Capturing the needs of amateur web designers by means of examples

Vera Hollink **Maarten van Someren** **Viktor de Boer**
University of Amsterdam
Kruislaan 419, Amsterdam, The Netherlands
{V.Hollink,M.W.vanSomeren,V.deBoer}@uva.nl

Abstract

Many sites are created by people who lack professional training in web design. We present 'SiteGuide', a tool that helps amateur web designers to decide which information will be included in a new web site and how the information will be organized. SiteGuide takes as input URLs of web sites from the same domain as the site the user wants to create. It automatically searches the pages of these example sites for common topics and common structural features. On the basis of these commonalities it creates a model of the user's needs. The model can serve as a starting point for the new web site. Also, it can be used to check whether important elements are missing in a concept version of the new site and, if necessary, to adapt the initial design.

1 Introduction

Even the smallest companies, institutes and associations are expected to have their own web sites. However, many small organizations lack the expertise to create high quality web sites themselves and neither can they afford to hire a professional web designer. Consequently, many sites are build by amateurs. For these inexperienced web designers building a site takes a lot of time and the resulting web sites are often of low quality.

Existing tools that support authoring of web sites, such as Abobe's Dreamweaver[1] and Microsoft's Frontpage[2], provide an environment in which sites can be created, but they do not help users to decide which topics need to be included in the site and how the content must be organized [Falkovych and Nack, 2006]. These tools are a solution for users who do not know HTML, but not for users who are not experienced web designers.

In this paper we present 'SiteGuide', a tool that helps a user to create a setup for the content and the structure of a new site. SiteGuide models the implicit needs and expectations of the user and gives the user advice on how to create a site that meets these requirements. The user does not need to enter his (or her) requirements explicitly. This would be undesirable, as beforehand designers of web sites often do not know exactly what they want. Moreover, learning the input format for the requirements can be a drawback of using such a system. Instead, SiteGuide offers a natural interface in the tradition of the web 2.0: the user can specify his requirements in the form of examples. The user provides some examples (typically 3 to 10) of web sites from the same domain as the one he wants to create. For instance, a user who wants to build a site for a small amateur soccer club enters URLs of web sites of other small soccer clubs that he likes. From the examples SiteGuide automatically deduces what the user has in mind and how this can be accomplished.

When a user enters a set of URLs in SiteGuide, the system overlays the sites in such a way that pages with similar topics or structures are mapped onto each other. The result of this process is a model of the sites that describes the features that the sites' pages have in common. For example, in the soccer club domain, a feature could be that all example sites contain information about youth teams or that pages about membership always link to pages about subscription fees. The model is presented to the user in human readable format and serves as a setup for the new web site. When the user has already created a first draft of his site, the draft is compared automatically with the model, so that missing topics or deviating information structures are revealed.

Reviewing example sites to get a better picture of what is needed is not a new idea. This step is commonly performed during the design of a web site [Newman and Landay, 2000]. However, until now this step had to be performed manually which is very time-consuming. Moreover, when a user goes through the examples by hand, there is a large probability that important features of the sites are overlooked. SiteGuide enables an efficient, thorough and structured review of a set of example sites.

In this paper we explain the algorithms behind the SiteGuide system. In addition, we present the results of a preliminary evaluation study, in which we apply SiteGuide to web sites from three different domains and evaluate the resulting example site models.

2 Related work

There are many tools available that allow users to create web sites without typing HTML. Tools such as Dreamweaver[1] and Frontpage[2] offer attractive graphical user interfaces that enable users to easily insert text, pictures, hyperlinks, forms, etc. However, as discussed, these tools do not offer any support to users who do not know which content or links they want to insert. Specialized authoring tools have been developed in the context of adaptive hypermedia, e.g. AHA! [De Bra *et al.*, 2007], MOT [Cristea and De Mooij, 2003] and VIDET [Armani, 2005]. With these tools users can specify relations between pieces of content that determine how the material is adapted to users with certain characteristics. When a user knows what

[1]http://www.adobe.com/products/dreamweaver
[2]http://office.microsoft.com/frontpage

the web site should look like and how the adaptive component must behave, these authoring tools enable him to implement the site in an efficient and intuitive way. However, again these tools do not help users to choose the appropriate pieces of content or the relations between them.

Web site optimization systems help owners of web sites to make their sites more efficient. These systems analyze the contents or the log files of a site and generate recommendations for improvement. A classic example is the PageGather system [Perkowitz and Etzioni, 2000], which generates index pages containing links to pages that are frequently visited together in a user session. Hollink et al. [2007] automatically analyze the log files of a site to discover patterns in the behavior of users who navigate through a menu. The patterns are used to optimize the efficiency of the menu. These and other optimization systems (see [Pierrakos *et al.*, 2003] for an overview) are of great value for the optimization of existing sites, but they do not offer support for the creation of new sites.

Ontology mapping refers to the task of finding concepts in one ontology that match concepts in another ontology. Most ontology mapping algorithms first compare the texts of the labels of the concepts and then use the relations between the concepts to refine the mappings (e.g. [Heß, 2006; Hu and Qu, in press]). This methodology is similar to the one followed in this work, where web pages play the role of concepts and hyperlinks are relations. However, there are a number of differences. First, labels of ontological concepts are usually much shorter than the texts of web pages. Second, ontology mapping aims to create one-to-one mappings between single concepts of two, or in rare cases three, ontologies. The topics that we try to identify can be spread over several pages and large numbers of web sites. Finally, ontological relations usually have clear semantics that restrict the allowed mappings. The meaning of hyperlinks is much less clear, so that we cannot use consistency checking to discover low quality mappings.

It is well known that users often cannot accurately formulate what they need. For this reason, systems for image and video retrieval allow users to specify their queries by means of examples: when a user enters an example image, the system retrieves similar images or videos (e.g. [Snoek *et al.* , 2004]). In text search engines example web pages can not be entered directly, but in some systems search results are accompanied by links called ‘more like this’ or ‘similar pages’ (e.g. [Spink *et al.*, 2000]). When one of these links is clicked the user receives a page with search results that are similar to the result that belongs to the link. This functionality is comparable to the SiteGuide system. In both cases the example-based approach helps users to formulate what they need: search engines focus on the needs of users who search for information and SiteGuide on the needs of web designers.

3 Problem definition

In this section we will describe the task of the SiteGuide system. There are two usage scenarios, which are in shown in Figure 1. In both scenarios the user starts the interaction by inputting the URLs of the entry pages of a set of example web sites. SiteGuide scrapes and analyzes the sites and captures their commonalities in a web site model. In the first usage scenario the model forms the end point of the interaction and is outputted in human readable form to the user. In the second scenario the user has already created a first draft version of his new site. He enters the URL

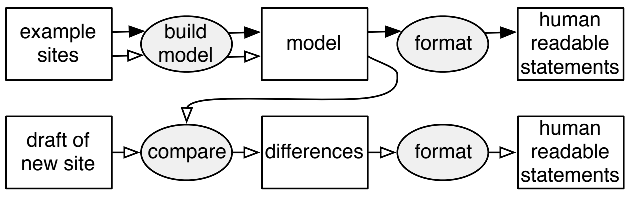

Figure 1: The two usage scenarios of the SiteGuide system. —► denotes the scenario 1. —▷ denotes scenario 2.

of the entry point of the new site in SiteGuide. SiteGuide compares the draft with the model of the example sites and outputs the differences.

Figure 2 shows the structure of an example site model. A model consists of a number of *topics* that represent the subjects that are addressed at the example sites. To communicate the model to a user, SiteGuide uses a subset of the following five properties to characterize a topic’s content and structural features:

- has keywords X
- has title that contains terms X
- has URL that contains terms X
- is pointed to by links with anchors that contain terms X
- is linked to or from topic Y

Here X is a list of terms and Y is another topic.

Besides the characterizing properties, a topic can have secondary features that describe how the topic is represented at the example sites:

- over how many pages the information on a topic is spread
- the number of incoming and outgoing links of the pages that address the topic
- examples of pages that address the topic

In the first scenario the model is outputted as a series of natural language statements. An example is shown in Figure 3(a). The output of scenario 2 are similar statements that describe the differences between the example sites and the draft (e.g. Figure 3(b)).

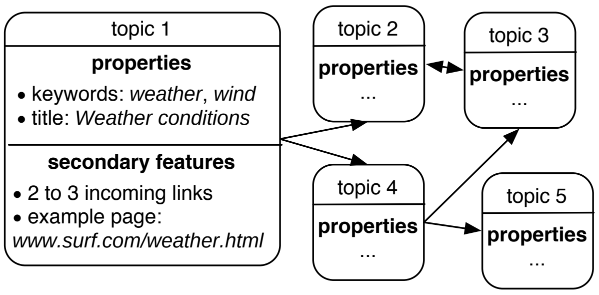

Figure 2: Example of an example site model. A model consists of topics that have properties and secondary features (only shown for topic 1). Frequently occurring hyperlinks between topics are denoted by arrows.

4 Method

SiteGuide creates a mapping between the pages of the various example sites. The mapping forms the basis of the example site model. In this section we first explain the format of the mapping. Then we describe how SiteGuide measures the quality of a potential mapping and how it finds the

a. All example sites have pages with keywords ‘lessons’, ’training’ (e.g. page ‘www.example.com/surfing_lessons.html’) that have a link to a page with the term ‘events’ in the title (e.g. page ‘www.example.com/upcoming.html’).

b. All example sites have pages with keyword ‘weather’ (e.g. page ‘www.example.com/weather.html’), but the new site does not.

Figure 3: Examples of output statements of SiteGuide for usage scenarios 1 (a) and 2 (b).

mapping that maximizes the quality measure. Finally, the generation of the example site model and the comparison between the model and a draft of the new site are discussed.

4.1 Mapping format

Figure 4 shows the format of a mapping between a set of example sites. The mapping comprises a number of mapping elements that represent topics. A mapping element consists of page sets. Each page set contains all pages from one example site that handle on the mapping elements’ topic. All pages of the example sites occur in at least one mapping element, but they can occur in multiple elements.

In the simplest case the page sets contain only one page, so that a mapping is created between individual pages. For example, a mapping element can contain for each site the page about surfing lessons. However, if on one of the sites the information about surfing lessons is split over several pages, these pages are placed together in a page set and mapped as set onto the other pages about surfing lessons. It can also happen that a mapping element does not contain page sets from all sites, because some of the sites do not contain information on the element’s topic. Pages occur in more than one mapping element, when they contain content about more than one topic. For instance, suppose that on one site the information about surfing lessons is on the same page as the information about membership fees, while on the other sites these topics are on distinct pages. In this case the combination page should occur in two mapping elements: one about lessons and one about fees.

4.2 Quality measure

The main task of SiteGuide is to find a good mapping between the example sites. Therefore, it must be able to estimate the quality of a potential mapping. Intuitively, the quality of a mapping depends on three criteria:

1. A mapping is better when the pages within one mapping element are more similar.

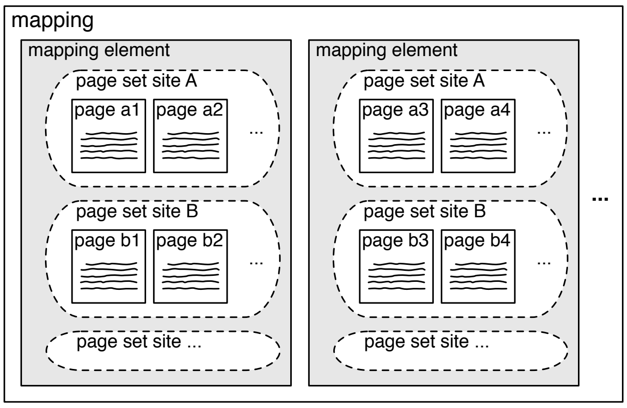

Figure 4: Format of a mapping between example sites.

2. A mapping is better when it maps the sites at a more detailed level.
3. A mapping is better when it connects more pages.

We formalize these criteria in a quality measure. The quality of a mapping is the average quality over all mapping elements in the mapping. The quality of an element is defined by the similarity between its page sets.

We identify five factors that contribute to the similarity of page sets. These factors correspond to the five characterizing properties mentioned in Section 3: similarity between texts of pages, between their titles, between their URLs, between the anchors of the links that point to the pages and between the pages’ places in the link structures. The quality of a mapping element is a linear combination of these similarities. The quality of element $\mathcal{M}$ in mapping M is:

$$quality(\mathcal{M}, M) = \sum_{sim_i \in Sims} (w_i \cdot sim_i(\mathcal{M}, M)) - c \cdot |\mathcal{M}|$$

Here $Sims$ are measures for the five types of similarity (explained below). The similarities are weighted with weighting parameters w_i, which add up to 1. c is a parameter. The term $-c \cdot |\mathcal{M}|$ substracts a fixed amount (c) for each of the $|\mathcal{M}|$ page sets in the mapping element. Consequently, a page set only improves the score of an element if it bears a similarity of at least c to the other page sets.

To determine the similarity between the texts of the pages in a mapping element, we compute for each page set in the element the average similarity to each other page set. Text similarity between two pages is expressed as the cosine similarity between the terms on the pages ([Salton and McGill, 1983]). This measure enables SiteGuide to identify the parts of the texts that pages have in common and ignore the site-specific parts. Stop word removal, stemming and $tf \cdot idf$ weighting [Salton and McGill, 1983] are applied to increase accuracy. The text similarity of an element is the sum of the text similarities of its page sets.

Anchor text similarity is the cosine similarity between the anchor texts of the links that point to the pages. In the computation of this similarity, the anchors of all links that point to a page are treated together as one document.

For the computation of page title similarity and URL similarity we use the Levenshtein distance [Levenshtein, 1966] instead of the cosine similarity. Levenshtein distance is more suitable for comparing short texts as it takes the order of terms into account and works at character level instead of term level.

The link structure similarity of an element is the proportion of the incoming and outgoing links that are mapped correctly. This means that if page a is mapped onto page b, then the children of a (the pages which a links to) must be mapped on the children of page b. Similarly, the parents of a must be mapped on the parents of b. The more children and parents are mapped correctly, the higher the link structure similarity.

Together the five similarity measures satisfy the criteria for a good mapping that we defined above. A mapping elements scores higher on the measures when the pages in its page sets are more similar (criterion 1). When the page sets contain a few, highly similar pages, the average similarity between the pages is higher than when large sets of pages are mapped onto large sets (criterion 2). Finally, similarity is maximized when an element contains page sets from more sites (criterion 3).

When the example sites differ strongly in the way the information is divided over the pages, a trade-off must be made between criteria 1 and 2. In this case similarities between individual pages are low, so that elements with small page sets will have low similarity. Creating elements with large page sets increases similarity, but decreases the number of elements. The quality measure enables SiteGuide to find the optimal trade-off.

4.3 Finding a good mapping

SiteGuide needs to find a mapping with a high similarity. A naive approach would be to list all possible mappings, compute for each mapping the similarity score, and choose the mapping with the highest score. Unfortunately, this approach is not feasible, as the number of possible mappings is extremely large. Each mapping element contains a subset of the pages of the example sites. A mapping is a subset of the possible mapping elements. As a result, the number of possible mappings is at most $2^{2^{ns}}$, where s is the number of example sites and n is the average number of pages per site.[3] To make the problem computationally feasible, we will use a heuristic approach that allows us to find a good mapping without considering all possible mappings.

We create an initial mapping in which each page occurs in exactly one page set and each page set contains exactly one page. In other words, the initial mapping is a one-to-one mapping between pages. The initial mapping is built incrementally. First, we create a mapping between the first and the second example site. For each two pages of these sites we compute the text, title, URL and anchor text similarity and weight these factors. In this stage we do not consider link similarity as link similarity cannot be computed for a mapping element in isolation but requires a complete mapping. The so called Hungarian Algorithm [Munkres, 1957] is applied to the two sites to find the one-to-one mapping with the highest similarity. Then, the pages of the third site are added to the mapping. We compute the text, title, URL and anchor text similarity between all pages of the third site and the already formed pairs of pages of the first two sites and again apply the Hungarian Algorithm. This process is continued until all example sites are included in the initial mapping.

The initial mapping is refined by means of a hill climbing method. In each step we try a number of modifications to the current mapping and compute the effect on its similarity score (including all five types of similarity). When a modification improves the score it is retained; otherwise it is undone. We keep trying modifications until we can not find any more modifications that improve the score with a sufficient amount.

Five types of modifications are used. Together, these modifications suffice to transform any mapping into any other mapping, so that, in theory, the optimal mapping can always be found. However, in practice this does not always happen, because the hill climbing algorithm can get stuck in a local maximum. The types of modifications are:

- Split a mapping element: each page set in the mapping element is placed in a separate mapping element.
- Merge two mapping elements: place all page sets of the elements in one large element. If the elements contain page sets from the same site, the sets are merged.
- Move a page from one mapping element to another mapping element.
- Move a page from a mapping element to a new empty mapping element.
- Copy a page from a mapping element to another mapping element.

[3]The actual number is a little lower, because we do not need to consider mappings with empty mapping elements or mappings with elements that are subsets of other elements.

These modifications can be applied to all mapping elements. To increase efficiency, we try modifications of elements with low similarity first.

There are various reasons why the hill climbing process can improve the initial mapping. First, page sets can be formed with more than one page and pages can be included in multiple page sets. Second, for the computation of the total similarity also link similarity is used. Finally, the hill climbing process can overcome possible negative effects of unfortunate choices made in the initialization phase as a result of the order in which the example sites are added.

4.4 From mapping to model

The next step is to transform the mapping into a model of the example sites. Each mapping element becomes a topic in the model. Topics are characterized by the five properties mentioned in Section 3. Which properties are used depends on the values of the various similarity measures. When an element scores high on text similarity, SiteGuide extracts keywords from the text. When URL gives a high similarity, terms from the URLs are used, etc.

To find the secondary features of the topics, SiteGuide analyzes the page sets and links of the corresponding mapping elements. It determines for each site over how many pages the information on a topic is spread and counts the number of incoming and outgoing links. In each element the pages that are most similar to the other pages in the element become the example pages for the topic.

In the first usage scenario (see Section 3), SiteGuide outputs the example site model in the form of human readable statements. An example is the statement in Figure 3(a).

In the second scenario SiteGuide compares the example site model to a draft of the new site. It creates a mapping between the example site mapping and the draft. For this it uses the same method that created the example site mapping, except that this time it does not alter the already created page sets of the example sites nor the mappings between them. Once the draft is mapped, SiteGuide searches for differences between the draft and the example site model. It determines which topics in the model do not have corresponding pages in the draft and reports that these topics are missing on the new site (e.g. Figure 3(b)). Conversely, it determines which topics of the draft do not have counterparts in the example sites and reports that the new site is the only site that contains these topics. Finally, it compares the secondary features of the topics in the new site to the features of the topics in the example site model and reports the differences.

5 Evaluation

We evaluate the mapping method on sites from three domains: windsurf clubs, primary schools and small hotels. For each domain 5 web sites were selected as example sites. We purposely chose very different domains: the windsurf clubs are non-profit organizations, the school domain is an educational domain and the hotel domain is commercial. To be able to evaluate the quality of automatically created mappings we manually constructed for each domain a gold standard mapping. In the gold standards, pages with similar topics are mapped onto each other. Table 1 shows the

domain	sites	total pages	min-max pages	topics in g.s	mapped topics in g.s.	% pages mapped in g.s.
hotel	5	59	9-16	20	10	81%
surfing	5	120	8-61	38	14	76%
school	5	154	20-37	42	24	80%

Table 1: The number of example sites and the total, minimum and maximum number of pages in the example sites. The number of topics in the gold standards, the number of topics in the gold standards that are found in more than one site (mapped topics) and the percentage of the pages in the gold standards that are mapped onto at least one other page.

main properties of the three evaluation domains and the gold standard mappings.

The quality of mappings is expressed by three measures: precision, recall and f-measure over mapped pairs of pages. For each two pages from different example sites, we check whether the two pages occur in one mapping element in the gold standard mapping and in the mapping that we want to evaluate. When P_{gold} are the page pairs in the gold standard and P_{test} are page pairs in the test mapping, the measures are defined as:

$$\begin{aligned} \text{precision} &= |P_{test} \cap P_{gold}| / |P_{test}| \\ \text{recall} &= |P_{test} \cap P_{gold}| / |P_{gold}| \\ \text{f-measure} &= (2 \cdot \text{precision} \cdot \text{recall}) / (\text{precision} + \text{recall}) \end{aligned}$$

5.1 Results

To test the influence of the various similarity measures, we performed a series of experiments in which we varied the weights of the measures. On all three domains text similarity proved to be the most important factor. In other words, giving a high weight to the text similarity resulted in high precision and recall. This corresponds to the intuition that the text on a page gives most information on the page's topic. In the hotel and the surfing domains URLs also appeared to be effective. In the school domain including URL similarity did not improve the mapping. In this domain 3 of the 5 sites had URLs that were meaningless identifiers, such as 'www.fendraytonschool.co.uk/page6.html'. The page titles were in all domains not very informative as on many sites all pages had the same title. The same holds for the link similarity: many sites had a simple one-layer menu that linked (almost) all pages to (almost) all other pages. The anchor texts did differ between pages. Most likely, the reason that anchors did not work well, is that anchor texts are too short to provide enough information.

The influence of the minimum similarity, parameter c, can be seen in Figure 5. The figure shows the situation for the school domain; figures for the other domains look similar. When the minimum similarity increases, we require mapped pages to be more similar. As a result, only the best matching pages are mapped, so that precision is increased, but recall decreased. Thus, with this parameter we can balance the quality of the topics that we find against the number of topics.

When the SiteGuide system is used by a real user, it obviously cannot use a gold standard to find the optimal parameter settings. Fortunately, we can estimate roughly how we should choose the parameter values by looking at the resulting mappings. Mappings with too few pages per mapping element do not reveal very much about the similarities of the example sites. Mappings with too many pages per mapping element do not give much information either, as they

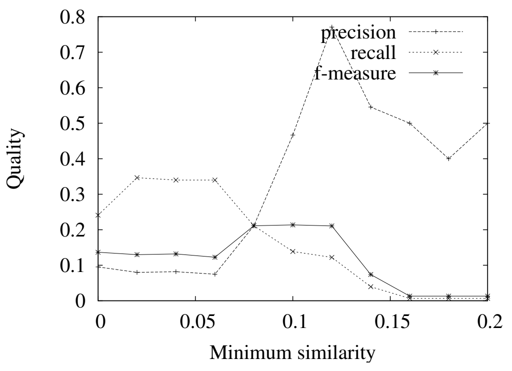

Figure 5: Quality of mappings of the example site from the school domain created with various values of the minimum similarity c.

map almost everything on everything. In our experiments we found that parameter c had a good value if the average number of pages per element was between 1.25 and 2.

Good similarity weights can be found by looking at the distribution of the similarities. If all page pairs have roughly the same score on a similarity measure, apparently this measure is not informative, which means that the measure should receive a low weight. High weights should be assigned to similarities with high entropy (high scores for a small number of page pairs).

Table 2 shows the scores of the mappings with the highest f-measures. The table suggests that SiteGuide creates better models for smaller sites. On smaller sites there are fewer possible mappings, so that the probability of finding a correct mapping is larger.

domain	precision	recall	f-measure
hotel	0.55	0.36	0.43
surf	0.34	0.21	0.25
school	0.47	0.14	0.21

Table 2: Quality of the example site mappings created with the parameter values that maximized f-measure.

Inspection of the example site mappings showed that SiteGuide was able to discover many useful topics. We give a few examples. In the school domain SiteGuide created a mapping element that contained for each site the pages with term dates. It also found correctly that 4 out of 5 sites provided a list of staff members. In the surfing domain, an element was created that represented pages where members could leave messages (fora). In the hotel domain all pages about the types of rooms were grouped into one element. These examples demonstrate that SiteGuide can indeed help a user to formulate his requirements. Output statements about these elements provide useful tips about the site that the user wants to build. In this way, the general idea of 'I want something like these sites' is transformed into a list of very specific requirements. Following scenario 2, a user can get feedback on his draft site. For example, when the owner of the fifth school site uses SiteGuide, he learns his site is the only site without a staff list.

Not all elements in the mappings represented exactly one topic. In an element about weather conditions, we found a page about surfing locations. It is easy to see where this mistake originated: the location page provided information about the conditions in the various locations and therefore contained weather related terms, such as 'wind', 'wave' and ‘water’. In the hotel domain an element with four pages about prices of rooms included one page about the restaurant of one of the hotels. Although, these elements are not entirely correct, the general topics are still clearly recognizable, so that useful statements can be generated. Some topics that the sites had in common were not found at all, because the terms did not match. Two school sites provided information about school uniforms, but on the one site these were called ‘uniform’ and on the other ‘school dress’. These examples show the limitations of the term-based approach. To be able to correct these mistakes, more semantically informed methods are needed. In the future, we will extend SiteGuide with WordNet [Fellbaum, 1998], which will enable it to recognize the proximity between terms like ‘uniform’ and ‘dress’.

6 Conclusions and outlook

In this work we addressed the problem of providing assistance to amateur web designers who want to build a web site but are not able to accurately express their requirements. The SiteGuide system allows web designers to specify their needs in an informal way by means of example web sites that are similar to the one they would like to create. SiteGuide automatically analyzes the example sites and returns a model that describes the main features that the sites have in common. In addition, it can show differences between example sites and a first version of a new site. In experiments done with example web sites from three domains, SiteGuide proved able to find many important features of the sites. The techniques presented here could be used for other, similar tasks as well. Examples include identifying website copycats or identifying the ’most typical website’ for a domain.

Although the first results of SiteGuide are promising, more evaluation is needed. We are currently testing the tool’s ability to discover mistakes in a draft design of a web site. We remove topics and links from a web site and add unnecessary information. The corrupted site is entered as draft in SiteGuide together with a set of example sites. These experiments enable us to see how many of the missing features SiteGuide is able to reveal. Another experiment that we are currently evaluating will reveal how correct and understandable the identified topics are to users. For this, we ask non-experts to describe the topics that are presented to them in the form of Figure 3. These descriptions are then evaluated with respect to a gold standard. Further user studies are also planned and will co-occur with the development of a more sophisticated user interface.

The current version of SiteGuide maps sites on the basis of superficial textual features. Our experiments suggest that the example site models can be improved by including more semantics. We are planning to connect SiteGuide to WordNet or domain specific ontologies to improve the mapping of web pages.

Besides the contents and the link structure, a web designer also decides on the web site’s style. Example sites can provide inspiration for style features, such as colors and the use of images. Therefore, including these features in SiteGuide could be a valuable addition to the system.

References

[Armani, 2005] J. Armani. VIDET: A visual authoring tool for adaptive websites tailored to non-programmer teachers. *Journal of Educational Technology and Society*, 8(3):36–52, 2005.

[De Bra *et al.*, 2007] P. de Bra, N. Stash, D. Smits, C. Romero and S. Ventura. Authoring and management tools for adaptive educational hypermedia systems: The AHA! case study. *Studies in Computational Intelligence*, 62:285–308, 2007.

[Cristea and De Mooij, 2003] A.I. Cristea and A. de Mooij. LAOS: Layered WWW AHS authoring model and their corresponding algebraic operators. In *Proceedings of the Twelfth International World Wide Web Conference*, Budapest, Hungary, 2003.

[Falkovych and Nack, 2006] K. Falkovych and F. Nack. Context aware guidance for multimedia authoring: Harmonizing domain and discourse knowledge. *Multimedia Systems*, 11(3):226–235, 2006.

[Fellbaum, 1998] C. Fellbaum, editor. *WordNet: An electronic lexical database*. MIT Press, Cambridge, MA, 1998.

[Heß, 2006] A. Heß. An iterative algorithm for ontology mapping capable of using training data. In *Proceedings of the Third European Semantic Web Conference*, pages 19–33, Budva, Montenegro, 2006.

[Hollink et al., 2007] V. Hollink, M. van Someren and B. Wielinga. Navigation behavior models for link structure optimization. *User Modeling and User-Adapted Interaction*, 17(4):339-377, 2007.

[Hu and Qu, in press] W. Hu and Y. Qu. Falcon-AO: A practical ontology matching system. *Journal of Web Semantics*, in press.

[Levenshtein, 1966] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[Munkres, 1957] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5(1):32–38, 1957.

[Newman and Landay, 2000] M. W. Newman and J. A. Landay. Sitemaps, storyboards, and specifications: A sketch of web site design practice. In *Proceedings of the Third Conference on Designing interactive systems*, pages 263–274, New York, NY, 2000.

[Perkowitz and Etzioni, 2000] M. Perkowitz and O. Etzioni. Towards adaptive web sites: Conceptual framework and case study. *Artificial Intelligence*, 118(1-2):245–275, 2000.

[Pierrakos *et al.*, 2003] D. Pierrakos, G. Paliouras, C. Papatheodorou and C. D. Spyropoulos. Web usage mining as a tool for personalization: A survey. *User Modeling and User-Adapted Interaction*, 13(4):311–372, 2003.

[Salton and McGill, 1983] G. Salton and M.J. McGill. *Introduction to modern information retrieval*. McGraw-Hill, New York, NY, 1983.

[Snoek *et al.* , 2004] C. G. M. Snoek, M. Worring, J.-M. Geusebroek, D. C. Koelma and F. J. Seinstra. The MediaMill TRECVID 2004 semantic video search engine. In *Proceedings of the second TRECVID Workshop*, Gaithersburg, MD, 2004.

[Spink *et al.*, 2000] A. Spink, B. J. Jansen and H. C. Ozmultu. Use of query reformulation and relevance feedback by Excite users. *Internet Research: Electronic Networking Applications and Policy*, 10(4), 317–328, 2000.