A Method Combining Syntax Analysis and Correction Rules to Re-construct the Re-flowable Document

Zhen ZHANG¹,²,*, Ning LI¹,², Ying-ai TIAN¹,² and Si GENG¹,²

¹Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, Beijing Information Science and Technology University, Beijing 100101, China
²School of Computer, Beijing Information Science and Technology University, Beijing 100101, China

*Corresponding author

Abstract. To improve the shortcomings of fault-tolerance ability in the re-construction method for re-flowable document structure, a new method combining left-corner method and correction rules is proposed, where the xml schema is applied to construct a syntax tree of typesetting rules of document components, and left-corner method is applied to analyze the logical components of the document supervised by the syntax tree. In the analysis process, the correction rules are used to correct the possible errors existed in document component and eventually get the most likely document structure. The results show that the algorithm can effectively improve the fault tolerance in the document structure reconstruction and the accuracy of document structure recognition, which forms the foundation for document understanding and format checking.

Introduction

Re-flowable document structure carries precious semantic information[1], which plays an important role in document understanding. For example, It is more easily to a computer automatically extracts the outline of a document if the computer knows the structure of the document. The document logical components represent the semantic information of a document paragraph, such as abstract, first-level heading. The method of document structural reconstruction is aim to analyze the hierarchical relationship of logical components of document. However, because of the complexity of the re-flowable document typesetting format, most of the re-flowable documents have a certain error, such as the heading style is not normal or the figure has no caption, these errors will cause the document components missing, redundancy and so on, the precious document structural reconstruction methods which have no fault-tolerate can’t process the above problem, and will get the wrong document structure. So an efficient fault-tolerant method is important to document structural reconstruction.

This paper focuses on the documents whose hierarchical relationships of logical components meet certain typesetting rules, the typesetting rules define the nesting structures of the components, the order and number of the logical components occurrence. This paper first constructs a typesetting rule syntax tree for document components. Second, based on this syntax tree, we use left corner analysis method to analyze document components. At the same time, the correction rules are used to improve fault tolerance when the analysis process occur error.

Related Work

Re-constructing the structure of a re-flowable document is a well-known research problem, Zhao et al.[2] proposes an approach for re-constructing re-flowable document structure based on a directed graph, which applies the single source shortest path algorithm. Similarly, Li et al. [3] presents an approach based on a finite state machine, which applies the state transition function to control the
sequence of document structure recognition. In both methods, they premise that the document component is absolutely right. Hence they can not get the accuracy result once the document has error. There is less relatively research on fault tolerance of re-flowable document structure re-construct. Wu et al.[4] defines a document re-construct model with fault tolerance, which uses the best priority search method to build document logical structure. Structural semantic labeling of Natural Language Processing [5] describes the relationship between elements and other elements in a sentence. In the semantic labeling research of fault-tolerant, semantic role labeling system fusion is usually used to reduce the effect of syntax errors on semantic role labeling. Zhuang et al. [6] proposes a method of minimum error weighted system fusion to give different weights for each system's labeling results. In the study of text proofing, Chinese automatic proofing technology corrects syntactic errors according to contextual information, Zhang et al.[7] proposes a method of combining grammatical analysis with words matching, which uses local corpus statistics and maximum matching algorithm to find scattered strings. Zhang et al.[8] constructs the semantic collocation knowledge base, then a top-down method is used to find the possible errors.

The goal of this paper is to propose a better performance algorithm, which can efficiently analyze the structure in the case of document logic components have error. In the process of document structure reconstruction, we need to solve the following key problems:

First, the logical components of documents belong to the metadata of documents, and a document metadata model needs to be constructed. The document metadata model need define logical components, describe document components, and define the order and number of occurrences or nested structures of logical components, and needs to be easily parsed.

Secondly, the structure analysis should be carried out efficiently. The structure re-construct method needs to avoid large, meaningless nodes traversing, and it also needs to prevent the error of the components hierarchical relationship.

Furthermore, the fault tolerance mechanism can handle common errors in document structure reconstruction. we need to find users' wrong writing habits, then add fault tolerance rules to judge users' writing intentions as accurately as possible.

Construction of Document Metadata Model

In the process of document structure reconstruction, the document metadata model need determine the order, number of occurrences and nesting structure of logical components, and finally form a syntax tree which is used to analyze the correctness of the logical structure. For the task of automatic recognition of constructing document metadata model, there are two main categories of approaches: template based method [3]and grammar based method[9]. Template based methods will have a certain impact on document structure reconstruction once the template is misplaced. However, In the grammar based method, we can use xml schema to define the attributes, data types, elements and their inclusion relations of document metadata, and the schema is easy to parse, so this paper uses xml schema to build the document metadata model. For example, suppose the logical component of an academic document is set as {abstract, text, keywords, first-level heading, list, figure, table, figure caption, table caption, second-level heading, third-level heading}, Figure 1 shows the typesetting rules syntax tree of logical components.

As the Figure1 shows, each article contains two parts: header and chapter, and these two parts appear sequentially. The header part contains three parts: abstract, text and keywords, each part must appear in the order. Chapter part can appear many times, and includes first-level heading, content, second-level chapter, the content and the second-level chapter are optional nodes, content contains text, list, figure, figure caption, table and table caption, which is optional node and can appear many times, second-level chapter is optional node, but if the node appears, the node must appear at least twice, the second-level chapter and the third-level chapter are similar to chapter.

The above document metadata model construction method takes academic document as an example, but this method also applies to other different types of documents. For example official
document and patent document, although the type of these documents is different, but there are certain typesetting rules in the document components, so we can use above method to construct the metadata model.

**Document Structure Reconstruction based on Left-corner Analysis Method**

In the previous section, we constructed the syntax tree of document typesetting rule, which can guide the process of document logical component analysis. In order to analyze the structural more efficiently, the process of document logic component analysis need avoid large and meaningless node traversal problems. In the research of sentence parsing, there are three main categories of sentence parsing: top-down method, bottom-up method and left-corner analysis method[12,[11]. The left-corner analysis method not only can avoid the exhaustive non-end extension of the top-down method, but also can avoid the problem of blind reduction of the parent node in the bottom-up method, which can effectively solve the problem in the traversal process of the document logic components.

![Figure 1. The typesetting rules syntax tree of the logical component of an article.](image)

The key of left-corner analysis method of the document structure is the left corner of the syntax tree, for the non-terminator symbols A and B, if $A \rightarrow BA$, So B is the left corner of A [10]. As we can see in Figure1, the "second-level heading" is the left corner of the "second-level chapter". The analysis process will not parse the rule if the current input component is not the left corner of a rule in the syntax tree, which can avoid a large number of meaningless node traverses. The process of document structure reconstruction is guided by the syntax tree. Document structure can be subdivided into several substructures, a non-leaf node of syntax tree represents a substructure. We define all of non-leaf nodes as state nodes which need to be judged by left-corner analysis method. The document structure reconstruct method starts with the root node in the syntax tree, and then expands the non-leaf node along the left side of the tree until reached at the bottom and left non-leaf node, the bottom and left non-leaf node is current state node which need to be judged. And if the current input component is the left corner of the current state node, the algorithm continues to judge the substructure of the current state node, the substructure is judged over once all the leaf nodes of the substructure are equals to the input, and we use the xml to store the result of substructure; but if the current input component is not the left corner, the algorithm will go to judge the brother node of the current state node. The algorithm about how to judge non-leaf node state is presented in Algorithm1.
Algorithm 1 State node structure judgment algorithm based on left angle analysis method

1: The bottom and left non-leaf node of the state node: state1
2: Put state1 into stack
3: Construct a xml whose root is state1
4: if(state1 is the parent of the current input component) then
5: Add the current input component into xml;
6: While(stack is not empty) do
7: if(the top of stack is \( \alpha \)) then
8: Put the next input component into stack
9: else if(The two elements top of the stack are \( \alpha\alpha \)) then
10: Remove \( \alpha\alpha \), put \( \alpha \) into xml
11: else if(The top of stack is \( \alpha\), \( B \rightarrow \alpha\gamma \)) then
12: Use \( \gamma B \) replace \( \alpha\), add \( \alpha \) into xml
13: if(state1 has brother node) then
14: Recursive call algorithm1 to judge next brother node
15: else
16: return xml

The algorithm1 works in two ways: search and lookup. The search category is represented by labels which add a horizontal line. we use \( \alpha \) to represent a single terminator or a non-terminator, and use \( \gamma \) to represent a string of terminators or non-terminators. Firstly, we start with the root node, and then expand the non-leaf node along the left side of the tree until the bottom and the left non-leaf node, then push the node into the stack if the input component is the left corner of current node, and construct a xml file whose root node is current node and child node is current input component, and then look up the element in the stack peek, if the element is \( \alpha \), push next component into stack, but if the element is \( \alpha\alpha \), we reduced the \( \alpha\alpha \) and add \( \alpha \) to the xml file, if the element is \( \alpha \), we construct a rule whose left corner is \( \alpha \) and then add \( \alpha \) to the child node of xml file. Then continue search until the stack is empty.

<table>
<thead>
<tr>
<th>step</th>
<th>stack</th>
<th>set of components</th>
<th>operate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>header</td>
<td>[abstract, text, keyword]</td>
<td>Put the search category ( \text{header} ) into stack; Construct a xml whose root node is header</td>
</tr>
<tr>
<td>2</td>
<td>header abstract</td>
<td>[text, keyword]</td>
<td>Put the abstract into stack</td>
</tr>
<tr>
<td>3</td>
<td>header header keyword text</td>
<td>[text, keyword]</td>
<td>Abstract is the left-corner of header, put the syntax tree rule into stack, add the abstract into xml</td>
</tr>
<tr>
<td>4</td>
<td>header header text text</td>
<td>[keyword]</td>
<td>Put the next component keyword into stack</td>
</tr>
<tr>
<td>5</td>
<td>header header keyword</td>
<td>[keyword]</td>
<td>Reduce ( \text{text text} ), add text into xml.</td>
</tr>
<tr>
<td>6</td>
<td>header header keyword text</td>
<td>[]</td>
<td>Put the keyword into stack</td>
</tr>
<tr>
<td>7</td>
<td>header header</td>
<td>[]</td>
<td>Reduce ( \text{keyword} ), add keyword into xml</td>
</tr>
<tr>
<td>8</td>
<td>[]</td>
<td>[]</td>
<td>Reduce ( \text{header header} ), return xml</td>
</tr>
</tbody>
</table>

Table 1. The analysis process of header.
For example, the set of logical components is \{abstract, text, keyword\}, the set of state node is 
\{article, header, first-level chapter, second-level chapter, third-level chapter\}. First of all, the root node article is judged. The bottom and left non-leaf node of the article node is header, the current input component abstract is the left-corner of header, the table 1 shows the analysis process of header.

**Fault Tolerance Processing based on Error Correction Rules**

**Construction of Error Correction Rules**

The analysis process of document structure can not be carried out once the components set can not satisfy the typesetting rules. This paper finds common errors in the process of analysis document structure re-construct. We formulate the following rules to correct the common errors.

[Rule1] In the analysis process of document reconstruct, if the current judgment state node is the grandfather of current input component, the following algorithm is used for error handling.

Algorithm 2. the relationship between state node and current input component is grandfather relationship

1: if(current input component is the parent node of next input component) then
2:     Change the location of current input component and next input component
3: else
4:    Count the same number of current logical component level until different level component
5:      if (count>1) then
6:         Add the upper level component before current input component
7:      else
8:         Set the level of current input component to the upper level

[Rule2] In the analysis process of document reconstruct, if the currently input component is not the first child of current state, we will use the following algorithm to deal with it.

Algorithm 3. the current input component is not the first child of current state node

1: if(the next input component is the first child of current state node) then
2:     Change the location of next input component and current input component
3: else
4:    Add logical component which match the first child node of state node

[Rule 3] The reconstruct result is saved as xml, but if the xml file failed to validate when using schema validates the xml, we use the following method to deal with it.

The schema restricts the chapter occurrences number, the xml structure is incomplete if the xml validates fail, and then return the name of the missing chapter. We read the heading $T$ of the chapter in xml, and find its superior heading $S$, and make a correction by comparing the similarity between the two heading. This paper selects the edit distance method to calculate the similarity. Suppose the length of $S$ and $T$ is $m$ and $n$, the matrix $D_{ij}$ indicates the edit distance that the logical component $S$ is transformed into component $T$ whose length is $j$. The definition of the matrix is shown in formula (1).

$$D_{ij} = \begin{cases} 
  i & j = 0 \\
  j & i = 0 \\
  \min(D_{i-1,j-1}, D_{i-1,j}, D_{i,j-1}) + a_{ij} & i, j > 0 
\end{cases}$$

(1)

$$a_{ij} = \begin{cases} 
  0 & S_i = T_j \ (0 \leq i \leq m, 0 \leq j \leq n) \\
  1 & S_i \neq T_j \end{cases}$$

(2)

Finally, with $|S|$ represents the string length of $S$, $|T|$ represents the string length of $T$, we can now calculate the value of the edit distance by formula (3).
\[ \text{sim}(S, T) = 1 - \frac{D(S, T)}{\max(|S|, |T|)} \]  

(3)

If the value of sim(S, T) is 1, it means that the string of two logical components are exactly the same, So we think the logical component T is redundant; Otherwise, comparing the similarity of logical component T and next component title K, if the value of sim(T, K) is greater than 0.5, we think the K is level error and should be the same level as the T, if the value is less than 0.5, we consider T is a redundant logical component.

**Fault Tolerant Mechanism**

The fault tolerant mechanism of document structure reconstruction combines fault tolerance rule with the left corner analysis method. In the process of using the left corner analysis, we can judge whether the component is error or not by analyzing the relationship between the state node and the current input component, if the current input component is the left corner of current state node, the algorithm will analyze the document structure using the third section method. Otherwise, we think the component is wrong, and using the algorithm4 to correct the error.

**Algorithm 4. Structural reconstruct with error correction rules**

1: if (current state node is the ancestor of current input component) then  
2: if (current state node is the parent of current input component) then  
3: if (current input component is the first child of current state node) then  
4: \begin{center} Reconstruct document structure with algorithm 1 \end{center}  
5: else  
6: Correct it with rule2  
7: else  
8: Correct it with rule1

The algorithm4 uses rule1 and rule2 to correct the common errors, after error correction, the left corner analysis is used to identify the document structure until the sub-structure of the state node is analyzed. At the same time, in order to ensure the accuracy of each state node structure recognition, the substructure recognition results of each state node are stored in the form of xml, and then using schema to validate the xml, we use rule3 to correct the error once validates fail.

**Experiment**

We evaluated the fault-tolerant effectiveness of our approaches using scientific articles from Beijing Information Science and Technology University. There are totally 50 samples which include 108 errors, every sample has one or more errors, we use our approaches to correct these errors, totally 96 errors are corrected, and the real errors is 89, we get 92.71% precision score, and get 82.41% recall score. The Figure2 shows the result of common errors.

As show in Figure2, our method has high-level accurate in dealing with common errors in documents. As for heading lack error and the location between figure and caption error, the previous document structure re-construct method like directed graph[2] and finite state machine[3] cannot handle these errors, because the previous method can not find the matching rule, as for the heading redundancy error will cause the wrong structure which not meet the author writing intention, but our method can handle it accurately by fault-tolerant rules.

We evaluated the document structure re-construct effectiveness of our approaches using 300 scientific articles from Beijing Information Science and Technology University. And we compare our method with based on directed graph[2] and finite state machine[3], the method based finite state machine gets a 94.22% precision, the method based directed graph gets a 97.38% precision. Our method is better than both of them which gets a 98.86% precision, our method can solve the problem that the previous structure reconstruct method cannot be reconstructed when the document has errors.
Summary

In this paper, we presented a document structure reconstruction method that combines with error correction rules and syntax analysis. In the analysis process, first, the analysis efficiency is effectively improved. Secondly, the method can generate a structure that meets the author's intention as much as possible by adding error correction rules. It can solve the problem that the previous structure reconstruct method cannot be reconstructed when the document has errors. Thirdly, this method is applicable to different types of documents. Experiments show that our method can handle common errors, and solve the shortcomings of fault tolerance in previous methods. However, there are also some problems in our method. In the future, we can consider increasing the semantic analysis of text content to analyze uncommon errors. For future work, it is worthy to analyze document structure with semantic information.

Acknowledgement

This research was supported by the National High Technology Research and Development Program (863 Program) of China (2015AA015403), National Natural Science Foundation of China: The Intelligent Analysis and Optimization Method for Re-flowable Documents (61672105);

References


