Concept Tagging and Dynamic HTML Generation for Adaptive Teachware

THOMAS T. FUHRMANN

Praktische Informatik IV
University of Mannheim
Germany
fuhrmann@informatik.uni-mannheim.de

Abstract: Static HTML links are not appropriate for quickly evolving teachware to which several authors independently contribute material on overlapping topics. Furthermore static links are not capable of providing students with an individualized linearization of the hypertext suited to their individual level of previous knowledge and competence. We hence propose a linking mechanism based on concept maps and a respective tagging of the source material for the course. These tags are used to dynamically generate HTML pages that reflect the learning level of the individual students.

1. Introduction

The ability to use mathematical language is one of the most important ingredients for a successful study of natural sciences and engineering. It is also one of the subjects feared most by students since training in its abstract way of thinking is not thoroughly practiced at school. Hence a dislike of mathematics or the self-perceived failure to understand its concepts is one of the most frequent reasons for not choosing technical subjects.

The Faculty of Physics at the University of Heidelberg is thus trying to smoothen the entry into the abstract world of physics by offering a two-week tutorial course in mathematics for first year undergraduates. During this intensive course the students can revise and practice their skills in central areas of mathematics like calculus, complex numbers, vectors, and matrices that are required in physics almost from the first day on. Especially students who are not fresh out of school but who have completed their military service or an apprenticeship prior to their studies show a great need for intensive recapitulation of mathematics.

Since the mathematical knowledge of first year undergraduates differs vastly from student to student a personal tutor for each would be highly desirable though of course not affordable. Therefore, two years ago we began experimenting with various electronically enhanced means for undergraduate mathematics education that can do more than the usual multimedia-enhanced teachware.

Two specific challenges characterize our project. Firstly, many authors with different objectives contribute on a continuous basis to our courseware even though it is already being employed. Secondly, the students using the courseware are rather disparate with regard to their previous knowledge, motivation and intended careers. The notion of links as incorporated in the HTML specifications suffers from some major drawbacks with respect to either challenge:

A link can only be established if its destination is known upon coding the source for the page containing that link. If for example an author decides later on to provide additional information on a specific topic all pages that are to link to that new page will have to be re-edited.

Links to a given page will be broken if that page is removed. If for example an author splits a page in order to better accommodate the topics contained in the original page, all links pointing to that page then have to be updated to the respective locations of the newly created pages. This requires not only a complete check of all pages that could possibly be linked to the page in question but also a certain understanding of both the source and the destination topic.

Human knowledge and human thinking differ profoundly in structure. While the structure of the former is interwoven the latter is in principle strictly linear. This linearity can only be broken by again picking up points of thought abandoned earlier. This pseudo-linearity is commonly realized as a hierarchical tree-like structure.
However, HTML courseware based on this principle tends to resemble what can be done in books. An unstructured linkage on the other hand is hard to use and therefore often rejected by students.
A novel mechanism able to circumvent these drawbacks should hence be able to link the different parts of the courseware using invariant features of the content itself. It must not depend on a specific representation of that content. The mechanism should also reflect possible linearizations that are meaningful to the reader.
We propose a three-step solution to that problem: Firstly, we describe the invariant structure of the course content by means of a simplified type of concept map. We then tag these concepts to the HTML material. In the third step we use the mapping and tagging in steps one and two to dynamically generate HTML pages that have now been individualized for each user.
Concept-based structures are a common tool for providing HTML metadata. Concept maps or graphs are described by several authors as a graphical tool for structuring ideas [1], [2]. Novak presents many examples of how concept maps improve the results of teaching efforts [3], [4]. Based on these and other experiences concept maps have meanwhile been directly employed in several hypertext applications [5]-[9].
This paper is structured as follows. In section 1 we have given an overview of our motivation and the problems encountered during the preparation of our courseware. In section 2 we present our concept map approach and the corresponding tagging mechanism that we developed to handle these problems. Section 3 explains the dynamic HTML generation and gives a practical example. Section 4 reports on the early experience with our tools. In section 5 we summarize our project and draw conclusions.

2. Concept maps and HTML tagging

Thinking is often equaled to speaking. Particularly if we also consider non-verbal symbols to be words in an abstract language. In this sense, words or phrasal combinations of them are the building blocks of our thinking. They reflect our concepts of a certain subject. Naming concepts is a major activity during the cultivation of new areas of knowledge. And being able to appropriately name and relate concepts during a dispute is taken as a sign of understanding.
Relying on this fact we made concepts the features on which we base our linking. Concepts, however, should not be confused with the words which describe them. The former are essential and unique, the latter somewhat arbitrary. The scientific context often establishes a rather strict terminology, mostly with one, sometimes with two but rarely more equivalent names for a concept. We will consider this equivalence problem later, for the time being taking uniqueness for granted.
As the building blocks of our thoughts and knowledge concepts are glued together by their mutual relations. These in turn can differ greatly in depth and complexity, ranging from easy 'has a'-type relations to more intricate relations such as 'changes color if in contact with' when for example describing an indicator - acid relation. A graphical arrangement of concepts and their relations is called a *concept map* and can serve as a tool for structuring one's knowledge. Some authors explore more complex mappings where relations can also be equipped with attributes, e.g. further concepts.
For our purpose that multitude of relations disturbs the automated handling of concept maps. Relations in our initial prototype were restricted to the following three types: 'requires an understanding of', 'explains' and 'is an example of'. While this simplified the coding and usage of the concept map it also spoiled the clear definition of concepts. Information had to be incorporated into the concept that originally should have been contained in the relation. An example presented in the course is no concept in the original sense but content related to another concept already defined in the concept map.

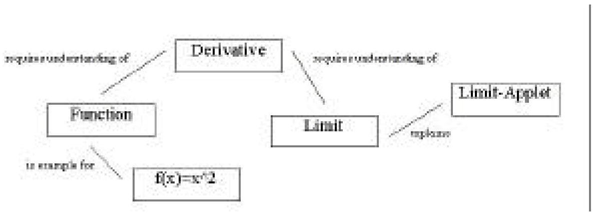

Figure 1: Simplified example of a relation structure used by our initial prototype. Concepts are linked to other concepts by three kinds of relations: 'requires an understanding of', 'explains' and 'is an example of'. The distinction into different types of relations depending on the didactic use of a piece of content spoils the clear definition of concepts.

Although our initial prototype that incorporated this scheme was rather successful we improved the scheme of concept maps. This new, improved scheme more clearly separates the two types of concepts which had been mixed in the initial prototype. Concepts describing the content itself and the associated didactic aspects are now handled by two different concept maps. The first contained concepts like "Function" and "Derivative", the second, concepts like "Introduction", "Example" and "Exercise". Didactic concept maps are much less complex and hence treated differently in our system. The content concept maps on the other hand have now degenerated into a prerequisite map, i.e. all relations have the meaning "is prerequisite to". For later reference we call concepts that are prerequisite to a given concept "ancestors" of that concept.

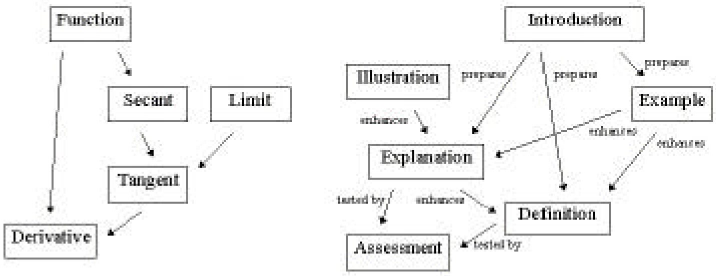

Figure 2: Simplified example of content concept maps (left) and didactic concept maps (right) used in our improved prototype. Content concept maps describe the relation "is prerequisite to". Pieces of content in our courseware are tagged using both content and didactic concepts. This allows for individualized linerarization of the material.

The HTML courseware is now tagged with pairs consisting of one concept from each category. Each piece of content is provided with a concept name and a tag indicating its didactic function.

```
<CONCEPT NAME= "Derivative" TYPE= "Definition"> The function mapping x to the
slope of the tangent to the function f in x is called the derivative of f. It will
be denoted by f'. <\CONCEPT>
```

This tagging can be done in parallel to the authoring of the text itself. Different authors have only to agree on the didactic concepts, i.e. they must have a common idea of basic didactic tools like introductory texts, examples and exercises. However, no further structural definition is required. Concepts can be identified by the author of the respective part of the course. Overlapping material is then combined automatically by means of the content concept map and the tagging in the HTML source.

3. Dynamic HTML generation

The content and the respective concept maps can be used in two ways. The first integrates the capability to deal with concept maps into the browser. The second possibility employs a modified HTTP server which itself processes the pages before they are shipped to the students.
Since our initial prototype pursued not only the integration of these concept-based navigation facilities but also more elaborate multimedia integration of mathematical texts and Java applets, we implemented a special browser in Java. Here Java animations could be seamlessly mixed with the bitmaps created from LaTeX sources, providing full rendering of all mathematical formulae without the usual hurdle of having to mix HTML and GIFs..
The improved prototype is currently implemented as a Java application running on the course's web server. The applets communicate the assessment results to that server process and receive in return the URL for the next course segment.

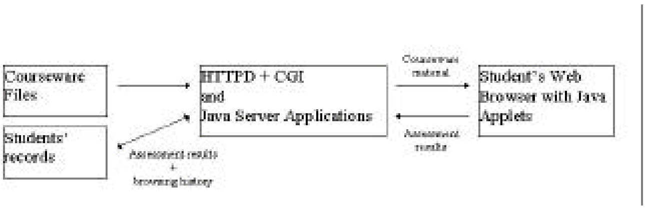

Figure 3: Brief outline of the data flow in our courseware. The sources containing the concept tags are stored on the web server together with the students' records. A mixture of CGI-scripts and Java applications ship the HTML pages to the browsers and maintain the records with the assessment results provided by the Java applets on the client side.

With the help of the concept map and the concept tags we now generate HTML pages that are transferred to the student's browser. The browsing history of each student is logged on the server together with the students' results obtained in the assessments. If a student's result is better than the defined threshold of 0.5 the prerequisite relation is assumed to have been satisfied for that student. A concept is said to be established if all its ancestors have been established and all its prerequisite relations have been satisfied. This mechanism provides at each point in time a set of concepts with fully established ancestors. From this set and according to the didactic concept map the server chooses a piece of content that is next shown to the student. Presentation of a new concept will typically begin with an introductory part. Where no introduction has been provided the server will choose an example to enhance an explanation or definition. Finally the student will do some exercises and take some tests as an assessment. Sufficient performance on these tests then establishes the concept.
In the common case that there are several concepts with fully established ancestors, the student is presented with several possible tracks. This provides the greatest possible freedom to the students while guaranteeing a sound trail of study.

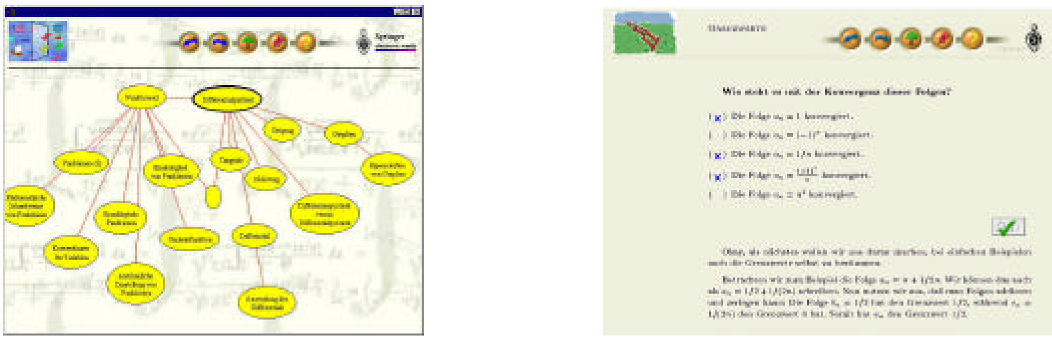

Figure 4: Screenshots from our intial prototype. The left hand side shows a content concept map for the current concept and its nearest descendants. On the right hand side a typical page from the course itself is shown. Besides multiple choice questions tests can require entering of numerical values and simple formulae.

In our implementation we paid less attention to the students' browsing history, relying more on their assessment results. This allows the students to quickly pass through topics that are already clear to them. Accordingly the courseware provides introductory pages that sketch the problem and directly lead to relatively tough questions and tests. Students are free to try these questions to quickly establish the concept or take the long way through the explanatory pages of the course.

4. Early experience

Two main reasons inspired us to investigate the techniques described above. Firstly, our students have a vastly disparate previous knowledge of the mathematics taught in our course. Secondly, the hypertext material and the accompanying Java applets are in constant evolution. Especially this issue could not have been handled conventionally. During the two years of this project the text for the entire course was rewritten three to four times, not counting the dozens of minor changes like splitting and combining chapters, or the insertion of new examples and deletion of old ones. Finally the exercises and tests were modified almost daily. During this time the course was taught by five professors and two more lecturers provided additional material.

Despite this ongoing evolution we were always able to provide the students with a working prototype of the course. That prototype could be handled by a single person, who administered the web and video streaming servers and supervised the students programming Java applets for the course. Our approach greatly reduces the amount of work required to maintain our teachware.

Students were also very impressed with the possibilities. Although the staff members did offer tutoring in persona the students used the electronic courseware. After the two-week introductory course more than 90% expressed the opinion that this courseware could provide them with significant tutoring if no traditional tutoring by staff members was available.

However, the results are somewhat unsatisfactory since the capabilities of our system were not really tested independently of the traditional course. A profound analysis can only be done if students work solely with the electronic courseware. Since our program is scheduled to be published by a large scientific publisher in 1999, we are confident we will soon obtain real data on usage and performance of our courseware.

5. Conclusions

Transforming tutorials that have been traditionally held by staff members into interactive courseware is a challenge for two reasons: Several authors, each with their own ideas and objectives about the course, contribute simultaneously to the teachware. The resultant constant evolution hinders any centralized planning. Secondly, the students differ greatly in their previous knowledge and are interested in different aspects of the course. It was hence necessary to design a mechanism that is able both to combine different contributions without the need to define a fixed structure or hierarchy of the content and to allow for individualized linearizations of the material.

We chose concept maps as the base for linking the different parts of the course. Content concept maps reflect the structure of the topics taught while didactic concept maps describe the different functions of the individual pieces of the course. Both concept types are used to tag the source of the courseware. Server applications then dynamically generate HTML pages that are shipped to the students' browsers.

The system has proven very successful. Despite the constant changes in the course a consistent view of the material could be given at any time. The necessary maintenance work was minimal. On the other hand, students could quickly navigate the material since the system was able to present them the material best suited to their individual level of competency. We are confident to soon be able to provide a thorough analysis of the capabilities of our system since the courseware is scheduled to be published for general use in 1999.

References

[1] J. D. Novak, D. B. Gowin, "Learning how to learn", Cambridge University Press, 1984
[2] John F. Sowa, "Conceptual Structures, Information Processing in Mind and Machine", Addison-Wesley Publishing Company, 1984
[3] Joseph D. Novak, "Learning, Creating, and Using Knowledge", 1998
[4] Joseph D. Novak, "Concept maps and Vee-diagrams: Two metacognitive tools for science and mathematics education", Instructional Science 19, 29-52
[5] Thor Anderson "Knowledge components" www.learningcomponents.com"
[6] Brian R. Gaines, Mildred G. Shaw, *International Journal of Human-Computer Studies*, Knowledge-Based Hypermedia, 43(3), pp. 323-361, 1995
[7] Gerd Kortemeyer "Prerequisite maps" www.lite.msu.edu
[8] J. G. Lambiotte, D. F. Dansereau, D. R. Cross, S.B. Reynolds, "Multirelational semantic maps", Educational Psychology Review 1(4) 331-367
[9] Brian R. Gaines, Mildred G. Shaw, "Concept maps indexing multimedia knowledge bases", AAAI-94 Workshop: Indexing and Reuse in Multimedia Systems, pp 36-45, Menlo Park California, AAAI