Optimizing hierarchical menus: a usage-based approach
Hollink, V.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Chapter 5

A semi-automatic usage-based method for improving hyperlink descriptions

The previous chapters addressed the optimization of structures of menus. In this chapter we turn our attention to descriptions of menu items. We present a novel algorithm to automatically detect links with problematic descriptions on the basis of usage information. The algorithm distinguishes several types of problematic descriptions and provides recommendations for how problematic descriptions of each type can be improved. The findings of the algorithm can help a webmaster of a site to gain insights in the behavior of the users of the site and to improve the site accordingly.

This chapter is based on a paper co-authored with M. W. van Someren and B. J. Wielinga, which has been submitted for publication. Earlier versions were presented at the Sixteenth Annual Machine Learning Conference of Belgium and the Netherlands (Hollink and Van Someren, 2007) and the Workshop on Data Mining for User Modeling (Hollink et al., 2007d).

5.1 Introduction

Users who navigate unfamiliar web sites in search of information choose links on the basis of link anchors and the text surrounding the links. When these descriptions are accurate, users can make the right choices and reach their target information efficiently. Unfortunately, it often happens that the meaning of link descriptions is ambiguous or otherwise unclear, so that users are forced to try out various links before they can find what they need.

Designers of web sites face the task of inventing descriptions that the users of the site will understand well. This already challenging task is further complicated by the fact that the information needs of the user population are often not known in advance. Descriptions can be clear from one perspective, but ambiguous for users with other points of view. Moreover, initially well-designed descriptions can become unsuitable when the content of the pages is changed or when new links are added.

Many tools have been developed to help a webmaster to improve a site’s link
structure (e.g. Wang et al., 2006; Nakayama et al., 2000; Wu et al., 2005), but the improvement of link descriptions has received much less attention. This is surprising as several studies have shown that the impact of navigation errors on navigation efficiency is much larger than the impact of the link structure (Miller and Remington, 2004; Hollink and Van Someren, 2006 (Chapter 2)). Since good link descriptions reduce the number of navigation errors, much can be gained from methods to improve link descriptions.

In this work we focus on the improvement of link descriptions in hierarchical link structures, such as menus and web directories. In these structures some links do not point directly to content, but to new sets of links. Finding good descriptions for these links is particularly difficult, because the descriptions must not only cover the contents of one page, but of the whole category of pages that are located under the link.

Although hierarchical link structures can be implemented in a variety of ways, we will refer to all such structures as menus. We define a menu as a whole hierarchical menu structure. Individual items in a menu are called menu items. The clickable part of a menu item we call a link, even though not all menus are implemented as HTML hyperlinks. The pages that are located in the subhierarchy under a link will be called the page set of the link. We make the assumption that users navigate through a menu to fulfill certain information needs. The term target pages\(^1\) will refer to the pages that provide a (partial) answer to their information needs.

We present a semi-automatic method that helps a webmaster to improve the link descriptions in a site’s menu. We identify six classes of problems that users can have with link descriptions. Each class is characterized by a specific usage pattern that can be observed in the site’s log files. Links with accurate and unambiguous descriptions are characterized by a usage pattern in which exactly those users who are looking for the pages behind the link, follow the link. Other usage patterns indicate that to some users the descriptions are not clear. We present an algorithm that analyzes the log files of a site and classifies the link descriptions in the site’s menu on the basis of their usage patterns in terms of well-understood descriptions or descriptions corresponding to a certain problem class.

Each problem class is associated with a number of generic solutions, such as making descriptions more specific or merging certain menu items. The solutions are presented to the webmaster together with the problems that the algorithm has found. The webmaster interprets the results and decides whether adaptations to the menu are necessary. He (or she) chooses the solution that he finds most appropriate and specifies the details of the adaptation.

The presented method is evaluated on the menus of three real web sites. During the first part of the evaluation we assess the responses of the webmasters of the sites to the identified problems and solutions. In the second part we measure in a user experiment whether the proposed changes to the menus cause users to make less navigation mistakes.

The remainder of this chapter is organized as follows. In the next section we review related work. In Section 5.3 we present the classes of problematic link descriptions.

\(^1\)In Cooley et al. (1999) target pages are called ‘content pages’.
Section 5.4 explains the algorithm to assign link descriptions to classes. In Section 5.5 we evaluate the method. The last section contains conclusions and discusses our results.

5.2 Related work

Several methods have been developed to automatically improve link structures. These methods analyze a link structure and a set of log files and determine which links need to be added or removed. For example, efficiency can be increased by placing more frequently used items at higher positions in a menu or by adding links between pages that are often viewed in the same sessions. Some of these methods autonomously change the structure (e.g. Smyth and Cotter, 2003; Wu et al., 2005; Wang et al., 2006). Others recommend possible adaptations and leave it to the webmaster to decide which adaptations are implemented (e.g. Nakayama et al., 2000; Hollink et al., 2007c (Chapter 3)). These methods have in common that they improve the topology of the link structure, but not the descriptions of the links.

Systems that add new links autonomously need to provide descriptions for these links. These systems often rely on hand-made rules to find descriptions. For instance, the m-Links system (Schilit et al., 2002) creates navigation menus for web pages that are suitable for viewing on mobile phones. The links in the menus are labeled with texts from predefined sources, such as page titles and URLs.

Adaptive link annotation refers to the process of attaching visual cues to links to help users select the most relevant links (Brusilovsky, 1996, 2001). For example, the Syskill and Webert system (Pazzani et al., 1996) asks a user to rate the pages he visits and learns a user profile from these ratings. On the basis of the profile, the system annotates links with symbols that indicate how interesting the system thinks the user will find the links. The methods described in Joachims et al. (1997) and Herder (2004) do not require ratings, but ask the user to describe the topic of his search. Links matching this topic are annotated with small pictures. Link annotations can provide specialized support as they are adapted to the personal needs of each user. A disadvantage of this approach is that elements are added to a page even when selecting the correct link is straightforward. In this case, the annotations unnecessarily increase the visual complexity of the page.

Nakayama et al. (2000) optimize both the structure and the descriptions of links. Their algorithm detects page pairs that are similar in content, but that are not frequently visited in the same session. They suggest to add a link between these pages if the pages are not yet linked. If a link is already present, they conclude that the page layout must be adapted to improve the visibility of the link. They propose several methods to improve the layout, including changing the link anchor or the text preceding the link. A limitation of their work is that they can only detect problematic links between pages that are very similar in content. The usage-based character of our method allows us to find problematic links between pages that are related in terms of user relevance, but that have different contents.

Srikant and Yang (2001) propose a method to discover the location in a web site
where users expect to find certain target pages. They assume that users follow links to the location where they believe a target is located and backtrack when they find out that no link to the target is present at the expected location. Their method computes for each target page the positions where users frequently backtrack and recommends to add links to the target at these positions. This approach is similar to our approach in that both methods aim to determine incorrect navigation paths. However, Srikant and Yang search for the end points of these paths (the backtrack points). They solve the problems at the end points by adding links to the target pages. In contrast, we determine the source of the problem, the point where users deviate from the optimal path. The problem is solved at these points by improving the link descriptions that gave users incorrect expectations about the contents of the underlying pages.

Even though the creation of link descriptions has not received much attention, link descriptions are recognized as a useful source of information for a wide variety of tasks. For example, Fürnkranz (1999) uses anchors of links and the text surrounding links for document classification. In Lu et al. (2002) anchor texts are used to translate search queries. Others use anchors to refine search queries (Kraft and Zien, 2004) or to improve search engines (Westerveld et al., 2002; Craswell et al., 2001). These methods all rely on high quality link descriptions to perform certain tasks, but are not aimed at obtaining such descriptions.

5.3 Problematic link descriptions

We divide problematic link descriptions in a number of classes. Each class represents a type of navigation mistake and is characterized by a particular usage pattern. The more classes we distinguish, the more insight we can give a webmaster in the navigation mistakes of the users. On the other hand, attempts to find too many classes can significantly reduce the accuracy of the classification. Therefore, we defined a set of classes that give enough information to choose an appropriate solution, but that can still be reliably detected in log data. In this section we present the problem classes and discuss possible solutions. Section 5.4 gives the algorithm that assigns link descriptions to the classes.

The classes are organized in a hierarchy as shown in Figure 5.1. At the top level the hierarchy distinguishes between well-understood (strong) and incorrectly understood (weak) descriptions. These classes tell a webmaster which link descriptions need improvement. The second layer of the hierarchy shows which links are clicked instead of the correct ones. This gives a webmaster information about the direction in which the descriptions need to be changed. The third layer tells a webmaster which target information the users were looking for when they made the navigation mistakes. This information can help him understand why the mistakes were made and which elements of the links need modification.

The classes of link descriptions are defined as follows:

**Strong** A description of a link is strong if users with target pages in the page set of the link can accurately predict that their target pages are located under the link.
5.3. Problematic link descriptions

A description of a link is weak if users with target pages in the page set of the link cannot accurately predict whether their target pages are located under the link or under another link.

Covered The description of link A is covered by the description of link B if users with target pages in the page set of link A frequently believe that their target pages are located under B.

Overlapping The descriptions of links A and B overlap if users with target pages in the page set of link A frequently believe that their target pages are located under B and users with target pages in the page set of link B frequently believe that their target pages are located under A.

Unclear A description of a link is unclear if users with target pages in the page set of the link cannot accurately predict that their target pages are located under the link, but the link is not confused with specific other links.

Partially unclear/covered/overlapping The description of a link A is partially unclear, covered or overlapping if the problem can be attributed to users with target pages in the page sets of certain specific subitems of A: the problematic subitems.

Completely unclear/covered/overlapping The description of a link A is completely unclear, covered or overlapping if the problem concerns all users with target pages in the page set of the link, in other words, if the problem cannot be attributed to users with target pages in the page sets of certain specific subitems of A.

The hierarchy shown in Figure 5.2 illustrates the various problem classes. This Figure shows the menu of the web site of a fictitious research project. The site contains information on the contact details of the department, the personal details of the two project members, the toolkit developed in the project and some practical information...
about various other aspects of the project. The link Project exemplifies the concept of an unclear link. The term Project is too general in this domain, as the whole site is about the project. As a result, people looking for the information located in this part of the hierarchy will not be able to infer that they need to click on Project. This holds for all three pages below Project, so that Project is a completely unclear link. Visitors who are looking for the email address of Mrs. Smith do not know whether they need to select People or Contact and will often incorrectly click on Contact. Therefore, we say that the link People is covered by the link Contact. As the problem occurs only for users who are looking for the Email page and not for the other pages under People, the link People is called partially covered.

![Figure 5.2: Example menu hierarchy.](image)

### 5.3.1 Finding solutions for problematic link descriptions

Once we determined which link descriptions are weak, the problematic descriptions need to be improved. The most obvious solution for a weak description is to change the text of the description. However, in some cases weak links can also be improved by changing the structure of the menu. For example, when a description covers only part of a link's page set, the problem can often be solved by moving the uncovered parts to another location in the menu. We provide several possible solutions to improve the various problem classes. The solutions are shown in Table 5.1.

The proposed solutions tell us in which directions link descriptions need to be changed, but they do not provide new descriptions. Several authors have proposed methods to automatically extract keywords from texts (e.g. Witten et al., 1999; Zamir and Etzioni, 1999; Lawrie et al., 2001; Zeng et al., 2004). The extracted keywords can be used as link descriptions, but the keywords are generally of poor quality. These alternatives will probably not help a webmaster to improve the link descriptions and possibly even tempt him to choose alternatives that are even worse than the original
5.4. Link description classification algorithm

In this section we explain how we assign link descriptions to the classes defined above. Figure 5.3 shows the algorithm in pseudocode. Below, we explain each element of the algorithm in detail. First, we describe how the log files are preprocessed. Then, we describe how the logs are used to distinguish between the problem classes in the various layers of the class hierarchy. The last subsection discusses the presentation of problematic descriptions to webmasters.

5.4.1 Preprocessing

Before we can detect patterns in user behavior, the log files need to be preprocessed. During preprocessing we restore the sessions of individual users and determine for each session the most likely target pages. In addition, the logs are cleaned by removing sessions that are not created by real users.

The sessions of individual users are restored with the method described in Cooley et al. (1999). All requests coming from the same IP address and the same browser are attributed to one user. When a user is inactive for more than 30 minutes, a new

<table>
<thead>
<tr>
<th>Problem class</th>
<th>Solution type</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A is partially unclear</td>
<td>Descriptions</td>
<td>Make the description of A broader, so that the problematic subitems are also covered. Move the problematic subitems to a new menu item.</td>
</tr>
<tr>
<td></td>
<td>Structure</td>
<td></td>
</tr>
<tr>
<td>A is completely unclear</td>
<td>Descriptions</td>
<td>Give A a new description that contains only terms that are known to the users and not too general.</td>
</tr>
<tr>
<td></td>
<td>Structure</td>
<td></td>
</tr>
<tr>
<td>B partially covers A</td>
<td>Descriptions</td>
<td>Make the description of B narrower, so that it no longer covers the problematic subitems under A. Move or copy the problematic subitems from A to B.</td>
</tr>
<tr>
<td></td>
<td>Structure</td>
<td></td>
</tr>
<tr>
<td>B completely covers A</td>
<td>Descriptions</td>
<td>Make the description of B narrower, so that it no longer covers the page set of A. Make A a subitem of B. Merge items A and B and give the new item the description of B.</td>
</tr>
<tr>
<td></td>
<td>Structure</td>
<td></td>
</tr>
<tr>
<td>A and B partially overlap</td>
<td>Descriptions</td>
<td>Make the descriptions of A and B narrower, so that they no longer contain the overlapping items. Create a new menu item that contains the problematic subitems.</td>
</tr>
<tr>
<td></td>
<td>Structure</td>
<td></td>
</tr>
<tr>
<td>A and B completely overlap</td>
<td>Descriptions</td>
<td>Give A and B new descriptions that make the differences between the two categories more clear. Merge items A and B.</td>
</tr>
<tr>
<td></td>
<td>Structure</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.1: Possible solutions for the various classes of weak link descriptions.
Chapter 5: A semi-automatic usage-based method for improving hyperlink descriptions

Algorithm 5.1: Evaluate_menu(Hierarchical menu menu)

```
problems ← ∅
for each fragment in menu
    fragment_problems ← ∅
    Create confusion matrix c
    for each Row in c with correct link l and incorrect links i
        do Add evaluate_link(i, l) to fragment_problems
    for each pair of problems pair in fragment_problems
        do if pair = covered_by(l, k) and covered_by(k, l)
            then Change pair into overlap(l, k)
            for each problem problem(link l) in fragment_problems
                do problem ← partial_or_complete_problem(problem(l))
            Add fragment_problems to problems
return problems
```

```
procedure evaluate_link(incorrect links i, correct link l)
    link_problems ← ∅
    if ψ ≥ row_frequency(l)/row_frequency(i ∪ l)
        then for each incorrect link k in i
            do if row_frequency(k)/row_frequency(i) ≥ 1/|i|
                then Add covered_by(l, k) to link_problems
            if link_problems = ∅
                then Add unclear(l) to link_problems
return link_problems
```

```
procedure partial_or_complete_problem(problem(link l))
    problematic_subitems ← ∅
    for each j in subitems(l)
        do if problem_proportion(j, problem) ≥ problem_proportion(l, problem)
            then Add j to problematic_subitems
    if problematic_subitems = ∅
        then return (complete_problem(l))
    else return (partial_problem(l, problematic_subitems))
```

Figure 5.3: The link description classification algorithm in pseudocode. ‘≥’ stands for ‘significantly larger than’.
session is started. A timeout of 30 minutes is used in many commercial and scientific systems (Cooley et al., 1999), including Fu et al. (2002) and Hay et al. (2004). All requests for other pages than HTML pages are removed.

We ignore sessions that are with high probability created by bots. These include sessions in which the bots have identified themselves in the agent field and sessions with extreme statistics. Sessions with more than 100 requests or an average time between two requests of less than 1 second or more than 6 minutes are called extreme.

After the sessions are restored, the pages in the sessions are classified into target pages and auxiliary pages. Auxiliary pages do not contain information that is interesting for the user, but only facilitate navigation. Several methods exist to determine whether a page is a target for a user, but most of the methods rely on domain-specific characteristics of the pages or on manually created page categories. For instance, in the WUM method (Spiliopoulou and Pohle, 2001) the pages of a site are manually split into pages that contain the content that the site wants to offer and auxiliary pages that facilitate browsing. Only pages of the first category qualify as potential targets. When no domain knowledge is available, the only available information about a user’s interest in a page is the time the user spent reading the page.

We use the time-based classification method described in Cooley et al. (1999). All pages with a reading time longer than or equal to a reference length are marked as targets. The other pages form the paths to the targets. Reading time is computed from the time difference between two consecutive requests. No reading time can be computed for the last page of a session. These pages are always classified as non-target pages. Classifying them as targets would mean that more navigation steps are seen as mistakes, which can cause links to be incorrectly classified as weak (see Section 5.4.2).

As reference length we use the median reading time of the hierarchy’s terminal pages (a terminal page is a page that does not have any subitems in the menu hierarchy). This means that we make the assumption that 50% of the times that a user views a terminal page, this page is a target page. The rationale behind this percentage is that target pages are content pages to which a user pays more than usual attention. This percentage falls in the range of optimal reference times (40-70%) that is found in the experiments of Fu et al. (2002).

5.4.2 Detection of unclear, covered and overlapping link descriptions

To evaluate link descriptions in a hierarchical menu, the menu is first divided in menu fragments (Figure 5.3 (i)). A menu fragment consists of one non-terminal node and its direct children. An example of a fragment from the menu in Figure 5.2 is shown in Figure 5.4.

The evaluation is performed fragment by fragment. For each fragment, we count in the log files how many times a user went from one page in the fragment to another page in the fragment. We count both steps from the parent node to one of its children and steps from child nodes to other child nodes. For each step, we determine which target page the user was looking for when he made the step. We assumed that this

---

2 The term ‘auxiliary page’ is introduced in Cooley et al. (1999). In other research these pages are sometimes referred to as ‘index pages’ (e.g. Fu et al., 2002).
Chapter 5: A semi-automatic usage-based method for improving hyperlink descriptions

Figure 5.4: Example fragment from the menu in Figure 5.2.

is the first target page that a user would reach. Clicks on links that led to the user’s target we call correct clicks. Clicks on other links, we call incorrect. For example, a user navigating in the menu of Figure 5.2 could have created the following trace:

Home (2) → Contact (3) → Home (1) → People (1) → Home (2) → Toolkit (46) →
Home (3) → Project (2) → Funding (105) → Project (0)

Here the letters denote the pages that the user has visited and the numbers between brackets denote reading times. The user’s target pages are shown in bold. In this example we see that the user clicked two incorrect links (Contact and People) and one correct link (Toolkit) before reaching target page Toolkit. He clicked no incorrect links before he correctly selected page Project which led to target page Funding.

The numbers of clicks within the fragment are counted in all user sessions. The total numbers of clicks are stored in a matrix. We call this matrix a confusion matrix as it shows how often users confused links with other links (Figure 5.3 (ii)). An example of a confusion matrix is shown in Figure 5.5.

<table>
<thead>
<tr>
<th>Clicked link</th>
<th>Contact</th>
<th>People</th>
<th>Toolkit</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>100</td>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>under</td>
<td>40</td>
<td>86</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Toolkit</td>
<td>0</td>
<td>6</td>
<td>90</td>
<td>2</td>
</tr>
<tr>
<td>Project</td>
<td>9</td>
<td>10</td>
<td>12</td>
<td>61</td>
</tr>
</tbody>
</table>

Figure 5.5: A confusion matrix with example frequencies for the menu fragment in Figure 5.4. Clicks on correct links are shown in bold.

The link descriptions in a menu fragment are evaluated using the corresponding row in the confusion matrix. Strong descriptions are characterized by a usage pattern where the correct link is chosen frequently while the incorrect links are chosen infrequently. In other words, people are able to select the links that lead to their targets. Large numbers of clicks on incorrect links indicate weak link descriptions.

Formally, the values in the confusion matrix are compared to a background model that represents the usage of links with good descriptions. According to this model,
users with a target in the page set of a link with a good description, click this link with at least probability $\psi$. The incorrect links have uniform probabilities of being chosen. Links with usage patterns that comply with the background model have strong descriptions. Deviations from the background model are used to assign links to the various problem classes. By default, the value of $\psi$ is set at the median of the observed probabilities of all correct links in a menu:

$$\psi = \text{Median}_{L\in\text{menu}} \left( \frac{\text{correct}_L}{\text{correct}_L + \text{incorrect}_L} \right)$$

where $\text{correct}_L$ is the number of times link $L$ was selected while the target was in the page set of $L$ and $\text{incorrect}_L$ is the number of times another link was selected while the target was in the page set of $L$.

Not every deviation from the background model signals a weak link as small deviations can result from chance. To determine whether a deviation is significant, we use a statistical test. For each row in the confusion matrix we compare the number of clicks on the correct link to the number of clicks on incorrect links by means of a binomial test (Figure 5.3 (iv)). If the proportion of clicks on the correct link is significantly lower than the expected probability $\psi$, the description of the link is marked as weak.

The subcategory of a weak link is determined by comparing the frequencies of the clicks on the incorrect links (v). For each incorrect link we compute with a binomial test whether the link has a significantly higher relative frequency than $1/i$, where $i$ is the number of incorrect links. If the frequency is too high, the correct link is covered by the incorrect link. If none of the incorrect links have a significantly high frequency, the description of the correct link is unclear.

When all links in a fragment have been evaluated, we check for overlapping links. For each pair of links we see whether they are covering each other (iii). Mutually covered links are called overlapping.

We illustrate this procedure by inferring the two example problems in the toy menu of Figure 5.2. We assume that the click frequencies of the first fragment of this menu are as shown in the matrix of Figure 5.5. $\psi$ is set at 0.88. We start with the evaluation of link Contact. The first row of the confusion matrix shows that in total there were 109 (100+6+2+1) occasions in which users with a target in the page set of Contact made a navigation step within the menu fragment. 100 of these users selected link Contact. The binomial test shows that 100/109 is not significantly lower than the expected value 0.88. Therefore, we conclude that Contact is a strong description. In 86 of the 127 occasions where the target was located under People, People was chosen. The test (iv) indicates that this is significantly lower than 0.88 and we conclude that the description People is weak. Next, we compare the frequencies of the clicks on incorrect links: 40, 0 and 1. The relative frequency of link Contact (40/41) appears to be significantly higher than $1/3$, which indicates that surprisingly many people with a target in the page set of People click link Contact. Therefore, we say that the description People is covered by the description Contact. Testing the third row of the confusion matrix indicates no problems with the description of link Toolkit. Link Project is chosen on only 61 out of 92 occasions, which is significantly low. If we compare the frequencies of the clicks on the
incorrect links, we find that none of them is too high. Consequently, the description Project is classified as unclear. In this example, none of the links are overlapping.

5.4.3 Detection of complete and partial problems

The next step in the classification process is to determine the target pages that the users were trying to reach, when they made the navigation mistakes found in the previous section. In particular, we determine whether these targets are located mainly in certain parts of the underlying menu tree or divided evenly over the underlying menu tree. For example, in the previous section we found that the description of link Project was unclear to many users. We will now determine whether Project was mainly unclear to the users who were looking for Partners, for Funding, or for Downloads or that Project was unclear to all of these users.

To further classify the problem of a weak link L, we look at all menu items that are in the menu hierarchy located directly or indirectly under L (L’s subitems). For each subitem we count the number of times users with targets in the page set of the subitem correctly clicked link L and how many times they made the navigation mistake corresponding to the problem. For instance, we count how often users with target Partners correctly clicked Project and how often these users incorrectly clicked another link. We test, again with a binomial test, whether the proportion of mistakes of users with a target in the page set of the subitem is significantly larger than the proportion of mistakes of all users with a target in the page set of L (Figure 5.3 (vi)). If users with a target in a subitem’s page set made significantly more mistakes, the subitem is called a problematic subitem.

Problems for which problematic subitems are found relate to only part of the users, namely the ones with targets in the page sets of the problematic subitems. Therefore, these problems are called partial problems. Problems for which no problematic subitems can be found are classified as complete problems. Note that items that form terminal nodes in a menu hierarchy can only have complete problems, because they do not have any subitems.

To demonstrate this step of the menu evaluation process we will complete the classification of the two problematic links found in the example fragment from the previous subsection. First, we classify the problem ‘covered_by(Person, Contact)’. For all 86 times when a user had a target under Person and correctly clicked Person, we determine under which subitems the target was located. Similarly, we determine where the target was located for the occasions in which Contact was clicked instead of Person. We assume the frequencies are as shown in Figure 5.6(a). For each subitem we compute the proportion of the times the problem occurred. For instance, for subitem Smith the problem proportion is 28/(56+28). The binomial test shows that this proportion is not significantly lower than the problem proportion of item Person as a whole (40/(86+40)). Thus, Smith is a not problematic subitem. If we do the same analysis for Email we find that its problem proportion is significantly high. Consequently, Email is marked as a problematic subitem. The other subitems are unproblematic. Because we found a problematic subitem, link Person is classified as partially unclear.

The next problem is ‘unclear(Project)’. For subitem Partners we count how many
Table 5.1: Example subitem frequencies and problem proportions for the problems (a) covered_by(People, Contact) and (b) unclear(Project). Problem proportions of problematic subitems are shown in bold.

<table>
<thead>
<tr>
<th>Clicked link</th>
<th>Problem proportion</th>
</tr>
</thead>
<tbody>
<tr>
<td>People</td>
<td>86</td>
</tr>
<tr>
<td>Contact</td>
<td>40</td>
</tr>
<tr>
<td>Smith</td>
<td>56</td>
</tr>
<tr>
<td>Johnson</td>
<td>30</td>
</tr>
<tr>
<td>Email</td>
<td>28</td>
</tr>
<tr>
<td>CV</td>
<td>28</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clicked link</th>
<th>Problem proportion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>61</td>
</tr>
<tr>
<td>not Project</td>
<td>31</td>
</tr>
<tr>
<td>Partners</td>
<td>10</td>
</tr>
<tr>
<td>Funding</td>
<td>10</td>
</tr>
<tr>
<td>Downloads</td>
<td>41</td>
</tr>
</tbody>
</table>

Figure 5.6: Example subitem frequencies and problem proportions for the problems (a) covered_by(People, Contact) and (b) unclear(Project). Problem proportions of problematic subitems are shown in bold.

5.4.4 Presentation of the problems

The confidence level that is used in the statistical tests determines how many links are called weak. Confidence levels usually vary between 0.9 and 0.999, but the exact value depends on how sure a webmaster wants to be before making changes. To make it easier for a webmaster to choose a sensible confidence level, we show with each identified problem the maximal level at which the problem is found. These levels can be determined by running the algorithm a number of times with various confidence levels. In the final presentation the problems are ordered from most certain to least certain. The webmaster can start at the top and stop when he feels he runs across too many unimportant problems.

As we cannot require that webmasters know our classification, we need to express our findings in normal language. For each problem class we formulated a sentence that explains the corresponding navigation mistake. The problematic link descriptions are filled in in these sentences automatically. In addition to problem statements we provide the solutions from Table 5.1. Figure 5.7 gives an example of a report as it is
Chapter 5: A semi-automatic usage-based method for improving hyperlink descriptions

Problem 1 (confidence 0.9999):
Many users who are looking for information under link People first open link Contact. These users are mainly searching for page Email.

Possible solutions:
- Make the description of link Contact narrower, so that it no longer covers page Email.
- Move or copy the link to page Email from People to Contact.

Problem 2 (confidence 0.9999):
Many users who are looking for information under link Project open another link first.

Possible solutions:
- Give link Project a new description that contains only terms that are known to the users and not too general.

Figure 5.7: Example of a report as it is shown to a web master.

sent to webmasters containing the two problems of the example fragment.

5.4.5 Computational complexity

Computation time is not a major issue for the link description classification algorithm, as the algorithm does not require a webmaster to pay attention while running. However, log files of even moderately sized sites are often extremely large, so that scalability is essential. In this section we discuss the time and space requirements of the link description classification algorithm.

To preprocess the log data the algorithm needs to go through the log files once. Consequently, the time complexity of the preprocessing phase is $O(L)$, where $L$ is the number of page requests in the raw log files. Building the confusion matrices requires one pass through the preprocessed logs, which requires $O(l)$ time, where $l$ is the number of requests in the preprocessed logs. The memory requirements to store the matrices are linear in the number of menu fragments and quadratic in the number of links in one fragment. This is maximally $O(n.b^2)$, where $n$ is the number of links in the menu and $b$ is the maximal number of links in one fragment (the menu’s breadth).

To classify the link descriptions in a menu as weak or strong, the algorithm first goes through all rows in the confusion matrices to determine $\psi$. The time needed for this operation is linear in the number of rows in the matrices which is equal to the number of links in the menu: $O(n)$. Subsequently, the algorithm passes through all rows again to determine which links are weak. For each weak link it checks all other links in the fragment to classify the link as covered or unclear. The time needed for this operation is linear in the number of weak links and the number of links per menu fragment: $O(w.(b-1))$, where $w$ is the number of weak link descriptions. Overlapping links are found by checking all pairs of weak links: $O(w^2)$. The space requirements are
5.5 Evaluation

\( O(n.b^2 + w) \), as the algorithm stores the confusion matrices and a list of weak links.

The algorithm passes through the preprocessed logs once more to determine the problem proportions of the subitems of the weak links. To decide which subitems are problematic, the algorithm looks at all problem proportions. The time complexity of this phase is linear in the size of the preprocessed logs, the number of weak links and the number of subitems under each link. This is smaller than \( O(l + w.n) \), because the number of subitems under a link is always smaller than the number of links in the menu \( (n) \). The space complexity of this phase is maximally \( O(2.w.n) \), as the algorithm stores for all subitems of weak links the numbers of correct and incorrect clicks. During this phase the confusion matrices are no longer needed.

In total, the time complexity of the link description algorithm is:

\[
O(L + 2l + 2n + w(b - 1 + w + n))
\]

The maximal space complexity of the algorithm is:

\[
O(max(n.b^2 + w, 2w.n))
\]

Thus, the total time requirements are linear in the size of the log files. This indicates that the algorithm will scale well to very large log files. The fact that the time and space complexity are (in the worst case) quadratic in the size of the menus is not problematic either, as in practice the size of a hierarchical menu seldom exceeds a few thousand links.

When the log files are very large, computation time can be further reduced by collecting all information needed for the whole classification process in one pass through the logs. However, this comes at the cost of higher memory requirements. With this modification the memory requirements become \( O(n.b^2 + n^2.b) \).

To give an indication of the running time of the algorithm in practice, we measured the time needed to analyze the menus of the sites that are used in the experiments in Section 5.5. The smallest site that was used in the experiments is the Reumanet site with 86 links in the menu and 56,463 log sessions (see Table 5.2). Apart from preprocessing, the analysis took for this site less than three minutes. The largest web site, the Leiden site, has both much larger log files (1,150,091 sessions) and a much larger menu (543 links). To analyze this site the algorithm needed about 5 hours.

5.5 Evaluation

We evaluated the method for improving link description on the menus of three Dutch web sites. The first site is the SeniorGezond site (SG) (SeniorGezond, 2007). It is aimed at elderly people and contains information about the prevention of falling accidents. The second site is called Reumanet (RN) (Reumanet, 2007) and contains information about rheumatism. The last site is the official site of the local government of the city of Leiden (Gemeente Leiden, 2007). The sites provide various navigation means, but in this study we restricted ourselves to the links in the hierarchical menus.

For each site we extracted the menu and collected log files. Properties of the menus and logs can be found in Table 5.2. At the time we extracted the menu of the Leiden
large parts of the menu consisted of uniform lists of links that functioned as archives. For instance, some lists contained minutes of all meetings of certain committees ordered and named by date. We excluded these parts of the menus from the evaluation. Without these lists the menu of the Leiden site consisted of 543 links.

The last two columns of Table 5.2 show the parameter values used in the experiments. The thresholds for determining target pages (the reference lengths) are set at the median reading times of the hierarchies’ end pages (see Section 5.4.1). The expected proportions of correct links ($\psi$) are set at the median of the proportions of all correct links (see Section 5.4.2).

<table>
<thead>
<tr>
<th>Site</th>
<th>Log period</th>
<th>Number of sessions</th>
<th>Menu size</th>
<th>Reference length</th>
<th>$\psi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG</td>
<td>29 months</td>
<td>150,568</td>
<td>92 links</td>
<td>14 seconds</td>
<td>0.75</td>
</tr>
<tr>
<td>RN</td>
<td>29 months</td>
<td>56,463</td>
<td>86 links</td>
<td>13 seconds</td>
<td>0.83</td>
</tr>
<tr>
<td>LE</td>
<td>26 months</td>
<td>1,150,091</td>
<td>543 links*</td>
<td>18 seconds</td>
<td>0.88</td>
</tr>
</tbody>
</table>

Table 5.2: Properties of the sites and log files that were used for evaluation. *These are the links that were used in the evaluation study. The whole menu is larger (see text).

During the first phase of the evaluation, we assessed how useful webmasters think the method is. We applied the algorithm to the logs and menus of the three web sites. The identified problems and solutions were shown to the webmasters of the sites. We interviewed the webmasters while they went through the lists of problems that were generated by the system. For each problem we asked them whether they thought the identified problem was really a problem that called for an adaptation in the menu. When they felt an adaptation was needed, we asked them to choose an appropriate solution from the provided solutions or come up with an alternative solution. Moreover, when they chose a solution that involved changing link descriptions, they had to provide the new descriptions. After the interviews we counted how many identified problems the webmasters found relevant and for how many problems they were able to select a solution.

The second phase of the evaluation concerned the influence of the solutions that were selected by the webmasters on user navigation. We implemented the selected solutions in offline versions of the sites’ menus. In a user experiment these menus were compared to the original menus.

Participants in the user experiment performed tasks in an interface that was developed for this purpose. The interface is shown in Figure 5.8. Each task started with a description of the information needs of a fictitious person (Figure 5.8(a)). When a participant had read the description, he or she pushed a button. The interface then showed a menu that was partly opened (Figure 5.8(b)). The links in the deepest menu fragment that was visible were preceded by option buttons. The participants were asked to click the button preceding the link that they would open first when they were the person from the task description. When they selected one of the buttons, the choice was recorded and the description of the next task was shown.
Each participant performed tasks in the menus of all three sites. For each site they saw either the original menu or the adapted menu. The version of the menu that they used was determined randomly at the start of the experiment and remained the same throughout the session. The participants did not know whether they were using an original or an adapted menu. In fact, they didn’t even know that there was more than one version.

The participants started the experiment with a practice task to familiarize themselves with the interface. After that, the order of the tasks was random and different for all users. However, we made sure that users always performed tasks in higher menu layers before they performed tasks in lower menus. In this way, the participants had not yet seen which links were located under the links from which they had to choose.

When we wrote the task descriptions we asked the webmasters to describe typical usage scenarios. In addition, we looked at target pages of users in the log files to determine with what kinds of questions users visited the sites. In this way, we tried to create tasks that realistically reflected the needs of real users. We avoided the use of terms in the task descriptions that were also present in the link descriptions in the menus, so that the participants’ choices would not be biased by our formulation.
For each task we defined a number of target pages: the pages that answered the information needs. All tasks had target pages that were located in parts of the menus that were modified by the webmasters. Examples of task descriptions can be found in Appendix B.

The main goal of the link evaluation method, is to reduce the number of navigation mistakes. To see to what extent this goal is met, we measured the number of mistakes made in the original and the adapted menus. Some adaptations introduced an extra layer in the menu hierarchy. In these cases the participants had to make a choice in both layers (the higher layer first). Only when both choices were correct, we counted the task as performed correctly.

Besides the users’ choices, we also recorded the time they needed to make a choice. The interface allowed us to distinguish between the time a user spent reading a task description and the time he needed for his choice. A timer was started when a user opened the menu and ended when he submitted his choice. To reduce the influence of users who were distracted from the tasks, we removed choice times longer than 1 minute from the analysis.

5.5.1 Results of the study with webmasters

On all three sites the link evaluation algorithm found many problems. Table 5.3 shows that 16-27% of all links were classified as weak. Thus, at about one fifth of the links users often make incorrect choices. This suggests that improving weak descriptions can have a large effect on the efficiency of the users’ navigation.

<table>
<thead>
<tr>
<th>Site</th>
<th>Number of links</th>
<th>Identified problems number</th>
<th>percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG</td>
<td>92</td>
<td>15</td>
<td>16.3%</td>
</tr>
<tr>
<td>RN</td>
<td>86</td>
<td>23</td>
<td>26.7%</td>
</tr>
<tr>
<td>LE</td>
<td>543</td>
<td>88</td>
<td>16.2%</td>
</tr>
<tr>
<td>Total</td>
<td>721</td>
<td>126</td>
<td>17.5%</td>
</tr>
</tbody>
</table>

Table 5.3: The number of links on each site, the numbers of problems that are identified by the algorithm and the percentage of links classified as weak.

Table 5.4 shows how many problems from each class were found. At the SeniorGezond site and the Reumanet site the algorithm mainly found complete problems, while many partial problems were found at the Leiden site. The SeniorGezond and Reumanet menus contain many terminal nodes, that can only have complete problems (see Section 5.4.3). Moreover, for these sites less log data was available than for the Leiden site, so that getting significant deviations was more difficult. This also explains that at these sites a larger proportion of the descriptions was classified as unclear.

Only one case of overlapping links was found. This case concerned the links ‘service points’ and ‘products and services’ on the Leiden site. Most problems were classified as covered even though they had overlapping meanings. This happened for
5.5. Evaluation

instance when one of the descriptions was more general than the other or when one of the descriptions contained more common terms. Only when descriptions were overlapping in meaning and had equivalent terms we found navigation mistakes in both directions.

As shown in Table 5.3, 88 problems were found in the menu of the Leiden site. For a webmaster who wants to improve the menu of his site, it is doable to look at all 88 problems. However, for the interview with the webmaster this number was too high, as the time of the webmaster was limited. Therefore, we restricted the evaluation for this site to the 21 problems that concerned non-terminal menu items. For the other two sites the algorithm found much smaller numbers of problems, so that all problems could be evaluated.

In total, the webmasters found that 38 of the 59 evaluated problems were really problematic, as can be seen in Table 5.5. This indicates that in general the algorithm is capable of finding relevant problems. On the SeniorGezond site and the Leiden site the majority of the problems was assessed as relevant. Only on the Reumanet site, many irrelevant problems were found. Reasons why the webmaster did not find these problems relevant are discussed below. Problems concerning non-terminal items were more often found relevant than problems with terminal items (respectively 76% and 48%).

<table>
<thead>
<tr>
<th>Site</th>
<th>Problems shown to webmasters</th>
<th>Relevant problems</th>
<th>Solutions found</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>number</td>
<td>percentage</td>
<td>number</td>
</tr>
<tr>
<td>SG</td>
<td>15</td>
<td>93.3%</td>
<td>9</td>
</tr>
<tr>
<td>RN</td>
<td>23</td>
<td>34.8%</td>
<td>6</td>
</tr>
<tr>
<td>LE</td>
<td>21</td>
<td>76.2%</td>
<td>16</td>
</tr>
<tr>
<td>Total</td>
<td>59</td>
<td>64.4%</td>
<td>31</td>
</tr>
</tbody>
</table>

Table 5.5: The number of problems shown to the webmasters, the number and percentage of the shown problems that the webmasters assessed as relevant, the number and percentage of relevant problems for which the webmasters found solutions.

For 81.6% of the relevant problems the webmasters could come up with a satisfying solution (Table 5.5). For 15 problems they chose one of the solutions proposed by the algorithm and for 19 problems they brought forth a solution of their own\(^3\) (Table 5.6).

\(^3\)The total of these two figures is larger than the number of solutions, as for some problems solutions were chosen that consisted of multiple actions, such as merging items and modifying link descriptions.
Most solutions involved changing either the structure of the menu or changing the link descriptions. However, in a few cases the webmasters chose a third type of solution: changing the structure of the content. When the algorithm found that users could not distinguish between items, the webmasters chose to merge the texts. Naturally, this solution was chosen only when the individual texts were very short. Besides helping to modify the menus, in some cases the findings of the algorithm also inspired the webmasters to improve other navigation means. During the interviews they came up with ideas to change in-text links and to add keywords to the indexes of the sites’ search engines.

In general, the webmasters were very positive about the approach. They felt that the lists of problems provided useful insights in the behavior of the users and could really help them to improve the site. Moreover, one of the webmasters felt that the findings backed up her own ideas about the site. She reported that one of the problems already came up during a pilot study in which the site was evaluated. Keeping the structure as it was, was a political decision.

The information that the algorithm extracts appears to be well-chosen. The more information the problem classes gave, the more they were appreciated. An illustration of this fact came up during the interview with the webmasters of the SeniorGezond site. When the webmasters encountered a complete problem, they asked themselves, which targets these users were looking for. This question was raised before they knew the algorithm could sometimes identify problematic subitems. Similarly, when links were unclear, the webmasters often wondered which items the links were confused with.

About one third of the identified problems were not found problematic. The webmasters gave various reasons for this. First, there were some problems for which the webmasters could not think of a reason why users would make the navigation mistake. For example, on the Reumanet site, the algorithm found that many people clicked on ‘Patient association’ when they were looking for information under ‘Accessibility aids’. The webmasters assumed that these problems resulted from noise in the data. Second, in some cases the webmasters found that the problems were sufficiently solved by in-text links. The target pages were not located under the selected menu items, but users could reach them through the in-text links on the pages under the selected items. The webmasters of the SeniorGezond site gave a third reason for not wanting to change certain link descriptions: they wanted to educate their users. They liked the fact that users did not know the meaning of certain descriptions. They thought that

<table>
<thead>
<tr>
<th>Site</th>
<th>Source of solution</th>
<th>Solution type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>proposed</td>
<td>other</td>
</tr>
<tr>
<td>SG</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>RN</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>LE</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>Total</td>
<td>15</td>
<td>19</td>
</tr>
</tbody>
</table>

Table 5.6: Numbers of solutions of various types that are selected by the webmasters.
5.5. Evaluation

these descriptions could trigger the users’ curiosity and make them open more pages, so that the users would learn more about the prevention of falling accidents. For the same reason, they did not want to move certain items, even though they believed users searched for these items at other locations. For instance, the algorithm found that many users searched for ‘mobility scooters’ under ‘walking aids’. The webmasters of SeniorGezond did not want to place it there, because they wanted to teach people that mobility scooters are not walking aids.

A particularly interesting reason why problems were not relevant, was given by the webmaster of the Reumanet site. Most of the irrelevant problems for this site (10 out of 15) concerned terminal items that were part of lists of accessibility aids. The webmaster explained that users usually do not search these lists for particular items, but look at all available aids to see whether there is anything they can use. In these cases, our assumption that users have target pages does not hold, so that the algorithm fails to produce sensible results. This insight suggests that the algorithm can be improved by first dividing the user population in users who are searching for particular information and users who are browsing.

5.5.2 Results of the user experiment

35 participants took part in the user study. 3 of the participants performed not all tasks, but the data from the tasks that they completed could still be used. None of the participants were familiar with the contents of the sites. 15 participants were computer science students. The background of the others varied. Table 5.7 shows the number of times a user performed a task on the two versions of the sites.

<table>
<thead>
<tr>
<th>Site</th>
<th>Number of different tasks</th>
<th>Number of times a task is performed in original menu</th>
<th>Number of times a task is performed in adapted menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG</td>
<td>12</td>
<td>176</td>
<td>214</td>
</tr>
<tr>
<td>RN</td>
<td>11</td>
<td>196</td>
<td>167</td>
</tr>
<tr>
<td>LE</td>
<td>15</td>
<td>267</td>
<td>222</td>
</tr>
<tr>
<td>Total</td>
<td>38</td>
<td>639</td>
<td>603</td>
</tr>
</tbody>
</table>

Table 5.7: Numbers of tasks formulated for the various sites and the number of times the tasks were performed in the original and adapted versions of the menus.

On all three sites the participants selected in the adapted menus more correct links than in the original menus, as shown in Table 5.8. In the adapted menus 10-28% less mistakes were made, which is a significant improvement. This indicates that the adaptations that the webmasters chose on the basis of the outcomes of the algorithm really improved the menus.

We measured for each task individually how well the task was performed in the original and the adapted menus. Table 5.9 shows the results per solution type. Tasks that addressed parts of the sites where structural adaptations were made, showed the largest improvements (25%), but the other solution types also gave considerable improvements. The only type of adaptation that did not help was changing the order
Chapter 5: A semi-automatic usage-based method for improving hyperlink descriptions

<table>
<thead>
<tr>
<th>Site</th>
<th>Original correct</th>
<th>Adapted correct</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG</td>
<td>0.52</td>
<td>0.61</td>
<td>0.10 (z=-1.89, p=.003)</td>
</tr>
<tr>
<td>RN</td>
<td>0.41</td>
<td>0.59</td>
<td>0.17 (z=-3.30, p=.001)</td>
</tr>
<tr>
<td>LE</td>
<td>0.34</td>
<td>0.62</td>
<td>0.28 (z=-6.18, p=.001)</td>
</tr>
<tr>
<td>Total</td>
<td>0.41</td>
<td>0.61</td>
<td>0.20 (z=-6.94, p=.001)</td>
</tr>
</tbody>
</table>

Table 5.8: Proportion of tasks in which the correct choices are made in the original and the adapted menus. Significance is tested with a two sample z-test.

of the items in a menu (a structural change). Apparently, in our study, users did not favor the top items in a list over the other items. If this also holds for real web users, this can be useful information for webmasters. However, it can also be an artifact of our experimental design: the participants were focusing entirely on making the correct choice, while real web users might be more inclined to choose quickly and try out various options. More research is needed to determine which explanation is true.

We measured the time users needed to perform the tasks, to rule out the possibility that the efficiency that is gained by making less mistakes is cancelled out by slower choice times. For tasks that included more than one choice, we used the total time needed for all choices. On average users spent 15.66 seconds thinking in the original menus. In the adapted menus they needed 15.73 seconds. Thus, the adaptations did not significantly slow down the navigation and were only beneficial.

<table>
<thead>
<tr>
<th>Solution type</th>
<th>Original correct</th>
<th>Adapted correct</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>descriptions</td>
<td>0.39</td>
<td>0.53</td>
<td>0.14 (z=-3.11, p=.001)</td>
</tr>
<tr>
<td>content</td>
<td>0.46</td>
<td>0.62</td>
<td>0.16 (z=-1.61, p=.05)</td>
</tr>
<tr>
<td>structure</td>
<td>0.42</td>
<td>0.67</td>
<td>0.25 (z=-6.36, p=.001)</td>
</tr>
<tr>
<td>Total</td>
<td>0.41</td>
<td>0.61</td>
<td>0.20 (z=-6.94, p=.001)</td>
</tr>
</tbody>
</table>

Table 5.9: Proportion of tasks in which the correct choices are made in the original and the adapted menus. Significance is tested with a two sample z-test.

5.6 Conclusions and discussion

In this chapter we presented a method that helps a webmaster to understand at which points in a navigation menu users make incorrect choices. Based on these insights the algorithm provides suggestions for improving the menu.

Evaluation of this method on three real websites gave promising results. The webmasters of the sites felt that the analyses made by the algorithm were very helpful. 64% of the problems that the algorithm identified were assessed as relevant. In a user study the usefulness of the method was confirmed: in menus that the webmasters had adapted on the basis of the analyses users made 10-28% less mistakes than in the original menus. These findings lead to the conclusion, that the algorithm can effectively
help webmasters to make navigation menus more efficient.

The current work demonstrated how link descriptions can be evaluated for a user population as a whole, but the algorithm can be applied equally well to smaller user groups. In domains with heterogeneous user populations, it can be necessary to create different descriptions for different groups of users or for users in different contexts. This can be accomplished by first clustering the log data in a number of user clusters with similar navigation patterns or similar contextual properties. Several methods have been developed for this purpose (e.g. Mobasher et al., 2002; Hay et al., 2004). Once the data is clustered, a link description analysis is performed for each cluster using only the sessions of the users from the cluster. In this way, it is possible to find link descriptions that are clear to one group of users but unclear to others.

From usage data we can compute how descriptive links are in their current context, but we cannot evaluate link descriptions per se as descriptiveness is highly context dependent. For instance, a description ‘seal’ is perfectly descriptive among ‘stamp’ and ‘envelope’, but when a link named ‘walrus’ is added it becomes unclear. Consequently, when new pages are added to the site, a new link description analysis needs to be performed for the modified branches.

The evaluation study involved a limited number of search tasks. Even though we made efforts to create tasks that were natural in the domains, studies using search tasks can never cover the information needs of all users of a site. To evaluate the effects of a new menu on the whole user population, the menu must be placed online, so that it is used by real users. After a while, the influence of the new menu can be determined from the user logs. A disadvantage of this method is that users who visited the site before will have to get used to the new menu. This could lead to a temporary drop in performance which biases the results towards keeping the menu as it was.

In the evaluation study we found that users sometimes do not search for specific target pages, but rather browse the site to see whether it contains anything of interest. The current version of the link evaluation algorithm does not take this into account and treats browsing behavior as noise. As a result, it produces poor results for parts of the menus where many users browse. This leads to the conclusion, that the algorithm can probably be improved by filtering out logs of browsing users. More research is necessary to determine what criteria can be used to distinguish between browsing and searching users.

Another topic for further research are methods to infer higher order problems from the problem classes detected in this chapter. For instance, if a large portion of the links in a menu fragment appears to be unclear, we might infer that the categorization of the menu fragment represents an inappropriate perspective. For example, if most users search movies by actor, then a menu consisting of a list of genres will lead to many navigation mistakes. In this case it is probably more effective to restructure the whole menu instead of updating individual links.

We presented a method for automated analysis of logfiles to identify menu items that cause navigation errors and to propose improvements of the descriptions or the structure of these items. The method is very generally applicable because it only needs data that is available in log files and it was proved to give useful advice to webmasters in practice.