

Implementing Teaching Strategies in the Classroom

Abstract. In this paper we propose a method and its implementation for structuring multimedia-based teaching-learning material. A new lesson manager tool generates lesson graphs following a pre-selected teaching strategy. These lesson graphs are then used for implementing the curriculum of the lesson. The system software architecture and the lesson navigation system are briefly explained. The lesson navigation system using different teaching strategies was tried out successfully in two standard lectures with three undergraduate learning groups. Different statistical tests were undertaken to evaluate the collected students' utterances.

Key words: Computer based learning material, Concept maps, CSCL, teaching strategy

1. Supporting teaching presentation in the classroom

Computer equipped classrooms to support teacher's presentation of multimedia learning material are becoming a standard in universities and high schools. Arrangements can vary from a simple computer attached to a beamer for projecting the display or hypertext slides [5] to very sophisticated classrooms equipped with high-technology "roomware" [16,10] where each student is also provided with a computer enabling computer-supported collaborative learning activities.

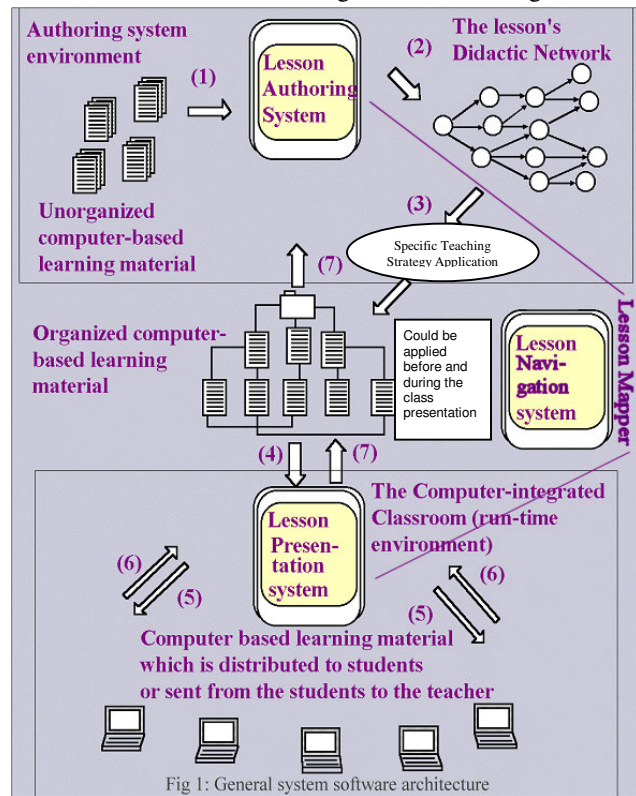


Fig 1: General system software architecture

In this paper, we do not preclude a specific physical environment, but we envisage a teaching-learning session as a coherent sequence of teaching-learning activities like teacher presentation, discussions, individual or collaborative problem solving. Our particular focus is on the structuring of material for this purpose to guarantee both high flexibility and a certain degree of guidance.

This work is based on a previous developed system called COSOFT as a basis for enabling teacher's presentation support in an XML based environment [1,2,3].

Figure 1 explains the software architecture that provides a general scheme of the preparation and development process based on the flow of the learning material and introduces the new lesson navigation system. The material (1) is created or selected and organized as a didactic network with the help of the authoring tool (2). Applying filters we construct a teaching path on the lesson network (3), an organized lesson with this material is obtained (4) to be used in the CiC. Depending on

the contents of the lesson prepared by an author, data and information may flow from the teacher's computer (attached to the e-blackboard) to the students' computers and vice versa. For example, the teacher distributes learning material complementary to what is being presented, or distributes an electronic worksheet with which the students are to work during the class (5). The students can contribute additional learning material during a lesson, e.g. relevant documents found in a database or in the WWW. A student also may present a solution to a problem given by the teacher (6). The teacher can present this material to the rest of the class and may include this as permanent material in the lesson. This can also be reflected in the author's environment (7). In the following, we assume this extended version of the original COSOFT architecture as a basic framework.

2. Didactic networks for structuring computer-based learning material

In the context of designing curricula for adaptive and intelligent computer tutors, Halff defines the curriculum as the selection and sequencing of learning material [9]. McCalla redefines this as the selection and sequencing of *knowledge elements*, which are adequate for the learner in order to meet certain learning goals according to his background and learning possibilities [14]. McCalla also distinguishes two phases in the development of the curriculum.

The first one is the so-called generic curriculum, which consists of the selection of material and learning activities. Specialists in the learning domain do this with a prototype student in mind. The second phase consists of tailoring the curriculum designed in the first phase to fit the learning needs and other requirements such as a preferred learning or teaching strategy, or a particular learning context (background, learning capabilities, etc.) for a specific learner group. The teacher in the classroom performs the second phase. McCalla [14] presents a number of self-adapting tutoring systems for supporting individual learners and he considers the directed graph as a key structure for the learning unit syllabus in order to achieve flexibility. This is because a graph allows more than one way to sequence concepts while maintaining a coherent order between predecessor and successor nodes. This structure is better than considering a syllabus as a sequential list of concepts to be taught. We also use here a graph to represent the lecture's syllabus, which we call Didactic Network. If a curriculum is represented by a graph in a didactic network, it is easy to draw a parallel between the first phase of the curriculum development with the construction of a graph (at "design time") and the second phase which takes place at "delivery time", e.g., in the classroom.

Recently, a number of projects have tried to develop flexible learning material or adaptive courseware, such as IDEALS MTS [8] or adaptive hyper-textbooks like Interbook [6] or Multibook [17]. Systems like Eon IST [15] and PERSEUS [13] are focused on the creation of a knowledge ontology to define topics and links in a semantic net representation. Although our application scenario, the support of *teaching* in a classroom is substantially different from the adaptation or automatic configuration of learning materials for individualized learning, there are parallels in the definition and use of didactic relations, e.g. with [17]. In the next two sections the characteristics of the nodes and links of our didactic network are introduced.

2.1 The nodes As we presented in details in (2), in a didactic network, a node represents an abstraction of a subject that is to be taught and learnt. Nodes are labeled with keywords that described its content. They are also typed with the learning activity, which should take place when visiting the node. To perform the learning activity, teacher and students (also) use the computerized learning material, which is associated with the node. The node types are shown in Table 1. The four first node types correspond to presentation activities performed by the teacher. The last three node types are for activities involving students. For these last node types the electronic learning material used can be of any type. We think that for Discussions and Group-work systems [7] these node types are very suitable.

Node type name	activity involved while visiting
<i>Graphic presentation</i>	for presentation of mostly graphic information to the students
<i>Animation presentation</i>	for presentation of animated & interactive simulation programs
<i>Audio Presentation</i>	for presentation of mostly audio type information to the students
<i>Video presentation</i>	for presentation of video clip to the students
<i>Text presentation</i>	for presentation of mostly textual information to the students
<i>Discussion</i>	for a group discussion
<i>Individual work</i>	the main activity is the individual work of the students
<i>Group work</i>	for structured collaborative work

Table 1: Node type and involved activity

By typing the node according to a learning activity, the author of the lesson graph must think first about the subject and the concrete activity through which the subject will be taught or learnt and afterwards about the computerized learning material required. The (multi-)media material can be associated with the nodes when it is created or afterwards. The later assignment of types enables the author to refine or modify his ideas about the dynamics of a lesson. Node types may also indicate the time needed to teach the specific content. For example, it is quite likely a video clip or an audio presentation will take less time than a discussion, individual work or group work. Such features are particularly modeled and described using new XML standards such as "Learning Object Metadata" (LOM) [12] to make them easily exchangeable between systems. Since our work is also based on XML, we plan to make our model compatible with this standard building XML-transformators (*XSLT*).

2.2 The links The link types to be provided depend on the intended kind of system support, i.e., the guidance functions the system will offer to its users. Intelligent systems which use a graph for knowledge representation tend to use semantic links [4]. Rhetoric links, on the other hand, describe the strategy or discourse used by the author to present the information represented by or contained in the nodes. Because the aim of our system is to support the authoring and presentation of learning information and not to give help based on the knowledge acquired by the learning group, it is natural to use rhetorical links for constructing didactic networks.

The set of rhetoric links presented by Trigg [19] has been modified and complemented in order to provide the curriculum designer a set of relations for constructing the lesson graph according to a preferred presentation strategy.

Based on on experience with this approach, we have recently introduced a new type of link: *introduces by example*. Table 2 represents the relationships established by a link which points node Y from a node X. These are

Link type name	represented relationship	intended usage
<i>X introduces to Y by examples</i>	X introduces the subject Y by graphics or examples	recommended for the beginning of a lesson and introduction of new topics
<i>X introduces to Y</i>	X introduces the subject Y	recommended for the beginning of a lesson and introduction of new topics
<i>X refined by Y</i>	the subject Y is a part or a detail of the subject X	may be used to split a topic in various sub-topics.
<i>X explained by Y</i>	the subject X is explained more deeply by the subject Y	may be used to justify or support the idea of the predecessor
<i>X exemplified by Y</i>	Y is an example of the subject X	may be used to illustrate the idea of the predecessor's subject
<i>X summarized by Y</i>	Y is a summary for all the nodes linked by this type of link, having Y as successor	there should be normally more than one predecessor linked with this type of link.

Table 2: Relationship and intended usage

As for the learning material metadata, the information about the structures of the lessons is kept inside our architecture in XML format.

3. Lesson navigation system and teaching path

To enable users to take advantage of the structure of the graphs defined as lessons, units and courses, we must implement a visual navigation tool. It should be possible to use the graph as a basis for supporting the teaching and learning process. There should be a mechanism to guide users within the didactic network during a lesson. This mechanism should be flexible enough to coach teachers to choose the best path according to their needs and the dynamic nature of the lesson, allowing them to modify strategies during the lecture. Therefore, the type and degree of guidance should be adapted to the user's (teacher's) requirement.

There are two basic functions which may give good advice in many possible situations when guidance is required: a) displaying the graph in a way that highlights the possible lecturing threads and b) answering the question "Which is the best next node to visit?" Both functions should take into account the user's profile. The lesson navigation tool may permit the user to control the path of the lesson (the order in which the nodes are visited) and gives information about the state of the lesson, that is, what is the current node and what are its predecessors and followers.

A simple help : A function for suggesting navigation routes through the lesson graph which is at once very easy to implement but powerful is the automatic positioning of a "current node" pointer at a node which has at least one non-visited son. The teacher may continue the lesson by choosing any of the non-visited sons or backtracking on the tree until eventually reaching the root, if desired. This is very helpful in the case of visiting a leaf or reaching a node whose sons were already visited by following another path. This can spare the need for pressing a "go back" button while navigating through the network. It is easy to note that this function will leave no node unvisited unless the teacher explicitly wants to do so.

Introducing teaching and learning preferences : By combining the ideas of the most related concepts spanning tree and the positioning of the current node pointer at a node with at least one non-visited son, it is possible to implement some interesting algorithms which support the traversing of the graph under different students' and/or teachers' profiles, or different requirements which arise from the teaching and learning context. The approach is the following: a spanning tree of the lesson graph is generated after the principle of the "most related concepts" but the table with the weight-function is given as a parameter, reflecting the teacher's and/or students' preferences. By calling the NEXT-NODE help function, the following algorithm is performed : if the current node has non-visited sons then select and return the one connected with the link which has the shortest distance. Otherwise, go up to the father of this node and recursively apply the algorithm. This will indeed find the node with the subject most related to the current one which has not been visited yet. The algorithm can be easily extended to construct a list with the possible next-nodes ordered by closeness according to the distances. Of course, if the function which assigns a distance to a link-type changes, the result of the help function also changes. Not only because the selection at the time of performing the help function will have different values to compare but also because the minimal spanning tree is generated in another way. Let us see how we can implement the support for different learning strategies by changing the distance function.

Inductive Learning : By *inductive learning* we understand the learning process follows an inductive strategy; this means individual examples and facts will induce the understanding of general rules and laws. This is why in such a

learning mode the examples have higher priority than the explanations. This can be implemented by assigning a short distance to the links labeled with *exemplified-by*. In Table 3, the second column shows an example of value assignment for implementing this learning mode.

Deductive Learning : In a *deductive learning* mode the rules and laws are explained first and then individual facts and examples are deduced from them. This can be implemented by assigning a short distance to the links labeled with *explained-by*. In Table 3, the third column shows an example of values assignment for implementing this learning mode.

A “short version“ of the lesson : In this learning mode only some of the nodes are visited in order to have an overview of the whole lesson without getting into details. In order to let some nodes out of the traversing path we can mark some links with an infinite distance. This means these links will never be selected during the construction of the minimal spanning tree. The nodes we can leave out of the lesson for a short version are the ones linked with an *explained-by* and/or *exemplified-by*. In Table 3, the fourth column shows an example of values assignment for implementing this learning mode.

Link Type	Value for in-ductive strategy	Value for de-ductive strategy	Value for short version
Introduces by example to	6	2	1 (or 2)
Introduces to	5	6	2 (or 1)
Explained by	1	4	∞
Exemplified by	4	1	∞
Refined by	2	5	3
Summarized by	3*	3*	4*
* this link is considered only if all the predecessors of the pointed node have already been visited			

Table 3: Examples of different teaching/learning strategies

4. The Lesson Mapper

The Lesson Mapper is a new software module with a graphical navigation interface which acts on two levels in the framework presented in the first chapter. On one hand, the Lesson Mapper will be used as “*lesson authoring system*” for organizing some learning material in the form of a didactic network. Secondly, it is to be used in the context of a face-to-face situation in which the teacher would present the lesson directly from the didactic network potentially following one of the specific strategies. Here, the Lesson Mapper serves a graphical navigation support tool that interfaces with the CiC repository in order to display, rearrange, distribute, and collect the learning material. In this sense the Lesson Mapper can also be seen as “*lesson presentation system*”.

The Lesson Mapper was developed in several steps. At the beginning, the existing FreeStyler system [11] was incorporated into the CiC environment as a visual tool for the generation, presentation, and manipulation of learning material. It manages and displays visual objects and handwriting add-on in a layered manner, both by the teacher and by the students. A powerful feature of FreeStyler is the usage of ad-hoc pluggable modules called “palettes” to define new functionalities. These palettes contain templates and active control elements and are defined in an XML-based standard representation. In a following step, FreeStyler has been extended with a palette to handle Didactic Networks. This “FreeStyler-DN” enables a teacher or lesson designer to define structure and contents of a teaching-learning unit. It provides a way to specify its contents together with a consistent navigation and display of didactic material. Following Colazzo & Molinari’s suggestion in order to avoid the students’ disorientation and cognitive overload [5], the use of FreeStyler-DN is limited to the teacher (whereas students use normal FreeStyler documents). During the lesson FreeStyler-DN supports the teacher in deciding which materials to display on the electronic board and which documents or collaborative tasks to distribute to the students. Fig. 2 shows a lesson graph built with FreeStyler-DN. Each node references a unit of didactic material. Double clicking on the node gives access to the corresponding material. Because of their features, we mainly use FreeStyler active documents as learning material. However, FreeStyler-DN accepts any type of material that can be opened by local applications added to the operating system like slide show, web browser, or media player.

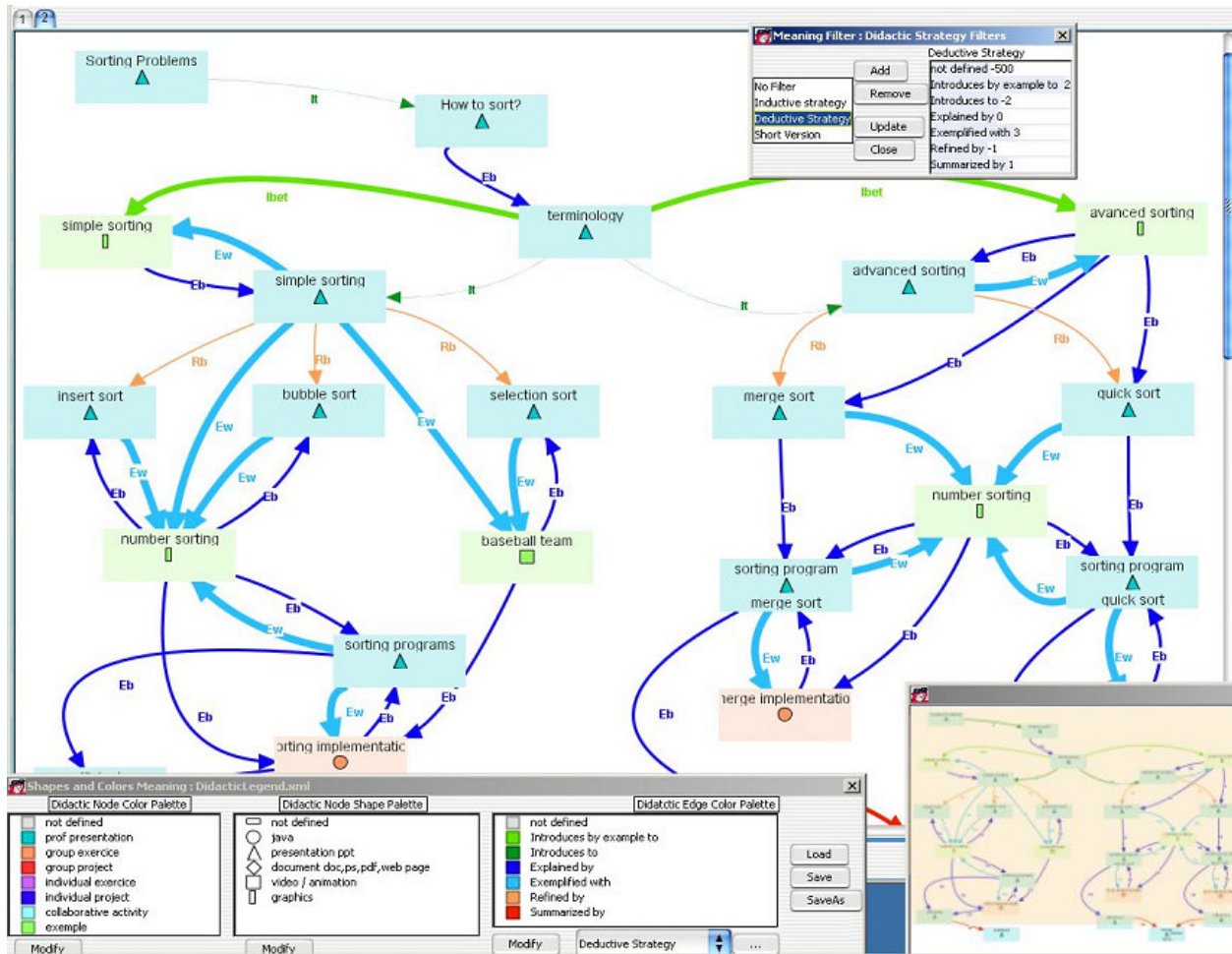


Figure 2: A lesson built with the Lesson Mapper

First, each new node is described with some keywords and edges as directed links between nodes are created. Second, a *legend* is attached to the graph. Legends give meaning to colors and shapes of the nodes and colors of the arrows in a way similar to legends used in cartography. Each node has an array of pairs that associates colors and shapes with legends. Similarly, each edge has an array of pairs that associates colors with legends. Therefore, when the user loads a legend, the graph takes the pre-specified appearance for this legend. The legends are XML files automatically built when creating the graph with FreeStyler-DN. The teacher can design and customize the legend for the specific purpose of the lesson. It can also be shared and modified by other teachers.

We designed a specific legend format for didactic purposes, the “didactic legend”, aware of the fact that the different teachers’ experiences and personalities would make it evolve. The Didactic Legend specifies the type of didactic activities with the color of the nodes and the type of documents with their shapes. For the arrows, we follow the concept of didactic networks and specify their color as different rhetoric relations between types of the nodes. FreeStyler-DN together with didactic legends and an algorithmic framework to filter the didactic network using different teaching strategies to compute and display teaching paths is called Lesson Mapper. A didactic network represented in FreeStyler-DN can also be used in “batch mode” to generate a linearized lesson plan based on one of the navigation strategies. The following section describes an experiment in which different linearizations were compared in a practical teaching scenario.

5. A case study for evaluating the significance of the notion of a teaching strategy

We wanted to test in practice the impact of using didactic networks and lesson planning systems for implementing the syllabus of a lecture and to find out whether it really helps to improve the planning and presentation of a lecture. Moreover, we wanted to find out whether students do in fact note the different strategies which are proposed by the

algorithms implemented. One of our research questions is: Are different learning paths and node orders discernible for students? For this, the lesson navigation system was tried out in two standard lectures in Algorithms and Data Structures (A&DS) at the University of Duisburg. The test groups consisted of 7 to 29 students aged from 20 to 28. The original HTML-based script material was used to create two different lesson graphs intended for two lectures. The first lecture was an introduction to storing and hashing techniques, the second to sorting algorithms.

The didactic network for the sorting chapter is given in Figure 2. Each graph consisted of about 30 nodes. Defining the amount of learning material a single node should contain is a subject of research by itself [17]. In this case, it was determined mainly by the node's type and the learning concept associated with it. The nodes were numbered according to the order of presentation of the learning material in the original script. Based on the resulting didactic network of the lesson we implemented two different learning strategies for each graph, in addition to the original script of the lesson developed without the use of didactic networks: the deductive approach and the inductive approach. After computing the spanning trees and calculating the traversing path according to weights assigned by both strategies, we obtained the following learning paths:

Original script path	1	2	3	4	5	6	7	8	9	10	11	12	13
	14	15	16	17	18	19	20	21	22	23	24	25	26
Inductive learning path	1	2	3	15	4	6	5	7	10	12	8	11	9
	13	16	14	17	23	18	19	20	21	22	24	25	26
Deductive learning path	1	2	3	4	5	7	12	6	10	9	8	11	15
	13	14	16	17	18	20	23	19	21	22	25	24	26

Table 4: Learning path for the original script, the inductive and the deductive teaching /learning strategies

The learning material was presented to three student groups. Both lessons were presented once to each group in different learning and teaching modes. For the first group the original script and the script resulting from applying an inductive strategy were used. For the second group the deductive strategy and the original script were used, and for the third group both strategies were used without announcing and explaining the differences between the different presentation modes. The learning material related to the nodes was presented within about 60 minutes; thus the same teacher held each lesson threefold. The lesson schedule for the test setting is shown in Table 5.

Group	Lesson 1 - participants	Lesson 1 - strategy	Lesson 2 – participants	Lesson 2 - strategy
1	29 p. November 13	Standard script	21 p. November 15	Inductive strategy
2	7 p. November 12	Deductive strategy	7 p. November 14	Standard script
3	10 p. November 13	Inductive strategy	8 p. November 15	Deductive strategy

Table 5: Lesson schedule for the test setting

To obtain uninfluenced utterances, students were not introduced into the different teaching strategies, we did not emphasize special presentation techniques, and the courseware was published in the WEB in the same way as before. After each lesson, students filled in questionnaires to assess their impressions and to characterise the presentation order of the different types of nodes. The test contained amongst other things several questions about awareness of the presentation order of the material and provided five levels of acceptance on a scale from 0 (not at all) to 4 (always). To both teaching strategies, Inductive and Deductive we can define a typical vector of “correct” answers V_I and V_D . These are the answers we expected from the students according to the order in which the learning material was presented for each learning/teaching style. The exact number of this value is not very important but the differences between them in each vector. Each vector has seven components, one for each question. In the Table 6 we can see the questions in the first column. The expected answers for the inductive teaching style in the second. The expected answers for the deductive teaching style are in the third column, and the actual answers (means) of the students for the inductive and deductive teaching sessions in the fourth and fifth columns respectively.

Question	TS Ind.	TS Ded.	TS Ind.	TS Ded.
1. The definition of the algorithms is introduced by application examples	3	2	2.45	1.87
2. Definitions and theorems were followed by application examples	2	4	2.81	3.2
3. At the beginning, all methods were highlighted before discussing them in detail	3	1	2.33	2.0
4. Algorithms and methods were presented in a strong order	1	3	2.9	3.07
5. At the beginning, all methods presented were formally defined	1	3	1.58	2.47
6. Methods were introduced by graphics and tables, algorithms and programs by animations	3	1	1.94	1.27
7. Methods were explained by graphics and tables, algorithms and programs by animations	3	3	3.0	3.0
Distributed scores: 0 (not at all) – 4 (always) TS: Teaching strategy, Ind.: Inductive, Ded.: Deductive	Expected values $V_{I/D}$		Observed means $\underline{V}_{I/D}$	

Table 6: Students' averages on the questions concerning the teaching strategies

Each strategy was used twice, thus we could calculate mean values for each vector coordinate of $V_I=(v_{I1}, \dots, v_{I7})$ and $V_D=(v_{D1}, \dots, v_{D7})$. It is evident that these values are different from the pre-calculated typical vectors V_I and V_D . However, we notice that the observed average values are order preserving: $v_{Ij} < v_{Dj} \leftrightarrow \underline{v}_{Ij} < \underline{v}_{Dj}$ holds for each $j=1, \dots, 7$.

We will now take a closer look at the mean values for group 3, which was the one with the fewest contradictions in their answers. In the Table 7 below the mean values are displayed only for group 3. All computations have been carried out with SPSS [18].

Statistics of Group 3		The definition of the algorithms is introduced by application examples	Definitions and theorems were followed by application examples	At the beginning, all methods were highlighted before discussing them in detail	Algorithms and methods were presented in a strong order	At the beginning, all methods were formally defined.	Methods were introduced by graphics and tables, algorithms and programs by animations	Methods were explained by following graphics and tables, algorithms and programs by animations
Strategy I								
Strategy D								
N SI	Valid	10	10	10	10	10	10	10
	Missing	0	0	0	0	0	0	0
Mean		2.70	2.50	2.00	3.00	2.00	1.90	2.90
Variance		0.90	1.39	1.33	0.22	0.89	1.21	0.77
N SD	Valid	8	8	8	8	8	8	8
	Missing	0	0	0	0	0	0	0
Mean		1.75	3.38	1.63	3.13	2.50	1.00	3.50
Variance		1.93	0.27	1.41	0.98	0.86	1.14	0.29

Table 7: Averages of group 3 on the questions concerning the teaching strategies

The non-parametric Mann-Whitney test was used to examine the collected answers concerning the different teaching strategies I and D. This test makes no assumption about the distribution of data. We used two samples derived from the questionnaires of group 3. The hypotheses for the comparison are:

- H_0 : The two samples come from identical populations and
- H_1 : The two samples come from different populations.

We transformed the samples into a ranked list, counting for all seven questions the deviations in the attended and the individual answers. Then we calculated the test variable U and the ratio $z=(U-\mu_U)/\sigma_U$. This value is compared to a table of critical values for z based on the sample size of each group. If z exceeds the critical value for z at some significance level (usually 0.05), it means that there is evidence for rejecting the null hypothesis in favour of the alternative hypothesis. Here, the calculated z is compared to the standard normal significance levels.

In the first case, the inductive strategy was used by the teacher. Here the hypothesis H_0 that both samples cannot be distinguished was right with 95% confidence. However, in the second case, using the deductive strategy, we found $z=(54.5-32)/9.52=2.36 > 1.96$ and the hypothesis H_0 that the sample can be related to strategy D had to be rejected. Therefore, on the 95% level of significance, the deductive strategy was detected intuitively by the students in group 3. This result was confirmed by the answers of this group to a further question asking for a characterisation of the relation in time between the node types in the lesson graph. The statistical evaluation proved that the majority of the students in group 3 described the deductive teaching strategy used in their own words in an appropriate way.

6. Discussion und further work

This work highlights the concept of didactic networks as semantic graphs for supporting lesson creation and presentation by teachers within the more general Computer integrated Classroom framework. Didactic networks permit to organize learning material in directed graphs so that the material is not introduced in a simple list structure but related to the other content represented as graph nodes. Those nodes can be presented to the class in a certain order, extracted and rearranged, and the relation between two nodes established by a typed link can be used to generate appropriate introductory words. The teacher presents the lesson as a specific learning path in that content graph. This path could be first based on a plan the teacher could expect for the lesson. Then the reaction of the students coupled with the application of adequate teaching strategies will make this learning path evolved.

To realize the Didactic Network concept, an XML-based Lesson Mapper Tool was designed and implemented for supporting the face-to-face classroom use of lesson graphs. In a first version of the tool, a learning path was calculated automatically depending on the pre-selected teaching strategy. This tool was used for the case study in paragraph 4. Those experiences demonstrate that automated application of teaching strategies are relevant in the intuitive perception the students have of a lesson and illustrate the significance of explicitly using those teaching strategies

when giving a lesson. Adding further material by replacing a node with an existing sub graph, bypassing several nodes due to a severe time management or modifying the teaching strategy on the fly is a need that teachers could expect when students do visibly not agree with the teaching strategy. A forthcoming version of the Lesson Mapper distinguishes in a layered graph visualization the lesson graph, a computed learning path and the choice of the teaching strategy. An actual node and its neighbourhood can be zoomed, an existing graph and its spanning tree can replace this node or future nodes of a path can be rearranged by modifying the teaching strategy: Thus the resulting learning path could include parts where different teaching strategies were applied. Furthermore, we plan a strong evaluation of the link weights assignment process for implementing different learning modes.

We believe that the use of the Lesson Mapper could be of great potential for preparing classes with more emphasis arranging algorithms, explanations, examples, graphics, and animations more carefully.

7. References

- [1] Baloian, N., Hoppe, H.U., Kling, U.: Structural authoring and cooperative use of instructional multimedia material for the computer-integrated classroom. Proceedings ED-MEDIA 95 Conference, Graz, Austria.1995, pp. 81-86.
- [2] Baloian, N., Hoppe, H.U., Luther, W.: Structuring Lesson Material to Flexibly Teaching in a Computer Integrated Classroom. GI-Workshop der Fachgruppe 'Intelligente Lehr-/Lernsysteme', Dortmund, Germany. October 8-11, 2001. Research Report 763, University of Dortmund, ISSN 0933-6192, pp. 187-194.
- [3] Baloian, N., Pino, J. A., and Motelet, O.: Collaborative Authoring, Use, and Reuse of Learning Material in a Computer-Integrated Classroom. J. Favela and D. Decouchant (eds.): CRIWG 2003, LNCS 2806, pp. 199-207.
- [4] Chaffin, R., Douglas, H.: The similarity and diversity of semantic relations. *Memory and Cognition* vol. 12, No. 2, 1984, pp. 134-141.
- [5] Colazzo, L., Molinari, A.: To see or not to see: tools for teaching with hypertext slides. Proceedings of the ED-Media 95. Graz, Austria, June 17-21, 1995, pp. 157-162.
- [6] Elklund, J., Brusilovsky, P. Schwartz, E.: Adaptive Textbooks on the World Wide Web. Proceedings Ausweb97, 3rd Australian World Wide Web Conference, July 5-9, 1997. (<http://ausweb.scu.edu.au/proceedings/elklund/paper.html>)
- [7] Hoppe, H.U.; Gassner, K.; Mühlenbrock, M.; Tewissen, F.: Distributed visual language environments for cooperation and learning - applications and intelligent support. *Group Decision and Negotiation* (Kluwer), Vol. 9 (2000), pp. 205-220.
- [8] Graf, F., Schneider, M.: IDEALS MTS – A Modular Training System for the Future. In Proceedings of ED-Media 98, Freiburg, Germany, June 20-25, 1998 (<http://www.igd.fhg.de/~schneider/papers/EdMEdia98.pdf>)
- [9] Halff, H.M.: Curriculum and instruction in automated tutors. In Polson, M.C., Richardson, J.J. (Eds.). *Foundations of Intelligent Tutoring Systems*. Hillsdale NJ: Lawrence Erlbaum Associates, 1989.
- [10] Hoppe, H.U., Luther, W., Mühlenbrock, M., Otten, W., Tewissen, F.: Interactive Presentation Support for an Electronic Lecture Hall. In *Advanced Research in Computers and Communications in Education*, G. Cumming et al. (Eds.). IOS Press 1999, pp. 923-930.
- [11] Hoppe, H.U., Gassner, K.: Integrating Collaborative Concept Mapping Tools with Group Memory and Retrieval Functions; in Stahl, G. (ed.): *Computer support for collaborative learning: Foundations for a CSCL Community*. Lawrence Erlbaum, Hillsdale, New Jersey USA 2002, pp. 716-725.
- [12] IEEE P1484.12: Learning Object Metadata Working Group. Draft Standard for Learning Object Metadata. An unapproved draft of a proposed IEEE Standard, 2000. (<http://ltsc.ieee.org/wg12/>)
- [13] Macias, J. A., Castells, P.: An Authoring Tool for Building Adaptive Learning Guidance Systems on the Web. <http://astreo.ii.uam.es/~atlas/perseus/AMT2001.pdf>
- [14] McCalla, G.: The search for adaptability, flexibility, and individualization: approaches to curriculum in intelligent tutoring systems. In Jones, M., Winnie, P. (Eds.). *Adaptive Learning Environments*. Springer-Verlag, NATO ASI Series, 1992, pp. 123-143.
- [15] Murray, T.: Authoring Knowledge Based Tutors: Tools for Content, Instructional Strategy, Student Model, and Interface Design. *Journal of Learning Sciences*, vol. 7, no. 1, 1998, pp. 5-64.
- [16] Norman, K.: HyperCourseware for assisting teachers in the Interactive Electronic Classroom. *Technology and Teacher Education Annual*, 1994, pp. 473-477.
- [17] Seeberg, C.: *Modulare Wissensbasen zur Erzeugung adaptiver und kohärenter Lehrdokumente*. Doctoral dissertation. Department of Computer Science, Technical University of Darmstadt, 2001.
- [18] SPSS 8.0. for Windows. Brief Guide. Prentice-Hall, Upper Saddle River, New Jersey USA, 1998.
- [19] Trigg, R.: A Network-based approach to text handling for online scientific community. PhD. dissertation. Department of Computer Science, University of Maryland, 1993.