A Concept Graph Editor for Computer-Aided Learning

Graham Horton, Richard Grillenbeck, Florian Kraus [*]

Abstract

The situation at a University Computer Science Department, where students are computer-literate and have access to workstations, makes computer-assisted teaching very attractive. Java is for various reasons, including portability, web compatibility and object orientation, the ideal choice for the implementation of learning programs. Concept graphs provide a visual representation of information which offers several advantages over traditional text-based sources. In particular, by emphasising the relationships between ideas, they deepen the student's understanding of the information and thus support the learning process. We present a Java-implemented software tool for the creation and viewing of concept graphs and discuss an application from an area of Computer Science teaching.

1 Concept Graphs

Concept Graphs are a graphical paradigm for the representation of information. Formally, they are characterized as directed graphs, with annotated nodes and edges, in which individual ideas and topics are contained in the nodes of the graph, while the edges serve to represent the relationships between these ideas. The information contained in nodes is usually strongly abbreviated, consisting often of just a single statement or key word, while the edges are annotated by just a single word. Thus concept graphs serve as an overview or summary tool, rather than a fully developed document. There are many references to concept graphs available on the web: we present a small selection here [1,2,3]. Information on our Java concept graph tool can be found at [4].

As a tool for information representation and communication, the graphical approach has many advantages over classical media, (usually plain, linear text), including

- Easier recognition of relationships between concepts, since these relationships are explicitly made visible as links (usually lines or arrows) between individual subjects. By contrast, in written texts or oral presentations, relationships between ideas are often less emphatically presented, often only implicitly, and sometimes – unfortunately – not at all.
- Easier recognition of overall structure of the information, as all the information is presented simultaneously, rather than sequentially.

[*]Lehrstuhl für Rechnerstrukturen (Prof. Dr. M. Dal Cin), Institut für Informatik, Universität Erlangen-Nürnberg, Martensstraße 3, 91058 Erlangen. email: `graham@informatik.uni-erlangen.de`

- Easier identification of the role or function of each item of information. This is achieved by the use of coding (of colours and geometric forms). For example, by always presenting examples in a red ellipse or definitions in blue rectangles, these roles become identifiable with a single glance.

Concept Graphs focus on ideas and relationships between them without worrying about the syntax and semantics of language, such as grammar or sentence construction. This focus makes Concept Graphs significantly easier to write and read than standard text. The typical linear, single-colour presentation of information (such as this paper) is not at all conducive to the effective communication of knowledge; Concept Graphs can achieve this significantly better.

The preparation of presentations and documents is greatly facilitated by the use of concept graphs, since

- ideas can be ordered easily and placed in context,
- the overall structure can grow dynamically and adaptively.

Creativity is enhanced by adding a visual element to thinking. By adding visual stimuli the right (creative) hemisphere of the brain is used in conjunction with the left, thereby taking advantage of both sides of the brain.

It is well known that it is easier to understand and remember information that is supported by visual elements such as colour, emphasis, and geometric shapes. Concept Graphs provide a medium in which these elements can be easily and naturally used. In general, the more interesting and "prettier" a concept graph is, the more effective it will be in communicating the information contained in it.

2 The Java Program

We are at present developing a program, written in Java, for the creation, editing and viewing of concept graphs *thus eliminating disadvantages of a paper-version of Graphs: Pre-Planning to fit the paper size; editing could reqire a complete redrawing; intricate related graphs could require a third dimension.*

Apart from the obvious functionality, the program has two special characteristics. The user may define reusable own "palettes" of node and edge types, each of which may be given a characteristic graphical appearance, in order to provide a consistent coding for the various concept and relationship types that is immediately recognisable across a set of graphs. Second, the program permits nodes to contain an variety of different types of content, including

- Text and graphics
- URL-links to web sites in the Internet
- links to other application programs or concept graphs

Apart from the advantages of an object-oriented language, we chose Java because of its portability between PCs and UNIX workstations, the simplicity of programming the graphical interface, and the integrability into a web site, which is planned for the future.

3 An Application in Computer Science Education

We have observed that finding coherence and connections between the different fields of Theoretical Computer Science seems to be one of the main difficulties for our students. Furthermore, it is often not clear to them that there are connections to other areas of Computer Science which are useful to notice and learn as well.

In conversations with examination candidates, we found that formalism and the complexity of the field often discourage and demotivate students, who then avoid deeper and further work on the material. Students know that there have to be links from theory to applications but they can't see them.

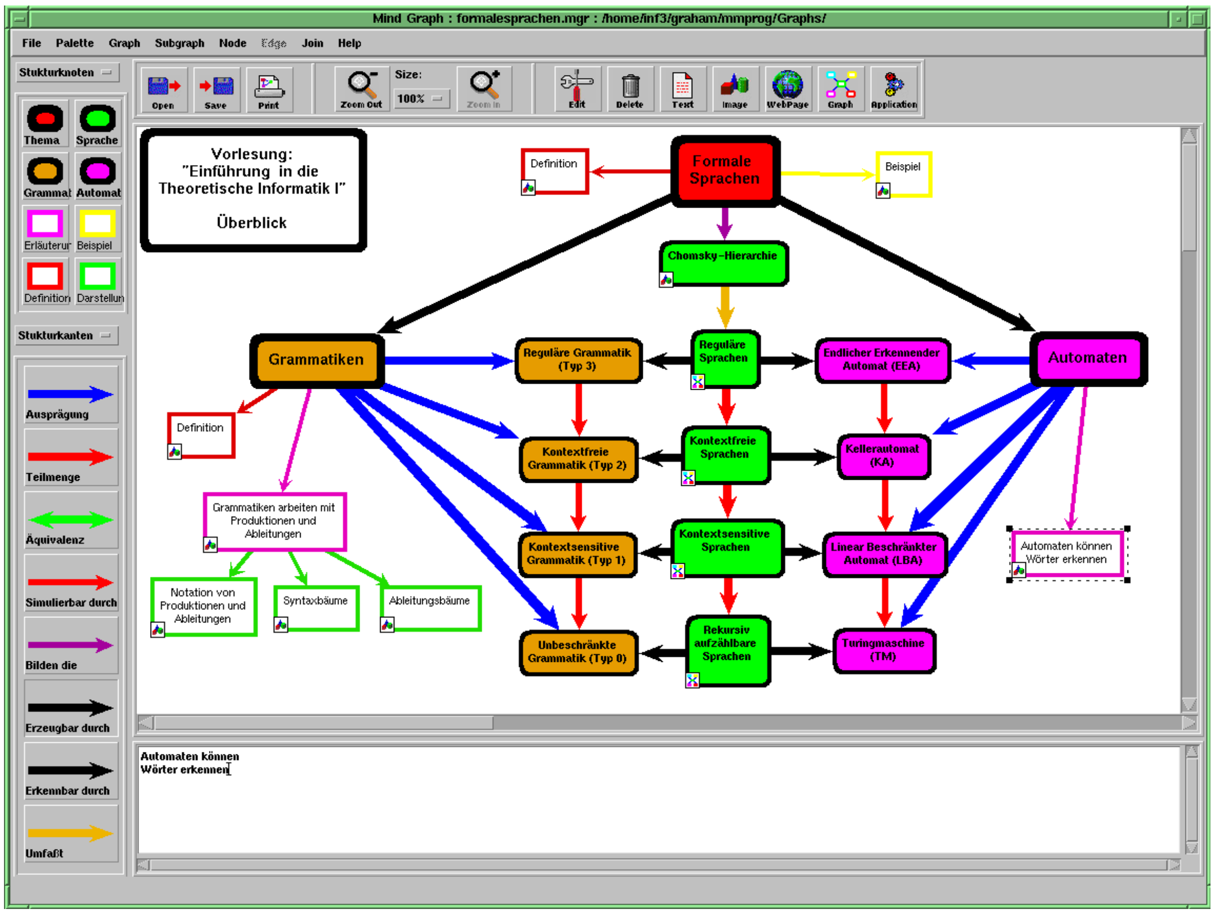
For example there are strong connections between the fields of Theory of Complexity, Automata Theory and Formal Languages. The individual fields can be more easily understood when these relationships are made clear. However, students seem to be so occupied with rote learning just the contents of lemmata and proofs that they cannot re-create correlations and associations between the fields (genuine learning!). They remember isolated facts without understanding the references to applications and other fields of computer science. We therefore decided that an overview of the above topics from Theoretical Computer Science should be our first application.

Since one of the the topics in the first part of the course about Theory of Computer Science is Formal Languages we built a Concept Graph showing the relations between the different Chomsky types of languages and the corresponding accepting models of automata. The layout itself is easy to memorize owing to its symmetric structure (and created the "aha-effect" on some participants). We asked our students to do further work on the lower levels. While creating the Concept Graphs, our students often discovered new relationships and connections that that had been missed during the corresponding lecture.

We found that working with the tool resp. Concept Maps engages the student in building and creating connections as well as just writing down contents, enabling them to learn deeper and faster. A Concept Graph creates its own momentum of (re-)finding associations. Drawing the Concept Graph for context-free languages students are engaged to review not only the content of the lecture but they concentrate on relations as well: what could be an examlpe for this type of language; what is the definition of context-free grammars; can we find a proof for the pumping lemma; what are subsets of context-free languages; give us an equivalent automata for a given problem; which automata accepts a certain language; look for a conclusion of the undecidability of the halting problem; what is the complexity of an given algorithm (which solves the word problem e.g.); is there a use of a given idea for a real application. The latter is as we found out sometimes the most powerful motivation for learning.

In emphazising(?) these kinds of relationships students learn that theory is a basis for some stuff they get to know in other lectures and that they can use theory as a tool for understanding, einordnen, bewerten and organizing topics of fields they didn't see in correlation to theory lessons. Some typical relations needed for creating concept graphs for computer science and shown below are: "proof", "follows", "equivalent", "subset".

A remarkable effect for us teachers is that we now also tend to look for relations and associations preparing a lesson in addition to think of how we could talk about an issue and what topics we should choose at all. We ask ourself questions like "How can I combine this topic with former lessons and/or other topics", "what is an application of that issue"


Mind Graph : formalesprachen.mgr : /home/inf3/graham/mmprog/Graphs/
File
Palette
Graph
Subgraph
Node
Edge
Join
Help
Stukturknoten
Open
Save
Print
Zoom Out
Size:
100%
Zoom In
Edit
Delete
Text
Image
WebPage
Graph
Application
Thema
Sprache
Grammat
Automat
Erläuterur
Beispiel
Definition
Darstellun
Stukturkanten
Ausprägung
Teilmenge
Äquivalenz
Simulierbar durch
Bilden die
Erzeugbar durch
Erkennbar durch
Umfaßt
Vorlesung: "Einführung in die Theoretische Informatik I" Überblick
Definition
Formale Sprachen
Beispiel
Chomsky-Hierarchie
Grammatiken
Reguläre Grammatik (Typ 3)
Reguläre Sprachen
Endlicher Erkennender Automat (EEA)
Automaten
Definition
Kontextfreie Grammatik (Typ 2)
Kontextfreie Sprachen
Kellerautomat (KA)
Grammatiken arbeiten mit Produktionen und Ableitungen
Kontextsensitive Grammatik (Typ 1)
Kontextsensitive Sprachen
Linear Beschränkter Automat (LBA)
Automaten können Wörter erkennen
Notation von Produktionen und Ableitungen
Syntaxbäume
Ableitungsbäume
Unbeschränkte Grammatik (Typ 0)
Rekursiv aufzählbare Sprachen
Turingmaschine (TM)

Automaten können
Wörter erkennen

Figure 1: Main Window of the Java Concept Graph Editor.

and, even more important, "what relations are there I didn't tell yet" or "can I find more, other associations which could help to understand, to einordnen". We surely empower or students to search for relations as well and to create associations which work for themself to learn and remember better. Talking with our students we experienced they appreciate this approach very much.

As small assignments or homworks we plan completions of a Graph. By doing so, we ensure students understand the fundamental ideas and relationships of a topic, and are not reciting memorized text passages. One can use these Graphs also as a means for evaluating ones lecture: Just comparing the different graphs gives you a very quick overview about well known issues and less known topics and relations. Analyzing the homework and discussions about the graphs leed towards an deeper insight which subject should I handle in a different manner to gain better learning results.

4 Future Plans

We plan to evaluate the program not only in Theoretical Computer Science, but also in other courses. A field study will be designed for testing how the tool can be used efficiently in all kinds of seminars, lectures, workshops and classes not only at university but also in industrie. In these environments both the descriptional and structuring power of Concept Mapping will be examined.

We expect to achieve most success in areas characterized by complex relationships, as opposed to merely learn-by-heart subjects. In this context we will create an “applet” version of the program, which will serve purely as a viewing, rather than an editing tool. This may also prove to be useful in a distance learning context.

We will provide automatic generation of standard formats from the concept graphs such as ASCII text, LaTeX, and HTML. These will include hyperlinks and graphics as appropriate. This will support the preparation of standard documents, where the concept graphs are used in the preparatory, “brainstorming” and structuring stages.

We will provide algorithms for the automatic layout of graphs, in order to improve readability. This function is especially important in conjunction with a further planned option: the selective hiding of node and edge classes.

Lastly we plan to automate the comparison of graphs. This may aid the teacher in checking for individual and common misunderstandings of both concepts and relations among the students.

References

[1] J. Lanzing: The concept mapping home page. `http://www.to.utwente.nl/user/ism/lanzing/cm_home.htm`

[2] University of Victoria: Learning skills program. `http://www.coun.uvic.ca/learn/program/hndouts/map_ho.html`

[3] P. Boyer: Concept mapping. `http://131.210.1.4/staff/boyer/100/bios.100.html`

[4] G. Horton: The MindGraph home page. `http://www3.informatik.uni-erlangen.de:1200/Projects/cg/html`