicating the logical superiority of the text. For each content element $v_c \in V_L$ (i.e. the leaf

node of the layout tree), we compute the values of these properties. Similarly, for each non-leaf node (a content block) $v_n \in V_L$, the value of the same property is computed as an average of the property values of its child nodes.

For any layout node $v \in V_L$, we compute the values of the following properties:

Font size Font size $fsize(v)$ is computed as a value relative to the average font size of the whole document. That means, for the visual nodes with greater font size than the average the resulting value is $fsize(v) > 1$; for example, $fsize(v) = 2.0$ means that the font used in v is twice as large as the average font size of the document. Similarly, $fsize(v) < 1$ means that the font size in v is smaller than the document average.

Font boldness The font boldness $fbold(v)$ is equal to 1 for the nodes that use bold font only, $fbold(v) = 0$ means that the node does not contain any bold text.

Colors During the analysis, we find all the text colors used in the document and for each used color, we compute the percentage of the text of the given color in the whole text content. The value $cperc(v) = 1$ means that the whole document content uses the same color as the node v. Small values of $cperc(v)$ mean that the color is quite rare in the document and therefore, it may indicate greater importance of v. During the color analysis, we apply a quantization to the color channels (we use four bits per red, geen and blue channel) so that very similar colors are considered as a single color.

Indentation The subordinate content elements are often indicated by their indentation. For each content block, we consider up to four levels of indentation based on the comparison of the mutual positions its child elements. The level 0 (not indented) child elements obtain $indent(v) = 1$, the elements indented by one step obtain $indent(v) = 0.75$, etc. Finally the elements indented by more than four steps obtain $indent(v) = 0$.

Centering We detect the child elements that are horizontally centered within their parent block by analyzing their position within the parent block and by comparing it with the positions of the preceding and following siblings. Centered elements have $center(v) = 1$, the remaining elements have $center(v) = 0$. When the element is centered, its indentation is not analyzed and we automatically consider $indent(v) = 1$.

Each of the above mentioned visual properties has different importance for computing the resulting weight. For example, the font size is the most important: a text written in larger font size is always interpreted to be superior to the text of a smaller font size. The relationships between the elements of the same size can be further distinguished by indentation etc.

In order to determine the actual importances of the individual properties computed above, we have analyzed a large set of web documents from the areas mentioned above and we have manually investigated how the content structure is presented i.e. what are the influences of the computed visual feature values (such as the font size, color, etc.) to the expected resulting weight of the layout nodes. As a result, we have obtained the following formula for computing the weight of a layout node $v \in V_L$:

$$\begin{aligned} weight(v) &= 1000.0 \cdot fsize(v) + 2.0 \cdot fbold(v) \\ &\quad +0.5 \cdot (1 - cperc(v)) \\ &\quad +5.0 \cdot indent(v) + center(v) \end{aligned} \tag{4}$$

This formula puts weights to the individual computed features according to their observed importance. The weight on font size is very high because a very small change in the relative font size has a great impact to the weight of the node. For example, with a 12pt average font size, 18pt text (i.e. $fsize = 1.5$) is usually the most important header that is recognizable independently on its remaining features.

The resulting formula and the chosen set of analyzed presentation features reflect the presentation styles common in western tradition. As we show later in section 8, it is generally applicable to a wide set of documents.

6.3 Logical Relationship Model Construction

The definition of the logical structure is similar to the definition of the layout structure (1). It is also defined as a tree:

$$S = (V_S, E_S) \tag{5}$$

where V_S is a set of tree nodes; $V_S = V_L$ as defined in (1). E_S is a set of tree edges that represent the logical relationships between the tree nodes. It is constructed using algorithm 6.3.

This algorithm recursively goes through the layout tree L. The functions `firstChild()`, `nextChild()` and `parent()` are used for obtaining the child and parent nodes in L. For each node $v_n \in V_L$, we compare the weights of its child nodes and we try to find the most appropriate parent node for each child node. Then, we add the appropriate two-tuples (v_i, v_j) to the resulting set E_S. For each $(v_i, v_j) \in E_S$ it must hold that $weight(v_i) > weight(v_j)$ and we try to find the closest parent element where this condition is met.

Algorithm 1 Logical relationship tree creation.

createLogicalStructure($v_n \in V_L$)

v_{cur} = firstChild(v_n)
while v_n has more child nodes
v_{next} = nextChild(v_n)
createLogicalStructure(v_{next})
v_{par} = parent(v_{cur})
Δ_{cur} = |weight(v_{next}) − weight(v_{cur})|
Δ_{par} = |weight(v_{next}) − weight(v_{par})|
if $\Delta_{cur} \leq \Delta_{par}$
if weight(v_{next}) < $weight(v_{cur})$
add (v_{cur}, v_{next}) to E_S
else
add (v_{par}, v_{next}) to E_S
end if
else
v_{anc} = closest ancestor of v_{cur} in L where
weight(v_{anc}) > weight(v_{cur})
add (v_{anc}, v_{next}) to E_S
end if
$v_{cur} = v_{next}$
end while

As the result of this last step, we obtain a domain-independent model of the logical relationships as they come from the interpretation of the visual presentation of the content.

7 LRM Refinement by Adding Domain Knowledge

In some cases, using an additional domain knowledge is necessary in order to interpret the logical relationships correctly. The most important problem is to detect the content elements in the layout model that form a single logical entity. In the first conference programme shown in Fig. 1, the names and affiliations of the authors are often presented in two lines, that means two separate content elements according to definition 3.1 as shown in Fig. 3 (a). However, from a logical point of view, they form a single logical element as shown in Fig. 3 (b). This is particularly important for headings that span for multiple lines and which need to be represented as a single node in the logical structure in order to be

able to assign the child nodes appropriately.

The basic problem here is to decide whether two or more neighboring visual blocks form a single information entity (e.g. a title split to several pages) or separate entities with different meaning (e.g. a title and author). Partially, this problem may be solved in the page segmentation phase by merging the neighboring content elements having the same visual style. Then, we obtain larger, visually consistent content blocks from the page segmentation as we have presented in [2]. However, in some cases, the presentation style of two different elements may be equal and the reader is expected to recognize the different nature of the presented information which is often domain-dependent (e.g. the personal names of the article authors). In other words, the reader is expected to recognize and distinguish the most common types of information from the given domain such as dates, times, personal names, places, etc. In such cases, it is not sufficient to analyze the visual presentation; an additional knowledge about the text content itself must be included into the LRM.

As a generic way of representing the additional information about the text, we propose adding *tags* to the individual nodes of the LRM that represent the domain knowledge about the content. Based on these tags, we may recognize the groups of nodes that possibly form a single information entity and we may refine the LRM. The whole process then consists of two phases:

Logical tree tagging To each node $v \in V_S$, a set of tags is assigned. These tags indicate the nature of the its content as for example `time`, `authors`, `locations`, etc. They are domain dependent and they are assigned based on the analysis of the text for example using regular expressions or by employing a NER classifier. The tags and the way of their assignment must be chosen according to the application domain. The details for our testing domain of conference programmes are provided in section 8. For other domains, a different set of tags should be used.

Tree node merging We go through the tree S and for each subsequent nodes $v_i, v_j \in V_S$ that share the same parent node, we compare the tag sets assigned to these nodes. If the intersection of the tag sets is not empty, we replace these nodes with a single node that contains the text content and the child elements of the two nodes. Similarly, a longer sequence of nodes may be reduced to a single one.

The tagging itself may seem to be closely related to logical structure detection or information extraction tasks where the meaning of some content parts is also analyzed. However, in our approach, the tagging is only used for a quick and approximate estimation of the element purpose in order to refine the LRM. As it comes from the above descriptions, only the tags of the neighboring areas are compared and therefore, we do not require a great tagging accuracy across the whole model.

For example, a simple regular expression is sufficient for tagging the speech titles in conference

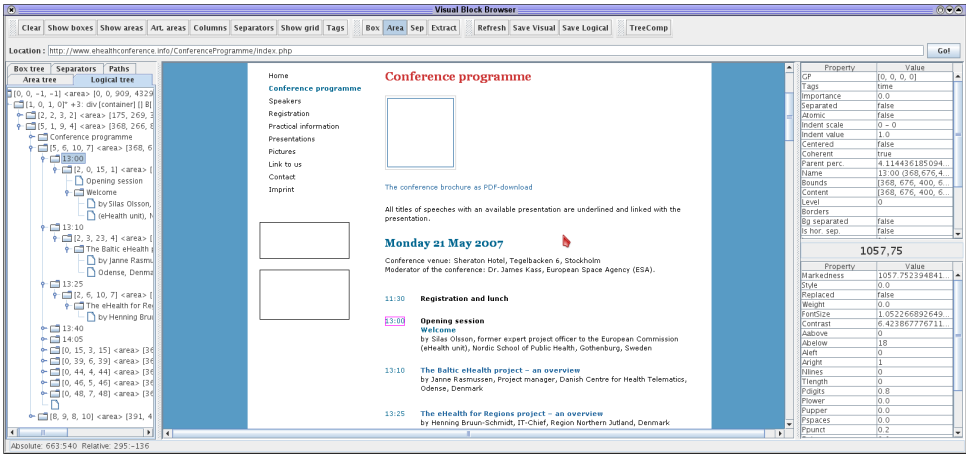

Figure 5: An interactive appliaction used for evaluating the obtained Logical Relationship Models

programmes because its purpose is not to discover all the speech titles accurately but to mark the content elements that "might look as a speech title" at first glance. For a precise information extraction, a thorougher analysis of the LRM must be implemented as we discuss further in section 9.

After the refinement process, the resulting LRM should correspond to the reader's interpretation of the hierarchical logical relationships in the document.

8 Method Evaluation

We have implemented the proposed method of the LRM construction in Java. For the page segmentation we have used our implementation of the segmentation algorithm published in [2]. This implementation is based on the CSSBox[1] rendering engine that is able to render both the HTML and PDF documents. The resulting tool has both the graphical and web-based user interfaces that allow to visually investigate the obtained layout structure and the LRM and their relationship to the actual page presentation (see Figure 5 and 6). The aim of the evaluation is to show, that the LRM may be used as a presentation-independent model of documents, i.e. that the structure of the obtained models is comparable for the documents from the same domain independently on how the document format and the presentation style.

As the testing domain, we have chosen the conference programmes. We have analyzed 68 different conference programmes from different areas (computer science, health, etc.) manually downloaded from the web. Each of the programmes uses a different way of the visual presentation of the given information. There were 7 PDF documents and 61 HTML pages in the analyzed set.[2]

For assigning the labels that represent the additional domain knowledge to the content according

[1] `http://cssbox.sourceforge.net`

[2] A complete table with the programme URLs and their evaluation and the copies of the documents are available at `http://www.fit.vutbr.cz/~burgetr/publications/ijcini`

Table 1: Results of the Relationship Identification in Conference Programmes

Relationship	Expected	Discovered	Ratio
Date – Time	44	35	0.80
Time – Title	68	56	0.82
Names	68	64	0.94
Total	180	155	**0.86**

to section 7, we have used the Stanford NER classifier [4] for recognizing authors (personal names) and locations (that are recognized but not used for evaluation). For recognizing the content elements containing the remaining data (*date*, *title*, etc.), we have used regular expressions.

For the evaluation, we have expected that each programme contains the information about the individual speeches. We have focused on the speech *date*, *time*, *title* and the *names* of the authors or presenters. We have processed all the documents with the described algorithms in order to obtain their LRMs and we have manually investigated whether the expected relationships shown in Fig. 2 have been successfully detected and represented in the model. We have checked the following three relationships:

Date – Time Typically, there are multiple presentations or sessions taking place in the same day and this is usually taken into account in the presentation of the programme. The content element containing the date should be an ancestor node of the element containing the time of the same presentation. In some cases, the date is not explicitly presented in the programme (especially in case of one-day workshops), i.e. this relationship does not have to be present in the document.

Time – Title In the programmes the time is usually assigned to the individual speeches or to a whole session (one time element shared by multiple speeches). In both cases the content element containing the time should be an ancestor of the element containing the speech title.

Names (Time – Names or Title – Names) The content elements containing the author names and the speech title should share the same *time* ancestor element. In case the Time – Title relationship was not properly detected for some reason, it would not be possible to evaluate this relationship. In that case, we have checked whether the author names can be uniquely assigned to the correct speech in the obtained LRM, that means there exists a parent-child relationship between the *title* and *authors* elements of the same speech or they both share a unique ancestor element marked *(presentation)* in Fig. 2.

We expect that all the relationships are presented consistently in a single programme. Therefore, we have considered the LRM structure to be correct, if the relationships are present in the LRM for at least 90 % of the speeches presented in the programme. This allows a few speeches presented in an unexpected way (e.g. special highlighting, etc.) The results of the evaluation are shown in Table 1. We

have expected the Date – Time and Time – Title relationships to be present in all 68 documents, the date was contained in 44 documents. From the total 180 relationships, 155 were correctly detected that gives the overall success ratio of 86 %.

As for the incorrectly processed documents, the most common problem is a more complex way of data presentation than expected – typically a table with a more complicated (two-dimensional) structure that is not analyzed properly. In a few cases, there was an insufficient weight assigned to the *time* content blocks which caused the time to be represented as a sibling of the title and authors instead of the ancestor. This indicates that the evaluation of the presentational cues could be further improved.

9 Logical Relationship Model in Applications

The motivation for creating the LRM was to provide a general document model for the tasks mentioned below. These tasks are usually solved separately; however, from the point of the LRM usage, they overlap significantly.

9.1 Logical Structure Detection

When discovering the logical document structure, the task is usually to recognize the important parts of the documents such as headings, labels, publication dates or authors and to model their relationships. Existing approaches are based either on pre-defined rules [5] or machine learning methods (for example, Conditional Random Fields approach is used in [8]). In both cases, LRM may provide an important information regarding the visually presented relationships between the individual content parts that is not directly available in the document. This information may be used for the rule construction (e.g. the author must be subordinate to an article heading) or for an automatic classification. This approach may be combined with further classification of the elements based on different criteria (such as font properties) as proposed for example in [2].

9.2 Information Extraction

Information extraction presents the most important expected application of the LRM. With the hierarchical LRM, the information extraction problem may be viewed as a generalization of the logical structure detection. While in the logical structure detection, the task is to identify the headings and labels, in information extraction, the task is to identify generally any information contained in the document such as for example names, presentation titles, personal data, etc. The visually expressed relationships may provide an important cue for identifying the pieces of data that form a single extracted record.

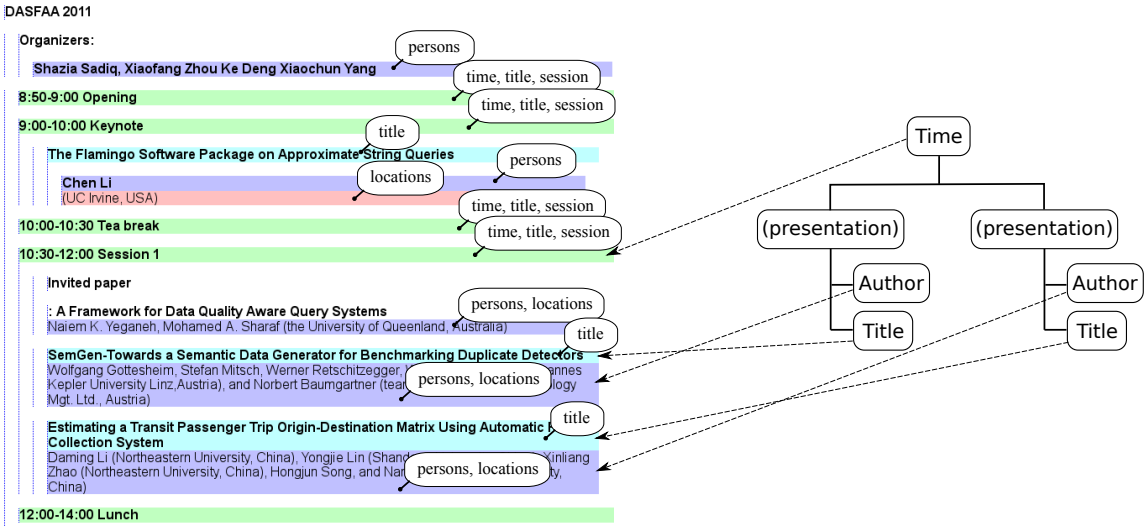

Figure 6: Structured record extraction using LRM and tree matching

Since the LRM represents an explicitly and formally described structure of a document, it can be used for identifying a particular information element in a way similar to the above logical structure detection task. When an approximate content element tagging is applied to the LRM as proposed in section 7, the logical relationships represented in the LRM may be used for the disambiguation of the meaning of the individual content elements. For this, the tree matching methods may be used that compare the individual LRM subtrees with the expected structure of the information to be extracted in order to identify complete data records. Based on this comparison, we may decide which relationships are (not) acceptable in the extracted records and we may choose the most probable candidate elements that cotnain a particular information.

This approach is demonstrated in Figure 6. The left part shows the output LRM tree obtained from a real conference programme using our implemented tool and visualised using a web interface. The different background colors and the annotaions show the tags that have been assigned to the individual LRM nodes as described in section 7. We may note that the tagging gives only an approximate information in this stage. For example, an erroneous speech title (invited paper) starting with a colon has not been detected properly. Further, it is usually difficult to distinguish the session titles and the speech title without some additional heuristics or deeper natural language analysis. Therefore, the session titles have both the `title` (the speach title) and `session` (the session title) tags meaning that the particular node may be potentially interpreted in both ways. In addition, the same nodes have the `time` tag assigned because they also contain the time information in this case.

The right part of the figure shows the expected data record structure. Using a tree matching algorithm as proposed for example in [1], we search for the best matching subtrees in the LRM. Using the

approximate matching that allows some missing or overlapping tags, we may resolve the above mentioned duplicite and missing tags. Compared to the previous work [1], LRM provides a more generic, formalized and robust model of the document structure while the information extraction process remains very similar. Our preliminary experiments with the tree matching algorithms show that the combination of the logical relationship model and the text analysis including the NER tagging gives very promissing results in the extraction of structured records from documents.

Similarly, in [18], the discovered hierarchical relationships are used as an input for a Tree-structured Conditional Random Fields classifier [16]. Again, our proposed LRM may be used to formalize this approach in a similar way.

9.3 Information Retrieval

Using the LRM for information retrieval tasks allows to consider the document structure in the document indexing and retrieval. Thus, this task overlaps with the logical structure detection as well. For example in [19], page segmentation is used to recognize the important parts of the page to be indexed. With the LRM, we may assign weights to the individual content elements based on their position in the LRM tree (for example based on the distance from root node or from some recognized important node such as main heading). This may be used for distinguishing the relevance of the individual parts of the document contents. During the document retrieval, the LRM may be used for answering structured queries (e.g. retrieving documents containing certain keywords in a particular hierarchical relationship) or for evaluating the similarity of document structure.

10 Conclusions

In this paper, we have proposed a format and presentation-independent document model (LRM) that represents the visually presented relationships in the document contents. We have also presented a method of extracting the logical relationship model from HTML and PDF documents based on the interpretation of their visual presentation. The obtained model may be further refined using a domain-dependent text content analysis including a NER classification.

In order to evaluate the proposed LRM constructuion methods and to show the generality of the model, we have created a testing set of conference programmes in various formats and with very different presentation styles and we have proposed evaluation criteria regarding the discoevred resulting structure. We have also implemented an interactive tool that allows to investigate the obtained structure and compare it with the actual presentation in the document.

The obtained results show that the method gives comparable results for documents from different

sources that use different formats and visual presentation. This demonstrates that the Logical Relationships Model may be used as a presentation-independent model for further document processing such as logical structure discovery, information retrieval or extraction.

Finally, we have suggested a way of using the LRM in the most important web document processing tasks with a focus to information extraction based on tree matching algorithms. Our preliminary experiments show that the explicitly modeled content element relationships represented by the LRM may be very useful mainly for extracting structured records from the documents.

Acknowledgements

This work was supported by the research programme MSM 0021630528 and the IT4Innovations Centre of Excellence CZ.1.05/1.1.00/02.0070. The research leading to these results has received funding from the European Community's 7th Framework Programme FP7/2007-2013 under grant agreement number 270001 – Decipher.

References

[1] R. Burget. Hierarchies in HTML documents: Linking text to concepts. In *15th International Workshop on Database and Expert Systems Applications*, pages 186–190. IEEE Computer Society, 2004.

[2] R. Burget and I. Burgetová. Automatic annotation of online articles based on visual feature classification. *International Journal of Intelligent Information and Database System*, 5(4):338–360, 2011.

[3] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. *VIPS: a Vision-based Page Segmentation Algorithm*. Microsoft Research, 2003.

[4] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, 2005.

[5] S. Klink, A. Dengel, and T. Kieninger. Document structure analysis based on layout and textual features. In *Proc. of International Workshop on Document Analysis Systems*, pages 99–111, Brazil, 2000. IAPR.

[6] X. Luo, Z. Xu, Q. Li, Q. Hu, J. Yu, and X. Tang. Generation of similarity knowledge flow for intelligent browsing based on semantic link networks. *Concurrency and Computation: Practice and Experience*, 21(16):2018–2032, 2009.

[7] X. Luo, Z. Xu, J. Yu, and X. Chen. Building association link network for semantic link on web resources. *Automation Science and Engineering, IEEE Transactions on*, 8(3):482–494, 2011.

[8] M.-T. Luong, T. D. Nguyen, and M.-Y. Kan. Logical structure recovery in scholarly articles with rich document features. *IJDLS*, 1(4):1–23, 2010.

[9] A. Namboodiri and A. Jain. Document structure and layout analysis. In B. B. Chaudhuri, editor, *Digital Document Processing*, Advances in Pattern Recognition, pages 29–48. Springer London, 2007.

[10] M. Nojoumian and T. C. Lethbridge. Extracting document structure to facilitate a knowledge base creation for the uml superstructure specification. In *Proceedings of the International Conference on Information Technology*, pages 393–400, 2007.

[11] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.

[12] R. Rauf, M. Antkiewicz, and K. Czarnecki. Logical structure extraction from software requirements documents. In *Requirements Engineering Conference (RE), 2011 19th IEEE International*, pages 101 –110, 2011.

[13] G. M. Shreve. Corpus enhancement and computer-assisted localization and translation. In K. J. Dunne, editor, *Perspectives on Localization*, pages 309–332. John Benjamins Publishing Company, Amsterdam/Philadelphia, 2006.

[14] A. Stoffel, D. Spretke, H. Kinnemann, and D. A. Keim. Enhancing document structure analysis using visual analytics. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 8–12, New York, NY, USA, 2010. ACM.

[15] K. Summers. Toward a taxonomy of logical document structures. In *In Electronic Publishing and the Information Superhighway: Proceedings of the Dartmouth Institute for Advanced Graduate Studies*, pages 124–133, 1995.

[16] J. Tang, M. Hong, J. Li, and B. Liang. Tree-structured conditional random fields for semantic annotation. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *The Semantic Web - ISWC 2006*, volume 4273 of *Lecture Notes in Computer Science*, pages 640–653. Springer Berlin Heidelberg, 2006.

[17] H. Yashiro, T. Murakami, Y. Shima, Y. Nakano, and H. Fujisawa. A new method of document structure extraction using generic layout knowledge. In *International Workshop on Industrial Applications of Machine Intelligence and Vision*, pages 282–287, Tokyo, Japan, 1989.

[18] Y. You, G. Xu, J. Cao, Y. Zhang, and G. Huang. Leveraging visual features and hierarchical dependencies for conference information extraction. In Y. Ishikawa, J. Li, W. Wang, R. Zhang, and W. Zhang, editors, *Web Technologies and Applications*, volume 7808 of *Lecture Notes in Computer Science*, pages 404–416. Springer Berlin Heidelberg, 2013.

[19] S. Yu, D. Cai, J.-R. Wen, and W.-Y. Ma. *Improving Pseudo-Relevance Feedback in Web Information Retrieval Using Web Page Segmentation.* Microsoft Research, 2002.