BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

WEB PAGE SEGMENTATION UTILIZING CLUSTERING TECHNIQUES

SEGMENTACE WEBOVÝCH STRÁNEK S VYUŽITÍM SHLUKOVACÍCH TECHNIK

EXTENDED ABSTRACT OF PHD THESIS
TEZE K DISERTAČNÍ PRÁCI

AUTHOR **Ing. JAN ZELENÝ**
AUTOR PRÁCE

SUPERVISOR **Doc. Ing. JAROSLAV ZENDULKA, CSc.**
ŠKOLITEL

BRNO 2017

Abstract

Information extraction and other techniques for mining data from the Web get more important with the development of web technologies and raising amount of information stored exclusively on the Web. However, with this information, the amount of content that is completely irrelevant in context of the presented information grows as well. That's only one of the reasons why it is so important to intensively study and develop preprocessing of information stored on the Web. Segmentation algorithms are one of the possible ways of web page preprocessing. This thesis is dedicated to utilization of clustering techniques for improving the efficiency of existing web page segmentation algorithms, as well as finding completely new ones.

Keywords

segmentation, clustering, template, VIPS

Reference

ZELENÝ, Jan. *Web page segmentation utilizing clustering techniques.* Brno, 2017. Extended abstract of PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Zendulka Jaroslav.

Contents

1 Introduction **2**

2 Web Page Preprocessing: State of the Art **3**
2.1 Page Segmentation 3
2.2 Template detection 7

3 Motivation and Goals of the Thesis **9**

4 Web Site Processing Using Clustering Techniques **10**
4.1 High level design 11

5 Box Clustering Segmentation **12**
5.1 Extracting boxes 12
5.2 Connecting the boxes 13
5.3 Composite Dissimilarity Model 13
5.4 Base Dissimilarity 15
5.5 Cluster Dissimilarity 17
5.6 Box Clustering 17

6 Template clustering **18**
6.1 Template clustering overview 19
6.2 Template storage and selection 19
6.3 DOM Tree Mapping 20

7 Evaluation **20**
7.1 Template Count 21
7.2 Evaluation of the BCS 21
7.3 Evaluation of BCS with template clustering 23

8 Conclusion **23**
8.1 Summary of Contributions 24

1 Introduction

In recent years, the World Wide Web has seen such a great expansion, it has become the most important source of information in the world. The number of algorithms designed to process the data stored on the World Wide Web and give it some meaning grows in proportion to the growth of the Web itself.

The largest group of the Web processing tasks falls into the area of data mining. This includes the tasks like information retrieval, content extraction, classification and others. In that context, it is important to realize that each web page is a combination of useful information, navigation, related topics, comments, advertisement and many more disruptive elements. Moreover, the useful information on the web can be disseminated, there can even be more different pieces of information (or topics) on one web page.

To address these issues, new methods of preprocessing web pages have to be developed to make the data processing algorithms themselves successful. These preprocessing methods often come in some form of web page partitioning into separate areas. Each of these areas will then carry a content somehow different from the content of other areas. The differentiation parameters depend on the intended usage of the preprocessed data. In case of the data mining tasks, the partitioning is often complemented by classification with the intended result being identification of the blocks that are either relevant in the context of the web page or that are important for the subsequent algorithm. In case of restructuring the information for mobile devices, the subsequent goal is to rearrange the web page or even remove some blocks that are not considered important.

Template detection methods are a good example of those that perform some form of web page partitioning. While they are usually perceived only as remotely related to this area, the definition of page partitioning fits what they do – they separate a page to useful content and the noisy remainder around it. In contrast, web page segmentation methods target specifically the task of web page partitioning. They output a set of different blocks, each block internally consistent. The consistency is usually defined either semantically or visually.

From the perspective of human intervention, there are supervised, semi-supervised and unsupervised segmentation methods. This work will focus on the area of unsupervised web page segmentation. Even though the area has been extensively researched, there are only a few ways how is the partitioning task approached. There are two aspects of the web page segmentation that are considered important – accuracy and speed. The existing segmentation methods emphasize one, keeping the other within the range of what is considered acceptable.

Algorithms in the group of vision-based segmentation try to perceive the page as user viewing it would perceive it and perform the segmentation based on visual cues. These methods therefore strongly prefer accuracy, with performance being only secondary. Their performance disadvantage is implied by the fact that the visual cues need to be calculated according to very complex specification.

This thesis proposes a new way how to segment web pages, using the aforementioned visual cues and clustering techniques. The motivation is to come up with such an algorithm,

that will be superior to all the existing algorithms in terms of speed-accuracy balance. The proposed method is built from ground up. While it does not share almost anything with current segmentation algorithms, some of its parts are inspired by template detection algorithms which are remotely related to segmentation algorithms.

Another important difference of the algorithm proposed here and all the others that can be found in the literature is its high level design. Compared to others, the algorithm described here can be split into two parts, each one working on a different level and addressing a different use case. While the first part addresses segmentation of individual web pages, the second part takes care of the use case where multiple pages are processed over any period of time. Taking care of this particular use case significantly improves scaling of the algorithm. Note that clustering techniques are utilized in both parts of the algorithm.

2 Web Page Preprocessing: State of the Art

First, it is important to define what should be understood by web page preprocessing. There are many techniques how to preprocess the web page. Their goals and results strongly depend on the main tasks that the preprocessing is preparing data for. The most common task class is data mining operations, where the goals include simple cleanup of the noisy and irrelevant information and detection of smaller and internally consistent areas where the internal consistency can be defined either visually or logically.

This chapter will introduce some preprocessing algorithms that can be found in the literature. While *page segmentation algorithms* primarily compete with the algorithm proposed in this thesis, *template detection algorithms* are here mostly because they introduce general mindset and some concrete concepts that are utilized in the design described in the following chapters.

2.1 Page Segmentation

The web page segmentation can be defined as a process of identification of coherent or significant parts within the page. There are two basic groups of methods how to perform web page segmentation – code-based and vision-based methods. In general, code-based methods are much faster but quality of their results highly depends on heuristics they use and type of page they inspect (some heuristics perform well only on some types of pages, e.g. news articles). Because they don't need the page to be rendered, they rely on inspecting either the HTML code or (more often) the DOM tree which is what makes them fast and scalable.

Vision-based methods on the other hand have greater potential for accuracy[1]. This potential comes from the fact that large portion of information from the web page, such as computed CSS styles, is not evaluated by code-based methods. The approach of vision-based methods is to render the page first and then utilize the calculated visual information. That means these methods try to view the page as user would.

[1]The accuracy is usually defined as a consistency between the result and human perception of the page.

2.1.1 Code-based Segmentation

As said before, code-based segmentation methods are further divided based on the type of heuristics that are used. In this thesis, the distinction is based on representation of the web page that is being inspected. *Text-based methods*[23, 11, 7, 18] inspect textual representation of the web page, whereas *DOM-based methods* first transform the textual representation to a DOM tree which is then inspected.

Text-based methods

Text based methods utilize features of the text content of the web page and build heuristics on them. An example of such is a method called NCE, proposed by Laber et al. in [23]. NCE is an algorithm intended to identify and extract the relevant content (i.e. the article) from news web pages. Finding the location of the article on the web page is based on density of links in the text. The link density is practically a measure specifying how much non-navigational text is contained in a given subtree. The core assumption for locating an article is that in every page's DOM tree, there is a node with very low portion of links in contained content and very high F-score calculated using set of words in the corresponding subtree, corpus of all words in the entire text and a set of relevant words, as defined by human observer. The algorithm itself just performs depth-first-search traversal of the DOM tree and tries to find a node N which fits the condition of high F-score.

DOM-based methods

Another group of code-based segmentation algorithms performs general inspection of the DOM tree and segments the content using various heuristics[15, 32, 17]. Some related information extraction [34] and document cleaning [35, 41] approaches share the same concept too.

There are three main categories of features that can be used for heuristic:

- *Markup-based features* — perhaps the most extensive category. It's possible to analyze parents of the node, the node itself and its child nodes. Some special nodes like `<strong>` and `<em>` are usually interesting, as they designate higher importance of the contained text and child nodes. Another possibility is to investigate line breaking elements (usually used for detection of content separator).
- *Content-based features* — these represent features gathered from analyzing characters, words and sentences.
- *Document-related features* — features in this category include position in the document or a content-based features related to the context of the entire document.

2.1.2 Vision-based Segmentation

Vision-based methods are all those that use visual features of elements in a web page to determine the partitioning. In the literature, the dominating method is the first one described here: VIPS. It is the pioneering method of this approach which has not been unambiguously surpassed by any other algorithm. That's also where this group of methods got its name.

VIPS

A pioneering work in vision-based page segmentation was laid down by Cai et al. in [9]. It's worth noting that while the algorithm is the first one that uses visual information contained in the web page, the originally described implementation is not entirely vision-based when strictly following the description at the beginning of section 2.1, as it does not use the rendered web page, it only uses the information stored within the DOM tree.

The algorithm is built on the concept of *Degree of Coherence* (or simply *DoC*). It is a metric represented by integer, defined for each block. As the name suggests, it is directly proportionate to the internal visual consistency of that block. Intuitively, parent node cannot have greater DoC than any of its children. A threshold value affecting granularity of the result is related to DoC. It's called Permitted Degree of Coherence or simply PDoC.

The VIPS algorithm processes the web page in several steps. The first step is an extraction of visual blocks. It consists of a top-down tree traversal through the DOM tree and inspection of visual cues. The traversal is iterative – in each iteration a new node representing visual block is detected in the DOM tree. After this detection a decision is made whether the block shall be recursively segmented further or not.

The second step is separator detection. *Separator* is defined as horizontal or vertical line or rather rectangular area which does not intersect any of previously detected blocks. The algorithm is initialized by a single separator covering the whole page. Then, a detection of relations between existing blocks and separators is performed and appropriate actions are taken based on the detected relations. Figures 2.1, 2.2 and 2.3 outline what the relations can be and what action is taken for each relation.

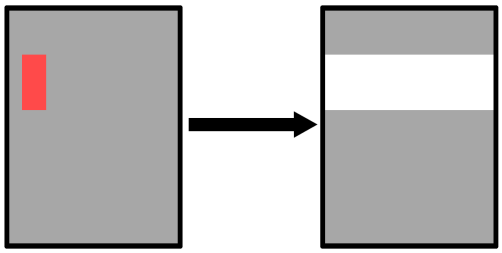

Figure 2.1: Divide the separator

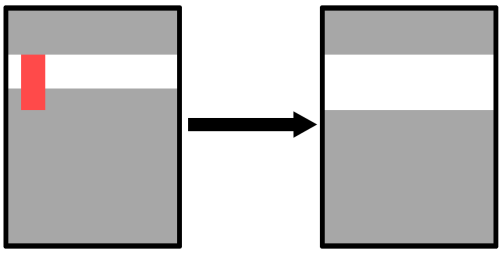

Figure 2.2: Shrink the separator

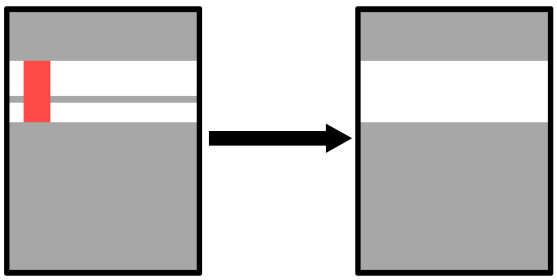

Figure 2.3: Delete the separator

The algorithm is finished by removing separators at the edges of document. Once VIPS has all separators, it assigns the weights based on visual difference of adjacent blocks. Note

that on each level of the block tree this algorithm produces either only vertical or only horizontal separators, but the direction is changing in each level of the block tree.

The final step of VIPS is content structure construction. In this step the algorithm iterates through a list of previously found separators and merge visual blocks adjacent to them. It's important to merge blocks adjacent to separators with the smallest weight first. The merging continues until all blocks meet the granularity requirement $DoC >= PDoC$.

Besides the shortcoming of not fully rendering the web page, there are a few others:

- In some cases direct division of a visual block is impossible and utilization of virtual blocks is required. This can have negative impact on further processing, because blocks are not really present in the document.
- Resulting tree represents page segmentation but some information such as mutual position of blocks is missing. That information might be useful for results refinement or by some subsequent algorithms.

Even though VIPS is the oldest algorithm and has its shortcomings, it has not been unequivocally superseded in both accuracy and processing time. There has been a lot of work in the area that improved its accuracy [25, 24, 45, 39, 40] and adapted it to modern standards in HTML[2, 1] but that work has been derived from the original approach and still uses the original VIPS as a core. Some alternative algorithms that rival VIPS exist as well, those will be covered in the following sections.

Web Content Clustering

One method entirely independent on VIPS was presented by Alcic et al. in [5]. While the paper does not offer any particular design, it brings up and analyzes the idea of segmenting the web page using clustering algorithms. The concept is based on the observation that each web page is just a set of pieces of the content and that these pieces can be grouped together in such a way that each group represents one topic. The underlying assumption is that each such group can be created by clustering the right content pieces. The paper offers validation of that assumption by presenting an overview of both parts necessary – clustering algorithms and metrics.

The first part of the paper covers the metrics that can be used. A few very basic metrics are explored individually:

- mutual position of nodes within the DOM tree of the web page (DOM-based distance)
- semantic distance of the content using WordNet-based generalization of concepts and TF-IDF[2]
- mutual position of the nodes on the rendered web page (vision-based distance)

Three clustering algorithms are compared in the second part of the paper: partitioning algorithms represented by k-means and k-medoids, hierarchical agglomerative clustering and density-based clustering represented by DBSCAN. Of these three the DBSCAN was proved to be the most convenient.

The most interesting conclusion of the paper was that using the vision-based distance didn't bring results that would be significantly better than the other two metrics, no matter which clustering algorithm was used. Instead, the DBSCAN in combination with the DOM-based distance showed the best average statistical results.

[2]Term Frequency-Inverse Document Frequency is distance metrics used in plain text data mining

Other Vision-based Segmentation Methods

In the latest research, authors often avoid using the HTML-based heuristics by using alternative ways of block detections. Liu et al. [26] constructs a graph of spatial relationships among the rendered elements that is later partitioned with a Gomory-Hu clustering algorithm and Zeng et al. [43] compute a seam degree and content similarity of the individual blocks in order to divide larger visual blocks to smaller parts. Finally, Xu et al. [42] applies the Gestalt laws of grouping on the extracted visual blocks.

Other vision-based algorithms use entirely graphical representation of the input document that allows to abstract from the HTML-related implementation details. Cormier et al. [10] uses an edge detection algorithm for detecting the visual separators between the content blocks. Similarly, Wei et al. [38] uses Hough transform for the same purpose and Kong et. al [19] recognize atomic objects using image processing methods and perform their grouping by using a spatial graph grammar. And in his work, Burget [8] proposes method that is remotely similar to VIPS but addresses its flaws.

2.1.3 Hybrid Segmentation

The hybrid approaches combine the DOM-based and vision-based ones in order to obtain higher segmentation accuracy or for specific applications. Sanoja et al. [31] and Manabe et al. [28] both combine the content structure (DOM) with the visual information obtained from a web browser in order to increase the accuracy in comparison to the VIPS algorithm. Safi et al. [30] process the input document in two steps: first, a visual information analysis is performed and in the second step, DOM tree filtering is performed based on the analysis results with the aim of supporting visually impaired users. Finally, Fumarola et al. [12] combine the DOM with a visual information model in order to extract visually presented lists from the input documents. More generic content extraction use case is presented by Song et al. [33]

2.2 Template detection

The goal of template detection methods such as [27, 22, 21, 13, 3] is to find and filter redundant information (noise) on the web page[6]. Because they do not try to separate individual segments on the page, they are generally faster than segmentation methods. The only expception are those template detection algorithms that detect templates using segments[4, 20] instead of DOM nodes. However, that group is not interesting for the purpose of this work and therefore will not be considered further. This paper will inspect the means these algorithms use to provide the result quickly. It will be later demonstrated that some of the features of template detection algorithms can be used to improve the performance of segmentation algorithms in certain use cases.

Besides operating on DOM trees rather than render trees, another key factor that makes these methods fast is the fact that they process multiple pages at once. For this approach to work, the pages that are processed at once have to be based on the same template. The detection of whether or not are two pages based on the same template is the part that will be utilized to boost the page segmentation. Only selected methods that support this use case will be presented below – the methods working with *tree edit distance*.

In template detection, the tree edit distance is used for evaluation of structural differences between DOM trees. In particular, algorithms working with the tree edit distance can tell

how similar are two given trees (i.e. to determine if they belong to the same template) or which parts do the two trees share (i.e. to identify the template itself).

The tree edit distance comparison uses three basic operations with nodes in the tree: insert, replace and remove. Each operation type can have its cost defined but usually they all have the same value. The problem of tree edit distance can be defined as finding a mapping from tree A to tree B with minimal cost. The mapping has three important rules: (1) only one operation can be performed on each node, (2) the order between siblings has to be preserved and (3) the ancestor-successor relation between any two nodes has to be preserved.

The generic tree edit distance problem on unordered trees is proved to be NP-complete [44] and algorithms solving generic tree mapping weren't much efficient. In the practical application, however, it is possible to define restricted formulation. By imposing some restrictions (e.g. [36]), four new definitions of the problem are formulated and fast algorithms utilizing them are proposed: *alignment*, *distance between isolated trees*, *top-down* and *bottom-up*. The latter two are used in template detection and the last one specifically will be briefly described further.

2.2.1 Traditional, bottom-up Tree Mapping

One of the best algorithms in the literature[37] is Valiente's bottom-up approach [36] – it has just a linear complexity. As its designation says, it is based on a tree traversal from its bottom. The mapping M between trees T_1 and T_2 is bottom-up if for every couple $(t_1, t_2) \in M$, where t_1 and t_2 are nodes of T_1 and T_2 respectively, the following condition is valid: $\forall c_i \in children(t_1) : \forall c_j \in children(t_2) : (c_i, c_j) \in M$. Less formally it is possible to say that a node can be mapped only if all of its children are mapped as well.

2.2.2 Tree Paths Approach

While the similarity of two pages is described in the previous sections as a mapping distance between the DOM trees, Gottron suggests[14] other distance measuring methods which he proves to be significantly faster than traditional mapping approach. His work can be considered somewhere between page segmentation and template detection, though it is closer to the latter one. For segmenting a page he retrieves other pages through hyperlinks on the first page. On these retrieved pages he performs template-detection and then uses the template to segment the original page. On the beginning of the process every text node is considered to be a page segment. For each such segment its frequency of appearing in different pages is counted. If this frequency is high, that very likely means that it is a part of the template. Basically the algorithm evaluates informational gain (Entropy) for each block and it discards those with low gain.

Gottron describes three methods to calculate structural distance between two pages: *Common paths distance*, *Common path shingles distance* and *Common tag sequence shingles distance*.

Using the first one, the distance is then measured as a size of intersection of sets S_1 and S_2 containing all paths from trees T_1 and T_2 respectively. The second method improves precision of the first one. It considers splitting paths to a smaller pieces, called *path shingles*. The advantage of this approach is that the first algorithm would detect two paths non-equal even with one node differentiating whereas this algorithm would only evaluate the particular shingle as non-equal and the rest would be included in the intersection. The third method solves the computational complexity of comparing two sequences of tags each representing

the entire web page. Instead, the sequences are again split into shingles and shingles are then compared to each other with much less demand on computing power.

2.2.3 Template Detection Algorithms as Clustering Metrics

When inspecting template detection algorithms, one prevailing feature emerges. It has been said in the previous text that the same methods that can identify common parts on the web pages can be used to determine how similar two web pages i and j are. In fact, these algorithms meet all the conditions that distance metrics for clustering algorithms have to meet:

$$d(i,j) \geq 0 \tag{2.1}$$
$$d(i,i) = 0 \tag{2.2}$$
$$d(i,j) = d(j,i) \tag{2.3}$$
$$d(i,j) \leq d(i,h) + d(h,j) \tag{2.4}$$

That effectively means it is possible to write a clustering algorithm that would group together web pages for further processing that are based on the same template. From all the algorithms listed above, the traditional tree-mapping algorithms might seem to be the best option for this task, as they tend to be as precise as possible in practical application. However as Gottron demonstrated in [14], the tree path method is comparably precise while being much more efficient, even in the most simple variant. That's why it's the best candidate for utilization in this thesis.

---

3 Motivation and Goals of the Thesis

Various methods for preparing web pages for data mining procedures have been introduced in the previous chapter. These methods approach the problem of web page preprocessing from several completely different directions, focusing on different problems which can be encountered during different data mining tasks.

In the initial phases of the research it became clear that the concept of page segments being atomic carriers of information on the Web is becoming increasingly important. However, while there has been some effort in the area of web page segmentation, the vision-based page segmentation has not progressed much further since its discovery. Some algorithms were created, but most of them are built on top of the VIPS algorithm. Many of them propose just some heuristics to improve its results, but the fundamental design remains the same. The research in this area has been stagnant most likely because of large time complexity of visual segmentation. Especially in the area of the Web, the time complexity is very important considering the need for algorithm application on a large scale. The problem is that this time complexity is not balanced out by equally significant gain in accuracy.

As a consequence of lack of development in recent years, vision-based page segmentation algorithms often don't fully take modern technologies like HTML5 and CSS3 into account. These technologies make the web pages more dynamic and, when rendered, often very

different from the DOM tree. This thesis focuses on solving both mentioned areas. Its goal is to design a new vision-based segmentation method, which:

1. Will consider new technologies used in the process of development of web pages
2. Will be optimized from the perspective of time complexity

The task of designing a new vision-based segmentation method can be divided into four parts, each one being an important partial goal that needs to be fulfilled in order to consider the design acceptable.

1. Formal specification of the problem
2. Design of new segmentation method and its formal specification
3. Design of data structures and algorithm optimization with focus on time complexity
4. Experimental evaluation of the new method's properties

4 Web Site Processing Using Clustering Techniques

In this thesis, the segmentation is expected to be only the first step in the web page preprocessing. In other words, some subsequent treatment (e.g. classification) is expected to take place in order to finish the process pages go through.
Single-page segmentation: this is very generic use case. It covers virtually any scenario where segmentation can be used. In context of this thesis it is rather marginal but it is included to demonstrate the quality of the segmentation core that is described in this thesis. URL address of the page that is to be segmented is the only input in this use case. The scenario is very simple:

1. A web page and all linked resources are fetched from the Internet
2. The code of downloaded resources is transformed to a corresponding DOM tree
3. The DOM tree is transformed to a structured representation of rendered page
4. Segmentation is performed on the structured representation of rendered page
5. A set of visual areas is presented on the output of the segmentation algorithm that can be used for further processing

Multi-page segmentation: this is an extension of the previous use case. This one is based on an assumption that in practical applications (e.g. automated environment) users will often want to extract information from more than just one web page. While the actual implementation may differ, the proof of concept designed in this thesis is just a simple breadth-first crawler mechanism. The use case can be described as follows:

1. A segmentation of one page is performed as described above

2. All hyperlinks are extracted from the DOM tree of the page and put at the end of a to-be-inspected queue

3. If there is a limit of the number of pages to inspect and it is reached, the algorithm ends

4. The first URL from the queue is taken and if it has not been inspected before, the segmentation starts from point 1.

4.1 High level design

By its design, the system is intuitively adapted to the two use cases described above. Graphical representation of the entire system is displayed in figure 4.1. The diagram shows high level view of the individual components and the data as they flow between them. The components with doubled side lines are the key parts of the system that will be closely described in the following chapters.

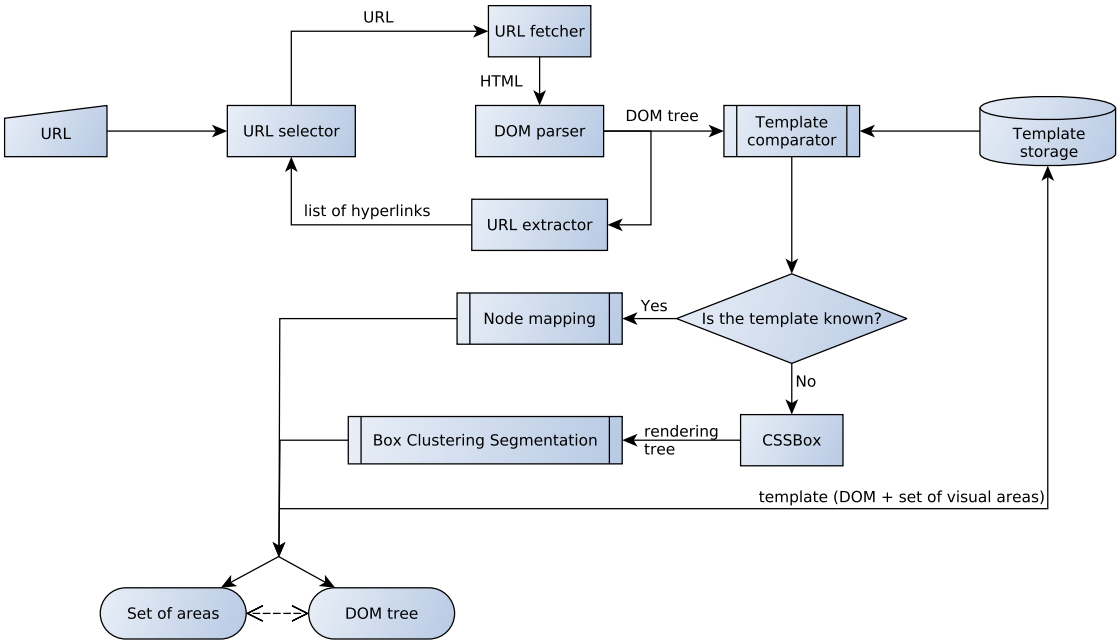

Figure 4.1: Architecture of the proposed segmentation system

Now a few facts about the rest of the components in their simplest design: *URL selector* uses just a simple queue initiated with the URL specified by user. The selector always selects the first URL in the queue. *URL fetcher* and *DOM parser* respectively produce HTML and DOM tree of the web page. *CSSBox* is a rendering engine used as black box. Any other rendering engine can be used as long as it provides access to the required data structures. Any external database can be used for Template Storage.

5 Box Clustering Segmentation

The Box Clustering Segmentation (BCS) is the only component that influences precision of the result and it has great effect on the performance as well. BCS is designed to be a pure vision-based method designed to give flat hierarchy of results. The algorithm takes rendering tree produced by CSSBox as an input. This rendering tree contains all the information about visual features of the web page but it also contains all the information that was stored in the original DOM tree. BCS can be decomposed into these steps:

1. *Box extraction* identifies the smallest visual elements on the page.
2. *Distance calculation* between individual elements using criteria described in section 5.4.
3. *Graph creation* is the initial iteration of the clustering algorithm.
4. *Clustering* the boxes iteratively into segments.

A set of clusters (i.e. segments) and boxes that didn't fit to any cluster are returned as output. Note that in further text, the distance metrics is also called *dissimilarity*, as the distance between two elements on the web page is determined by their visual dissimilarity. After the dissimilarity is calculated, it is used to create and then iteratively extend clusters.

5.1 Extracting boxes

The box extraction is the first step of the Box Clustering Segmentation. Even though it can be perceived just as a preprocessing, it is an important part of the BCS algorithm, as it influences both precision and performance.

The box extraction algorithm traverses the rendering tree and selects boxes that should be used for further processing. In the following text, the designation *box* will be used for simplified data structures describing the atomic units of the content. Each box contains just minimal amount of information:

Definition 1 (Box structure) *Let the box m be defined as 7-tuple $m = (left, right, top, bottom, width, height, color)$ where $left$, $right$, top and bottom are integer values that represent positions of the respective edges of the box; $width = right - left$; $height = bottom - top$ and color is a composite value representing mean color of the box.*

The boxes are extracted from the rendering tree using a recursive algorithm performing a pre-order traversal of the rendering tree. The general idea is to consider the nodes of the tree that are actually visually accessible in the page, i.e. the user looking at the page can see them. This refers to the fact that many tricks are used on modern web pages to achieve the intended layout and some boxes on the web page might not actually be visible. Most of the visible content consists of the leaf nodes in the rendering tree but various exceptions

are considered as well, such as subtrees with no branches where another node can represent the content.

The output of the box extraction algorithm after the filtering step can be described as a set of boxes where no two boxes overlap. Unless specified otherwise, the rest of chapter 5 assumes that boxes that are being worked with are non-overlapping.

5.1.1 Box color

While other values are straightforward to get, the box color has to be calculated during the extraction process. The goal is to find that color of each box that is the most visually significant in it. There are three types of boxes from the perspective of what they represent: lines of text, images and other content (for example boxes with no content, their purpose being only visual on the web page).

Boxes with no content have the most simple color representation, as their background color is picked (or the background color of their parent in the rendering tree if they are transparent). For every image the average color is calculated. All the pixels in the image are iterated over and the mean values of the red, blue and green components put together the average value. For text, the situation is much more complex. The representative color of the text box takes properties such as background color, font weight, slant and text decorations into account.

5.2 Connecting the boxes

The previous step produces a set of boxes which are atomic elements of visual content of the web page. The goal of connection-creating algorithm is to iterate over this set and for each box to identify other boxes that will be preferentially evaluated for clustering with that box.

This part of the segmentation process is crucial for performance of the entire algorithm, as the number of connections between boxes strongly influences the computational time of the clustering step. A naive approach would be to create a connection between every two boxes. However, with thousands of boxes being on a web page (which is not uncommon), the size of the box set increases so much it raises time and space demands of the algorithm beyond practical usability. It is therefore desirable to optimize the number of connections as much as possible.

The chosen optimization of the connection-creating algorithm is to connect only boxes that are a) adjoined and b) semi-aligned. There are two reasons why this representation was chosen:

- It's easier to extract directly adjoined boxes during subsequent processing.
- Experimental observation showed that visually related boxes are always organized like this

Figure 5.1 graphically depicts semi-alignment of boxes. The dark blue box is being inspected, the light blue boxes are semi-aligned with it while the red boxes are not.

5.3 Composite Dissimilarity Model

A proper model for evaluating the dissimilarity between the elements on the web page, that will be described more closely in this section, is one of the main pillars of any segmentation

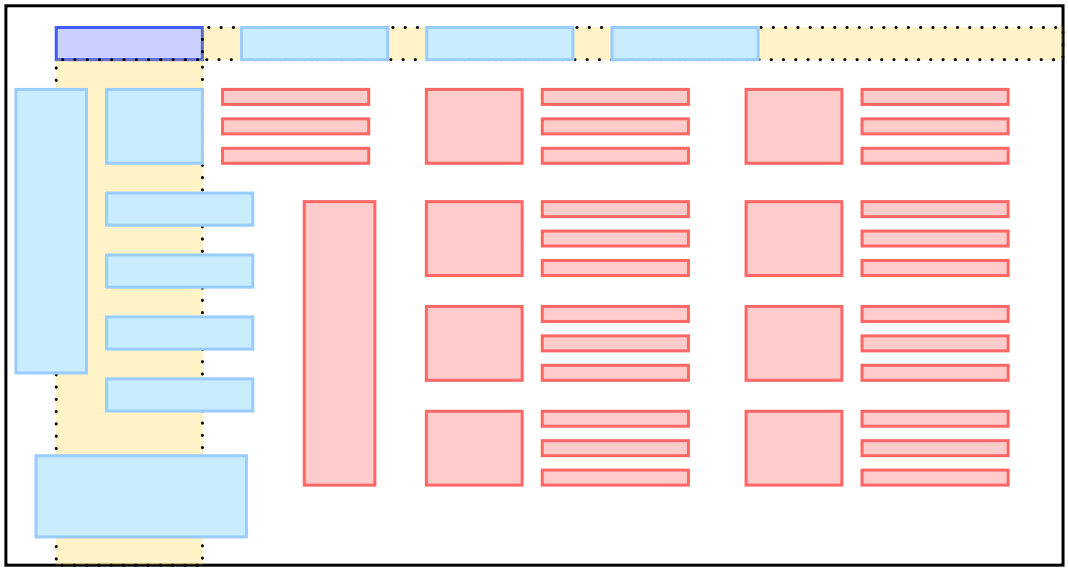

Figure 5.1: Boxes that are selected to be connected

algorithm. There are two types of entities featuring in the dissimilarity model – *boxes* and their *clusters*. The relation between these two types is denoted in figure 5.2.

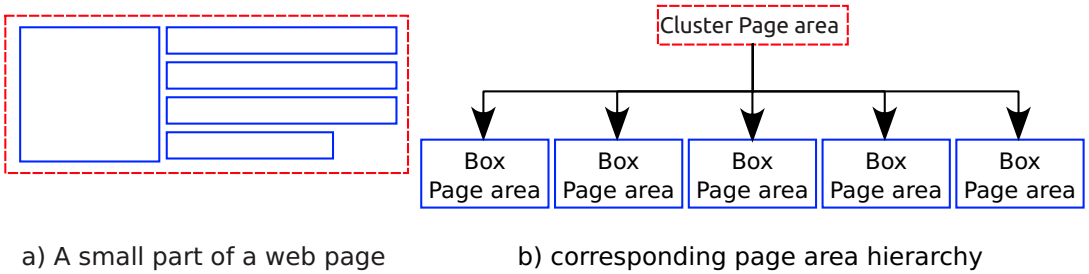

Figure 5.2: Hierarchy of visual areas representing boxes and their clusters

The model is called composite because it contains two main dissimilarity components, based on the type of entities between which the dissimilarity value is calculated. The first compound, called *base dissimilarity*, is based on visual features of individual boxes and is therefore defined only between two boxes. On the other hand, the second one, called *cluster dissimilarity*, is used to express the dissimilarity when at least one of the two entities is a cluster.

Generally, the value of dissimilarity and/or its compounds is floating-point number in the range of $< 0, 1 >$ with straight proportionality semantics, i.e. the higher the value gets, the higher the dissimilarity is.

5.4 Base Dissimilarity

Base dissimilarity can be theoretically calculated between any two boxes on a web page. But to follow the optimization rule established in section 5.2, it is calculated only between those pairs of boxes that are connected.

The model of base dissimilarity can be also perceived as a composite one but from a different perspective. It does not combine multiple measurement methods but it is essentially an average value of *relative distance* between two boxes, their *shape* dissimilarity and the dissimilarity of their *colors*.

There are three reasons why only three visual aspects of boxes are used: 1) all the three aspects are defined for all the boxes on every web page that exists, 2) it makes the dissimilarity model simple and efficient and 3) simple dissimilarity model enables fine-tuning of the clustering step significantly. Furthermore, having just this simple model opens door to subsequent research that can quickly increase the Box Clustering Segmentation precision further.

5.4.1 Relative Distance

The distance measurement is based on relative distances between adjacent boxes. To calculate relative distance, it is first necessary to know absolute distances. This starts with identifying a *direct neighborhood* of each box – the boxes that are the closest to the inspected box in each respective direction in terms of absolute distance. The way how the absolute distances are computed is graphically expressed in figure 5.3.

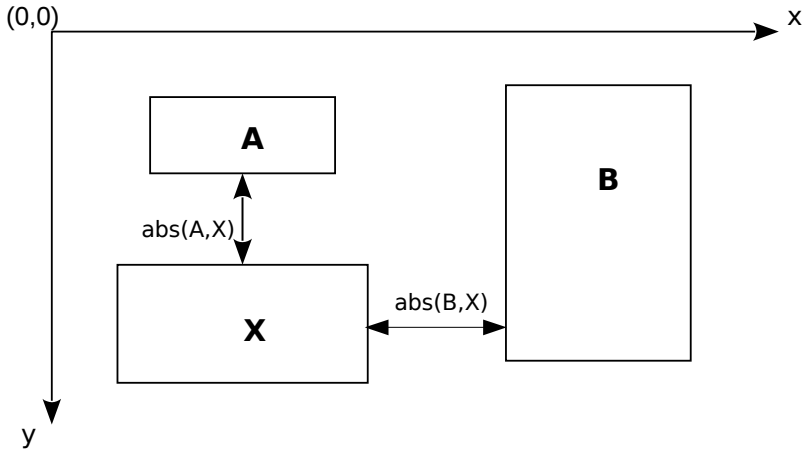

Figure 5.3: Absolute distance measurement between boxes

Using the direct neighborhoods and absolute distances, it is possible to calculate the relative distance, denoted *distance*():

Definition 2 (Relative distance) *Let B be a set of all boxes on a web page and N_m be a set of boxes in direct neigborhood of box $m \in B$. Furthermore, let $maxd(m)$ be maximal absolute distance between m and N_m. The relative distance between m and each $n \in N_m$ is:*

$$distance(m,n) = \frac{\frac{abs(m,n)}{maxd(m)} + \frac{abs(m,n)}{maxd(n)}}{2} \tag{5.1}$$

5.4.2 Shape

The premise here is that two adjacent boxes that have similar shape are more likely to belong to the same cluster than two boxes with completely different shapes. The most common case where the shape dissimilarity plays significant role are menu items, especially in vertical menus. Another very common situation is demonstrated by a simple paragraph of text where each line creates a separate box and all such boxes have very similar shape.

Shape dissimilarity is an average value of aspect ratio dissimilarity and size dissimilarity. Correspondence to semantics of other base dissimilarity elements is the factor driving how are both parts of shape dissimilarity calculated. In the following equations, $ratio()$ and $size()$ are the respective functions for aspect ratio and size components of shape dissimilarity:

$$r_m = \frac{m.width}{m.height} \tag{5.2}$$

$$r_n = \frac{n.width}{n.height} \tag{5.3}$$

$$s_m = m.width * m.height \tag{5.4}$$

$$s_n = n.width * n.height \tag{5.5}$$

$$ratio(m,n) = \frac{max\{r_m, r_n\} - min\{r_m, r_n\}}{\frac{max\{r_m, r_n\}^2 - 1}{max\{r_m, r_n\}}} \tag{5.6}$$

$$size(m,n) = 1 - \frac{min\{s_m, s_n\}}{max\{s_m, s_n\}} \tag{5.7}$$

5.4.3 Color

Color comparison is again an obvious one. Boxes with the same color are likely to belong to the same cluster than boxes with completely different colors. Color dissimilarity of two boxes is based on color distance calculation.

Lab and *LCH* based color distances were evaluated for their potential utilization in the BCS algorithm. However an observation was made during the experiments that simple euclidean *RGB*-based difference offers much better results. The most likely cause for this observation is that on most web pages, hue is used much more often than chroma to visually distinguish components on the web page that belong to different semantic blocks of the web page (e.g. navigation vs. article heading). Because the results of the color distance have to match all the other components of the box comparison, a linear normalization of the Euclidean distance to a target range of $< 0, 1 >$ is performed.

5.4.4 Alignment

Alignment is not strictly one of the components of the dissimilarity model, as it is not a visual aspect of either the boxes or their dissimilarity. It is however used as adjusting value that is applied as the last step of dissimilarity calculation. Alignment is represented by an integer value that specifies how many boxes are aligned with the two that are being inspected. Two boxes are aligned when either their top or left edges have the same coordinate. Obviously for the alignment to be higher than one, the two inspected boxes need to be aligned themselves.

5.5 Cluster Dissimilarity

Cluster Dissimilarity is used only during the clustering step of the algorithm. It is different from the base dissimilarity because clusters can't inherit features of the boxes they contain. The only option how to do that reliably would be to determine how exactly individual boxes contribute to the appearance of the entire cluster. However this task would be too time consuming for the algorithm to be practically usable.

Cluster dissimilarity model is demonstrated in figure 5.4. The model derives direct neighborhood of each cluster from direct neighborhoods of all the boxes contained in that cluster. Informally, it is possible to say that direct neighborhood of a cluster includes boxes that are in direct neighborhood of any of the boxes within that cluster and all clusters containing at least one box that is in direct neighborhood of any box within that cluster.

For dissimilarity, the situation is similar – the value of dissimilarity between a cluster and any entity in its direct neighborhood represents the mean value of dissimilarities between that entity and all the boxes contained in the cluster. In figure 5.4, the box-cluster connection `X` represents connections `A`, `B` and `C` when cluster `c` is created. The dissimilarity assigned to `X` therefore is calculated from their respective dissimilarities.

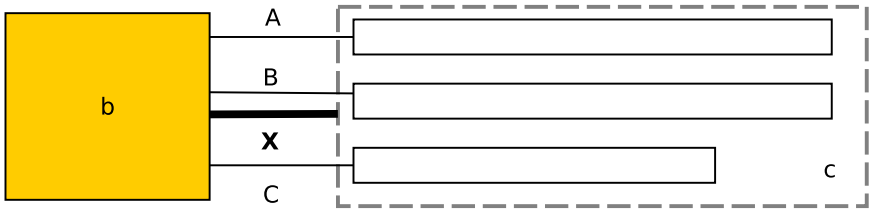

Figure 5.4: Demonstration of cluster dissimilarity model

5.6 Box Clustering

The clustering algorithm implements bottom-up hierarchical clustering where the depth of the hierarchy is limited to two levels – clusters at the top level and boxes at the bottom level. The goal of the algorithm is to find such a set of clusters that will minimize the set of unclustered boxes. The algorithm is iterative, exactly one new cluster is created by merging two existing entities in every iteration. Several steps take place for that to happen:

Selection of the most similar pair of entities

Selection of the most similar pair of entities is straightforward, as all the information required in directly stored in the graph of entities.

Stopping the clustering algorithm is also a part of this step. The algorithm is stopped when there are no pairs of entities that can be merged – the list of pair is either empty or no pair has a dissimilarity lower than *Clustering Threshold*, designated also as CT. The CT is a numeric value that can assume values between 0 and 1. Its purpose corresponds to that of Permitted Degree of Coherence (PDoC) used in VIPS algorithm[9]. It has to be set in advance and it stays the same for the entire page. Picking the right value of CT is

a difficult task, as it is application and page specific. For that reason, selecting the right value of CT is out of scope of this thesis, much like selecting PDoC is out of scope of the VIPS algorithm.

Mergeability testing

Mergeability testing is important especially in later iterations of the clustering algorithm. The idea of this testing is to prevent merging clusters that should not be merged because of their visual inconsistencies. These inconsistencies are based on the shape of clusters, their density and their disjunction. The merge test of two entities contains the following set of conditions.

- If the two entities are side-by-side columns and one is more than twice as big as the other, forbid the merge
- If the two entities are rows one below another and one is more than twice as big as the other, forbid the merge
- If the shapes of the two entities differ and cluster created by their merge overlaps with other entities, forbid the merge
- Otherwise allow the merge

Overlap testing and cluster extending

At this stage of the process, the newly formed cluster is only temporary and its viability needs to be tested before it is included in the set of clusters.

The point of viability testing is to adjust clusters so they don't overlap with other clusters and/or boxes that are not part of the cluster. If the candidate cluster cannot be adjusted to match the conditions, its creation is prevented and the clustering algorithm continues by starting another iteration (selection of the most similar pair of entities, ...). The testing/extending process is iterative – in each iteration, unclustered boxes overlapping with the temporary cluster are included in the new cluster minimal bounding box is verified not to overlap with previously non-overlapping standalone boxes or clusters. If that happens, the creation of the new cluster is stopped.

If the viability of the cluster is verified, it is included in the set of clusters that were detected.

---

6 Template clustering

This chapter will focus on the rest of the non-trivial components that were not covered in previous chapters. In figure 4.1, these are designated *Template storage*, *Template comparator* and *Node mapping*.

In multi-page segmentation scenario, the standard approach is to segment every web page separately before sending it to further processing. For some large servers like world-wide news servers, this means performing the segmentation task hundreds of thousands of times. It is obvious that this approach does not scale well and the time required to process

even mid-sized web sites is unacceptably long. The template clustering algorithm is designed specifically to address this scaling issue.

To avoid confusion created by combination of box clustering and template clustering, the clusters as defined in the previous chapter will be referred to as *visual areas* in this chapter. The reason for introducing this alias is to clearly distinguish between clusters of boxes and clusters of web pages.

6.1 Template clustering overview

The general idea of the template clustering can be presented as follows. When processing more pages within the same site, it is possible to indirectly increase the performance of segmentation by actually performing it only on a limited number of pages and transform these pages so that they can represent their respective template-based clusters. When, in one of the following iterations of the multi-page segmentation scenario, the inspected page is matched to an existing cluster, an isomorphic mapping between the page and the structure representing the corresponding cluster can be used to get the results of page segmentation without performing it. The time optimization achieved by this approach is in direct correlation with the number of web pages processed.

6.2 Template storage and selection

Time optimization often implies higher space requirements. However, due to the nature of the data, a stream clustering algorithm is required and the footprint of the *Template storage* is therefore rather small – only one rather simple data structure per cluster. These data structures will be further called Cluster Representatives and each one consists of three parts:

- Template represented by lightweight version of DOM tree
- Set of the visual areas that are returned by the Box Clustering Segmentation algorithm
- Mapping between elements of the previous two. This can also be obtained by reading data produced by the Box Clustering Segmentation.

The goal of *Template comparator* is to determine if the *Template storage* contains any record of previously segmented web page that the incoming one resembles. If yes, the record is used for further processing and the segmentation process is skipped. In its simplest design, the Template comparator selects all the Cluster representatives for the site that the input page belongs to and tests its resemblance to the representatives one-by-one. Since the number of Cluster representatives for each site is relatively small (see section 7.1), this approach scales sufficiently.

The resemblance is tested using slightly modified version of Gottron's Common Paths Distance algorithm. The Template comparator looks for the best match among the templates that share at least 70% of tree paths with the inspected page. If no such match is found, the inspected page is considered to be using a new template and the corresponding Cluster Representative is created.

6.3 DOM Tree Mapping

The purpose of *Node mapping* is to assign the right nodes of the input DOM tree D_{in} to the right visual areas of the Cluster Representative. To do that, it is necessary to find an isomorphic mapping between D_{in} and the simplified DOM tree of the Cluster Representative D_r. However, considering the required purpose, it is possible to significantly simplify the tree-mapping problem.

The most important simplification is that for purposes of locating the right content within the input DOM tree, the algorithm does not have to perform full node-to-node mapping. Instead, it is sufficient to just locate a subtree rooted at a specific node. The assumption is that once the root is found, all descendant nodes correspond in both trees. To verify this assumption, the mapping algorithm performs a validation of the found root node by calculating the same modified version of Gottron's Common Paths Distance algorithm that is used in Template comparator.

When looking for the root node of a subtree, there is one key feature of DOM trees that is utilized and that is the fact that order matters in DOM trees. Therefore it's safe to consider that DOM node of D_{in} and a corresponding DOM node of D_r will be on the same position within their siblings, at least in those parts of DOM tree that are considered a template. The same applies for the parent of such node and the same applies recursively up to the root node of a DOM tree. We can therefore use *path of positions* – a string of numbers that describes the path from root to the searched node – to identify a node within DOM tree.

Extending the path of positions with error detection mechanims, we get a *path of distinguished positions*. Such a path contains not only positions of nodes among their siblings but it also contains a combination of attributes/labels that describe each node in the path as unambiguously as possible. In this thesis a combination of the DOM node label together with `id` attribute (if the element has one) and the number of nodes on the level of the node (i.e. the number of siblings including the node itself) is used to identify each node in the path. The reason is that all these attributes are either universally available or at least quite common in most DOM trees.

7 Evaluation

This chapter will summarize and evaluate various aspects of the proposed segmentation algorithm and it will compare it with the VIPS algorithm which is still the most widely used segmentation algorithm. Both the quality of results and the performance boost will be evaluated here. The implementation of VIPS used here was written as master's thesis by Tomáš Popela [29] and it was designed specifically for the purpose of this comparison. That implies two things: it was written in Java and it uses CSSBox as the rendering core. That way it's possible to make a safe comparison – otherwise the results would be influenced by both runtime environment and different results provided by the rendering core. The VIPS implementation is in compliance with the algorithm as described in [9].

7.1 Template Count

One of the key points of the template clustering described in chapter 6 is to make the algorithm as scalable as possible. However the entire concept is based on the assumption that the number of templates on one web site is very small compared to the total number of pages generated using these templates. Another important assumption was that at a certain point the number of templates does not grow any more and in general it grows only very slowly in comparison with the number of pages inspected.

Figure 7.1 graphically demonstrates the growth of template count with raising number of inspected pages within a single site. It confirms that the number of templates discovered on a single site converges fast to a relatively small number.

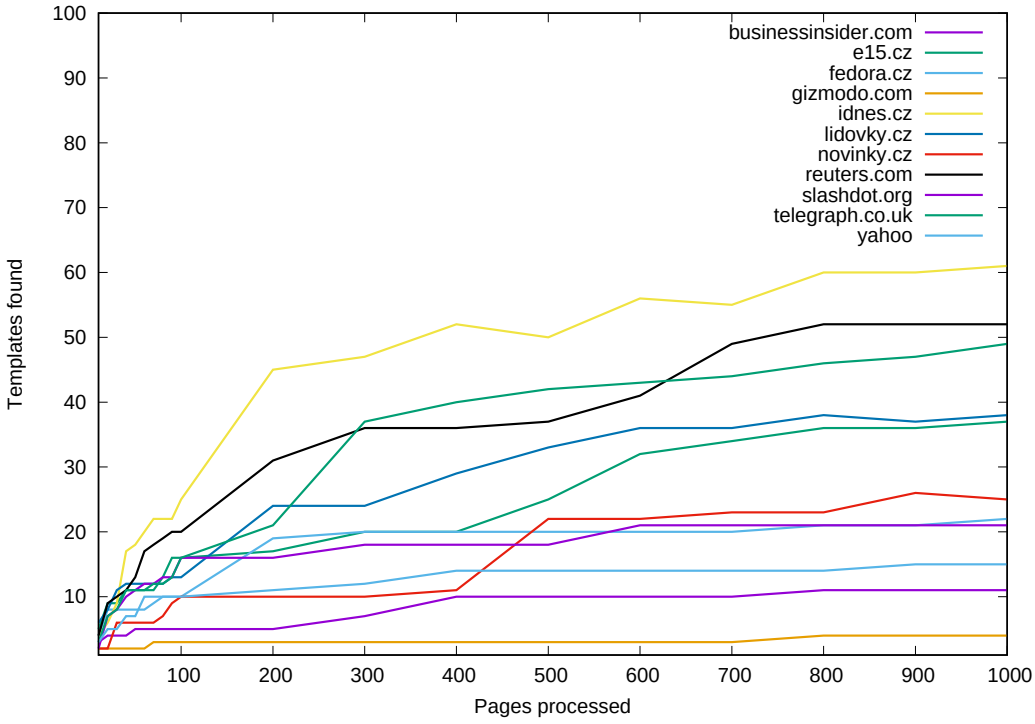

Figure 7.1: The dependency of cluster count on page count

7.2 Evaluation of the BCS

BCS is direct competitor of VIPS, it will be therefore evaluated against it. For the evaluation, a dataset of 2400 annotated web pages of three different types was created: *Complex index pages*, *Articles* and *Simple web pages*. The annotation was semi-automatic – some pages were annotated manually and the results were then mapped to the rest of the pages.

7.2.1 Performance

Table 7.1 demonstrates the first part of the algorithm evaluation – the run times of both algorithms. As explained above, multiple web page types are represented in the table. As the table 7.1 demonstrates, the BCS algorithm is superior to VIPS in terms of time necessary to process a web page. This difference gets bigger with decreasing complexity of evaluated web page.

page	VIPS time	BCS time
businessinsider.com (article)	522 ms	20 ms
idnes.cz (index)	1079 ms	39 ms
idnes.cz (article)	723 ms	53 ms
novinky.cz (index)	28126 ms	699 ms
novinky.cz (article)	390 ms	18 ms
reuters.com (index)	475 ms	15 ms
reuters.com (article)	442 ms	37 ms
yahoo news (article)	342 ms	21 ms

Table 7.1: Algorithm run time comparison

7.2.2 Accuracy and stability

Evaluating the accuracy is more complex. In statistical analysis in general, F-score is a common way how to evaluate accuracy. At the same time, as this thesis proposes, the page segmentation task, regardless of how it's performed, is basically a clustering task. In data clustering, Adjusted Rand Index (ARI) [16] is used to measure similarity between two clusterings (evaluated and reference segmentation). Due to the lack of existence of commonly accepted reliable method for segmentation accuracy evaluation, both the F-score and ARI are being used in this evaluation.

Table 7.2 shows F-score and ARI comparison of BCS and VIPS. For each comparison, the best respective values of both algorithms were chosen. The F-score value is a real number between 0 and 1, where higher values are better. ARI score is between -1 and 1, higher values are better.

page	BCS ARI	VIPS ARI	BCS F	VIPS F
businessinsider.com (a)	0,5704	0,7010	0,6345	0,7394
idnes.cz (article)	0,6629	0,7240	0,5570	0,5720
idnes.cz (index)	0,5954	0,7926	0,5522	0,7259
novinky.cz (article)	0,7670	0,7877	0,6446	0,7191
novinky.cz (index)	0,5303	0,9121	0,4265	0,9043
reuters.com (article)	0,6123	0,6786	0,5914	0,6943
reuters.com (index)	0,5832	0,8160	0,5145	0,7569
yahoo news (article)	0,7556	0,5626	0,7102	0,5446

Table 7.2: Algorithm accuracy comparison using the ARI and F-score metrics

The results show that the accuracy of VIPS is slightly better, especially when processing structured pages. The reason is that BCS is too aggressive when creating clusters, thus effectively overlooking the structure. VIPS on the other hand does much better job in

finding repeating patterns in the web page. When processing pages with less structure, the accuracy of BCS and VIPS is comparable, in some cases BCS is even better than VIPS.

Stability of both algorithms was calculated using the same data that was used to populate table 7.2. The algorithm is stable if the quality of its results is stable across different web pages. The stability was calculated for each page type in each web site (i.e. different pages with the same template but different content in that template).

Both algorithms are comparable in terms of stability. In some cases the stability of BCS is almost three times better than that of VIPS, in others, it's exactly the opposite. Not looking at the degree of superiority, the stability of BCS is better in 62.5% of measurements.

7.3 Evaluation of BCS with template clustering

The comparison is between the algorithm combining both Box Clustering Segmentation and template clustering and the plain VIPS algorithm. The test consisted of running segmentation on 500 web pages and calculating total time it took to perform the segmentation. The results in table 7.3 clearly say that in the use case of multi page segmentation, the proposed method is on average more than 80 percent more efficient than plain VIPS.

site	**VIPS**	**BCS + template clustering**	savings
businessinsider.com	1 538 298 ms	33 741 ms	97,81%
idnes.cz	1 173 891 ms	262 486 ms	77,64%
novinky.cz	350 613 ms	21 791 ms	93,78%
reuters.com	882 502 ms	38 752 ms	95,61%
yahoo news	1 307 889 ms	40 955 ms	96,87%

Table 7.3: Performance of the proposed algorithm

8 Conclusion

The proposed method can be used in real life applications as a preprocessor of the content on the World Wide Web. Obviously the method offers the greatest potential when used in unsupervised high volume preprocessing. In this context, it means processing thousands of web pages as a stream of data. In these volumes, even a small relative performance boost is significant in absolute numbers. The significance of performance boost this thesis offers is even higher. Even when web pages are not processed in such high volumes, the boost is considerable.

As for precision of the Box Clustering Segmentation algorithm, this thesis demonstrates that the results are comparable to the most commonly used existing method – VIPS. In some cases the precision of BCS algorithm is even better while its stability is slightly better in general.

Some issues of the proposed method remain when segmenting large and complex web pages. Considering the targeted use case, their impact can be perceived as marginal, as the complex web pages usually serve as guide posts, with minimal amount of useful content. If

the significance of such pages was higher, there are techniques how to enhance the proposed method to overcome this problem.

8.1 Summary of Contributions

The proposed method presents a completely fresh perspective on the web page segmentation task. The main contributions contained in this thesis include:

1. Performance superiority: the segmentation method proposed in this thesis performs better on every level when compared to current algorithms. The combination of these individual levels multiplies the performance benefit.

2. *Quality of result:* while the overall quality of results is comparable, there are some aspects that make Box Clustering Segmentation better than the rest. Using the flat model for results is much more convenient for consumers of these results, as no further post-processing is required. Better handling of some web pages is also one of the significant contributions this thesis makes.

3. *Pure vision-based method:* even though previous methods claimed to be vision based, this is, to the best knowledge of the author, the first method that is really based purely on visual cues. This contribution is most significant on highly dynamic web pages where other cues, such as DOM tree features, might be highly misleading.

4. *Re-usability:* the fact that the Box Clustering Segmentation segmentation algorithm is not tightly bound to the DOM model implies that it can be used for other document types, if there is a convenient parser that provides the right input to the segmentation algorithm.

5. *Boost of segmentation methods in general:* the template clustering approach not being tightly bound to the Box Clustering Segmentation presents a way how to significantly improve the performance of all vision-based segmentation algorithms in the field.

Aside from these main contributions, this thesis offers some that are more general but almost equally important.

1. *Practical application:* we rarely want to segment just one page and never visit it again. While the existing methods focus on this only task, this thesis provides more complex view of the segmentation problem. When considering the big picture (i.e. how is the algorithm likely to be used), the design of the methods shifts to address the specific needs of targeted use cases.

2. *New research areas:* this thesis proofs that clustering techniques can be successfully used at different levels of the web page preprocessing. While there was some existing work in the area before, the author is not aware of any comprehensive research on this topic. The research provided in this thesis can serve as a base for future scientific work in this area.

3. *Connection of two different research areas:* to author's best knowledge, this thesis is the first one that combines web page segmentation with template detection to get the benefits of both while minimizing their drawbacks.

Finally, one of the contributions of this thesis is a working implementation of both techniques described in this thesis. The architecture of this implementation allows simple utilization of each part separately.

Bibliography

[1] Elgin Akpinar and Yeliz Yesilada. Vision based page segmentation: Extended and improved algorithm. Technical Report eMINE Technical Report Deliverable 2 (D2), Middle East Technical University, Ankara, Turkey, 2012.

[2] M. Elgin Akpinar and Yeliz Yesilada. Vision based page segmentation algorithm: Extended and perceived success. In *Revised Selected Papers of the ICWE 2013 International Workshops on Current Trends in Web Engineering - Volume 8295*, pages 238–252, New York, NY, USA, 2013. Springer-Verlag New York, Inc.

[3] Julián Alarte, David Insa, Josep Silva, and Salvador Tamarit. Temex: The web template extractor. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 155–158, New York, NY, USA, 2015. ACM.

[4] Derar Alassi and Reda Alhajj. Effectiveness of template detection on noise reduction and websites summarization. *Information Sciences*, 219:41 – 72, 2013.

[5] Sadet Alcic and Stefan Conrad. Page segmentation by web content clustering. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*, WIMS '11, pages 24:1–24:9, New York, NY, USA, 2011. ACM.

[6] Jayendra Barua, Dhaval Patel, and Ankur Kumar Agrawal. Removing noise content from online news articles. In *Proceedings of the 20th International Conference on Management of Data*, COMAD '14, pages 113–116, Mumbai, India, India, 2014. Computer Society of India.

[7] Zhan Bu, Chengcui Zhang, Zhengyou Xia, and Jiandong Wang. An far-sw based approach for webpage information extraction. *Information Systems Frontiers*, 16(5):771–785, 2014.

[8] Radek Burget. Visual area classification for article identification in web documents. In *Proceedings of the 2010 Workshops on Database and Expert Systems Applications*, DEXA '10, pages 171–175, Washington, DC, USA, 2010. IEEE Computer Society.

[9] Deng Cai, Shipeng Yu, Ji rong Wen, and Wei ying Ma. VIPS: a vision-based page segmentation algorithm. Microsoft technical report MSR-TR-2003-79, November 2003.

[10] Michael Cormier, Karyn Moffatt, Robin Cohen, and Richard Mann. Purely vision-based segmentation of web pages for assistive technology. *Computer Vision and Image Understanding*, 2016, 2016.

[11] Hassan F. Eldirdiery and A. H. Ahmed. Detecting and removing noisy data on web document using text density approach. *International Journal of Computer Applications*, 112(5):32–36, February 2015.

[12] Fabio Fumarola, Tim Weninger, Rick Barber, Donato Malerba, and Jiawei Han. Extracting general lists from web documents: A hybrid approach. In *Proceedings of the 24th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems Conference on Modern Approaches in Applied Intelligence - Volume Part I*, IEA/AIE'11, pages 285–294, Berlin, Heidelberg, 2011. Springer-Verlag.

[13] Bo Gao and Qifeng Fan. Multiple template detection based on segments. In *Advances in Data Mining. Applications and Theoretical Aspects*, pages 24–38. Springer, 2014.

[14] Thomas Gottron. Bridging the gap: from multi document template detection to single document content extraction. In *Proceedings of the IASTED International Conference on Internet and Multimedia Systems and Applications*, EuroIMSA '08, pages 66–71, Anaheim, CA, USA, 2008. ACTA Press.

[15] Jer Lang Hong, Eu-Gene Siew, and Simon Egerton. Information extraction for search engines using fast heuristic techniques. *Data Knowl. Eng.*, 69(2):169–196, February 2010.

[16] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[17] K. Jiang and Y. Yang. Noise reduction of web pages via feature analysis. In *Information Science and Control Engineering (ICISCE), 2015 2nd International Conference on*, pages 345–348, April 2015.

[18] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate detection using shallow text features. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 441–450, New York, NY, USA, 2010. ACM.

[19] J. Kong, O. Barkol, R. Bergman, A. Pnueli, S. Schein, K. Zhang, and C. Zhao. Web interface interpretation using graph grammars. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):590–602, July 2012.

[20] S. S. Krishna and J. S. Dattatraya. Schema inference and data extraction from templatized web pages. In *Pervasive Computing (ICPC), 2015 International Conference on*, pages 1–6, Jan 2015.

[21] AH Kulkarni and BM Patil. Template extraction from heterogeneous web pages with cosine similarity. *International Journal of Computer Applications*, 87(3):5, 2014.

[22] Harshal H Kulkarni and Manasi K Kulkarni. Template extraction from heterogeneous web pages. *International Journal of Electrical, Electronics and Computer Engineering*, 4(1):125, 2015.

[23] Eduardo Sany Laber, Críston Pereira de Souza, Iam Vita Jabour, Evelin Carvalho Freire de Amorim, Eduardo Teixeira Cardoso, Raúl Pierre Rentería,

Lúcio Cunha Tinoco, and Caio Dias Valentim. A fast and simple method for extracting relevant content from news webpages. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1685–1688, New York, NY, USA, 2009. ACM.

[24] Long Li, An Min Zhou, Yong Fang, Liang Liu, and Qian Wu. An improved VIPS-based algorithm of extracting web content. In *Material Science, Civil Engineering and Architecture Science, Mechanical Engineering and Manufacturing Technology II*, volume 651 of *Applied Mechanics and Materials*, pages 1806–1810. Trans Tech Publications, 11 2014.

[25] Wei Liu, Xiaofeng Meng, and Weiyi Meng. ViDE: A vision-based approach for deep web data extraction. *IEEE Trans. on Knowl. and Data Eng.*, 22(3):447–460, March 2010.

[26] Xinyue Liu, Hongfei Lin, and Ye Tian. Segmenting webpage with gomory-hu tree based clustering. *Journal of Software*, 6(12):2421–2425, 2011.

[27] Erik Lundgren, Panagiotis Papapetrou, and Lars Asker. Extracting news text from web pages: An application for the visually impaired. In *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '15, pages 68:1–68:4, New York, NY, USA, 2015. ACM.

[28] Tomohiro Manabe and Keishi Tajima. Extracting logical hierarchical structure of html documents based on headings. *Proceedings of the VLDB Endowment*, 8(12):1606–1617, 2015.

[29] Tomáš Popela. Implementace algoritmu pro vizuální segmentaci www stránek. Master's thesis, Brno University of Technology, Faculty of Information Technology, 2012.

[30] Waseem Safi, Fabrice Maurel, Jean-Marc Routoure, Pierre Beust, and Gaël Dias. A Hybrid Segmentation of Web Pages for Vibro-Tactile Access on Touch-Screen Devices. In *3rd Workshop on Vision and Language (VL 2014) associated to 25th International Conference on Computational Linguistics (COLING 2014)*, pages 95 – 102, dublin, Ireland, Aug 2014.

[31] A. Sanoja and S. Gançarski. Block-o-matic: A web page segmentation framework. In *Multimedia Computing and Systems (ICMCS), 2014 International Conference on*, pages 595–600, April 2014.

[32] Shengsheng Shi, Chengfei Liu, Yi Shen, Chunfeng Yuan, and Yihua Huang. Autorm: An effective approach for automatic web data record mining. *Knowledge-Based Systems*, 89:314–331, 2015.

[33] Dandan Song, Fei Sun, and Lejian Liao. A hybrid approach for content extraction with text density and visual importance of dom nodes. *Knowledge and Information Systems*, 42(1):75–96, 2015.

[34] E. Uzun, H. V. Agun, and T. Yerlikaya. Web content extraction by using decision tree learning. In *2012 20th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4, April 2012.

[35] Erdinç Uzun, Hayri Volkan Agun, and Tarık Yerlikaya. A hybrid approach for extracting informative content from web pages. *Information Processing & Management*, 49(4):928 – 944, 2013.

[36] Gabriel Valiente. An efficient bottom-up distance between trees. In *Proceedings of the 8th International Symposium of String Processing and Information Retrieval*, pages 212–219. Press, 2001.

[37] Karane Vieira, André Luiz Costa Carvalho, Klessius Berlt, Edleno S. Moura, Altigran S. Silva, and Juliana Freire. On finding templates on web collections. *World Wide Web*, 12(2):171–211, June 2009.

[38] T. Wei, Y. Lu, X. Li, and J. Liu. Web page segmentation based on the Hough transform and vision cues. In *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pages 865–872. IEEE, 2015.

[39] Daiyue Weng, Jun Hong, and David A. Bell. Extracting data records from query result pages based on visual features. In *Advances in Databases: 28th British National Conference on Databases, BNCOD 28, Manchester, UK, July 12-14, 2011, Revised Selected Papers*, pages 140–153, Berlin, Heidelberg, 2011. Springer.

[40] Daiyue Weng, Jun Hong, and David A. Bell. Automatically annotating structured web data using a svm-based multiclass classifier. In *Web Information Systems Engineering – WISE 2014: 15th International Conference, Thessaloniki, Greece, October 12-14, 2014, Proceedings, Part I*, pages 115–124, Cham, 2014. Springer International Publishing.

[41] Yu-Chieh Wu. Language independent web news extraction system based on text detection framework. *Information Sciences*, 342:132 – 149, 2016.

[42] Zhen Xu and James Miller. Identifying semantic blocks in web pages using gestalt laws of grouping. *World Wide Web*, pages 1–22, 2015.

[43] Jun Zeng, Brendan Flanagan, Sachio Hirokawa, and Eisuke Ito. A web page segmentation approach using visual semantics. *IEICE Transactions on Information and Systems*, E97-D(2):223–230, February 2014.

[44] Kaizhong Zhang, Rick Statman, and Dennis Shasha. On the editing distance between unordered labeled trees. *Inf. Process. Lett.*, 42:133–139, May 1992.

[45] W. Zhu, S. Dai, Y. Song, and Z. Lu. Extracting news content with visual unit of web pages. In *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2015 16th IEEE/ACIS International Conference on*, pages 1–5, June 2015.

Author's Publications

1. Zelený, J.: Web Page Segmentation and Classification. In *Proceedings of the 17th Conference Student EEICT 2011* Volume 3. Brno, Czech Republic. 2011. ISBN 978-80-214-4273-3
2. Zelený, J., Burget, R.: Cluster-based Page Segmentation – a fast and precise method for web page pre-processing. In *The Third International Conference on Web Intelligence, Mining and Semantics*. Madrid, Spain. 2013.
3. Zelený, J., Burget, R.: Isomorphic mapping of DOM trees for Cluster-Based Page Segmentation. In *Proceedings of the Twelfth International Conference on Informatics INFORMATICS'2013*. Spišská Nová Ves, Slovakia. 2013.
4. Zelený, J., Burget, R.: Accelerating the process of web page segmentation via template clustering. In *International Journal of Intelligent Information and Database Systems* Vol. 9, No. 2, 2016.
5. Zeleny, Jan, Radek Burget, and Jaroslav Zendulka. Box clustering segmentation: A new method for vision-based web page preprocessing. In *Information Processing & Management* Vol. 53, No. 3. 2017.

Author's Other Activities

1. Teaching
   (a) 1 semester of *Introduction to Programming Systems*: projects, labs
   (b) 1 semester of *Introduction to Software Engineering*: projects
   (c) 2 semesters of *Web Design*: projects, labs
2. Leadership of master's and bachelor's theses: 13 defended
3. Participation at FRVS 2013 grant – Update of *Web design* course
4. Participation in project *Advanced recognition and presentation of multimedia data*
5. Participation in project *IT4Innovations excellence in science*
6. Red Hat Ambassador with FIT: coordination of shared R&D projects, negotiating new courses

Author's Structured CV

Professional experience

Since 2016	**Manager, Red Hat** Manager of two teams within Next generation platform organization. Handling all the tasks of Supervisor. Experience with Scrum – started three different scrum teams spread across multiple time zones, acted as product owner for two of them.
2012-2016	**Supervisor, Red Hat** Handling hiring, people management tasks, representation of the team within Red Hat, driving cross-team collaboration, project planning and execution. Also driving some site-wide activities such as collaborative research with local university and employee orientation.
2010-2012	**Software Engineer, Identity Management team, Red Hat** Working on SSSD project – developing new features, representing the team in all Red Hat programs for university cooperation.
2009-2010	**Associate Software Engineer, Base OS team, Red Hat** Taking care of multiple different packages in Fedora and Red Hat Enterprise Linux – debugging, fixing bugs, developing new features, backporting patches.
Since 2015	**CTO and Chief Software Engineer, Tiketbox s.r.o.** complete product and technical design of web application for online ticketing; setting technical strategy, positioning the flagship product in the market; development, testing and deployment; budgeting; presenting the product to potential customers
Since 2003	**freelance contractor** developing mostly mid-size projects for various customers. Using variety of languages and technologies.

Education

Since 2010	**Ph.D. study programme, Brno University of Technology**
2010	**M.Sc. in Information Technology, Brno University of Technology**

Language skills

Czech:	native
English:	advanced, both written and spoken