Adaptive Knowledge-Based Visualization for Accessing Educational Examples

Peter Brusilovsky, Jae-wook Ahn, Tibor Dumitriu, Michael Yudelson
School of Information Sciences, University of Pittsburgh
{peterb, jaa38, mvy3}@pitt.edu

Abstract

A number of research teams are working to organize personalized access to the modern repositories of educational resources. The goal of personalized access is to help students locate resources that match their individual goals, interests, and current knowledge. The project presented in this paper is focused on the least explored way of personalized access – adaptive visualization. Here, we present the NavEx ADVISE visualization system, which provides personalized access to a repository of educational examples. The system combines spatial, similarity-based visualization with adaptive annotations of resources. The spatial layout and the adaptive annotations are generated using a knowledge-based indexing of examples with domain concepts.

1. Introduction

Dedicated repositories of educational materials such as educational digital libraries (DL) and pools of reusable learning objects are now accumulating a large volume of educational resources. The abundance of resources available to students has created a new challenge—how to help students locate resources that match their individual goals, interests, and current knowledge.

We first faced this challenge in our WebEx project, when we developed a repository of about a hundred annotated program examples for an introductory programming course [2]. The repository contains lines of example code that has been annotated by teachers. Once an example is selected, the WebEx system allows students to explore it interactively inside a Web browser, by clicking on the annotated lines of code and reading the teacher's explanations. But the question is—how can the most relevant examples in a repository be located when there are dozens of examples accessible at any one time?

Our first solution was to apply the adaptive navigation support approach by using NavEx [9], an adaptive interface for the WebEx system. It provided a list of links to all examples and augmented each link with an adaptive icon that visualized the status of the example, adapted to the current state of the student's knowledge and history of past interactions (Figure 1). These icons help students to: a) distinguish new examples from examples that have already been partially or fully explored in the past; as well as to b) distinguish examples that are ready to be explored from examples that demand prerequisite knowledge the student lacks.

The problem that we are addressing in this paper is how to help students to locate examples that are relevant not only to their current knowledge, but also to their current learning goal. What if a student has problems with understanding a specific example and wants to read explanations for one or more similar examples? Or, in contrast, what if a student fully understands the programming constructs explained in a specific example and now wants to explore an essentially different example to better cover the content of the course? Unfortunately, the ordered list of links to examples used in NavEx offered no help in selecting similar or dissimilar examples.

Past research on information visualization suggests that goal-based selection of documents is better supported by two-dimensional visualization than by a one-dimensional list. In particular, the Lighthouse system [5] applied spatial similarity-based visualization and relevance marking to assist users in finding documents most relevant to their search goals. Due to the nature of similarity-based visualization, similar documents were positioned close to each other and dissimilar far from each other. This allowed the users to visually receive better guidance in selecting documents than is usually provided by a ranked list.

This paper presents our attempt to implement ideas of adaptive visualization in the context of personalized access to a repository of annotated program examples. We present our new visualization interface, NavEx ADVISE, which combines spatial similarity-based visualization with adaptive annotations. The similarity-based layout allows students to easily locate the most similar and dissimilar examples visually, helping the student to select examples relevant to their current learning goal. Adaptive annotations support knowledge-based and progress-based adaptation. NavEx ADVISE was implemented using ADVISE 2D, a visualization tool developed by the authors.

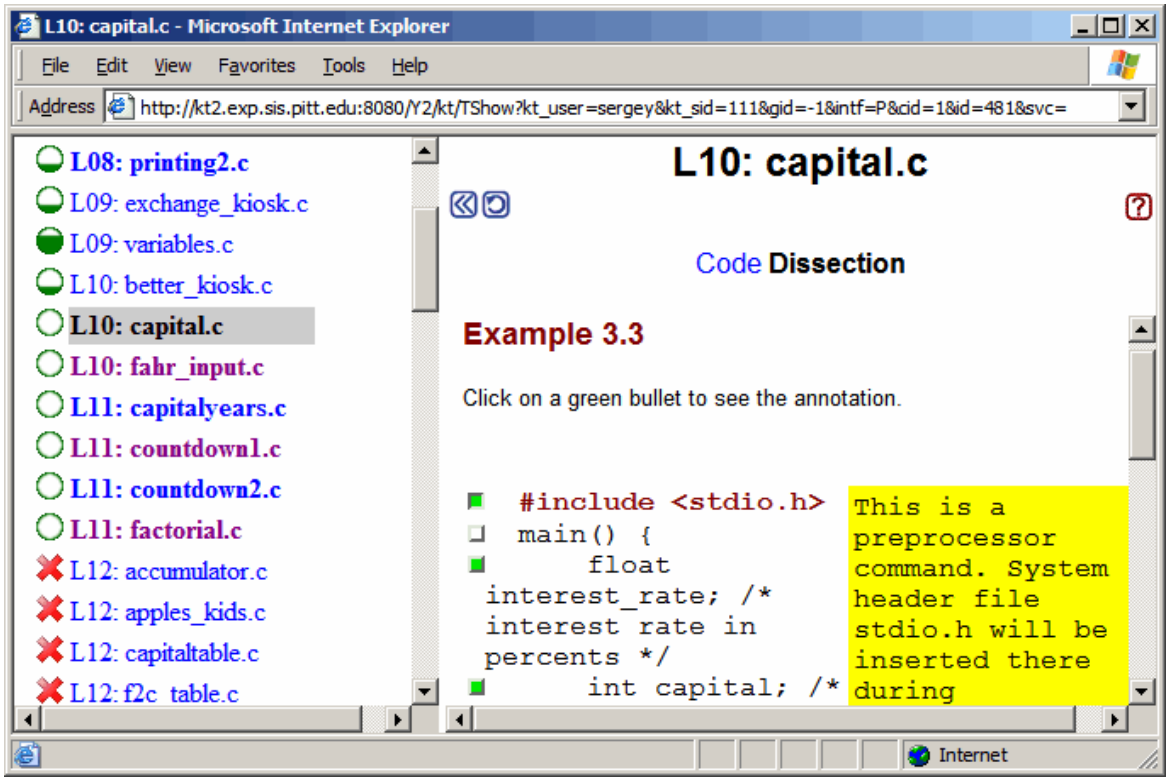

Figure 1 Adaptive navigation support in NavEx system helps students choose the example to browse by augmenting each example link with an adaptive icon that visualizes the status of the example

The need to provide personalized access to program examples served as the motivation to develop ADVISE 2D; however, we attempted to make ADVISE 2D generic enough so that it can support a range of similar information access needs. The paper focuses on both NavEx ADVISE, a specific adaptive visualization system, and ADVISE 2D, a tool that we have created to support this and similar projects. The next section presents the user's view of the adaptive visualization interface in NavEx ADVISE. The remaining sections present ADVISE 2D and its application to guiding students through examples with personalized access.

2. Accessing program examples with adaptive visualization

The visualization-based interface for accessing examples organizes and displays the whole repository of examples on a two dimensional example map. Figure 2 shows the map with its examples distributed according to similarity. Each rectangle represents an example. The distance between two documents on the map represents how similar they are to each other. If their knowledge-level content is similar, they are placed closer to each other, but if dissimilar, they are placed farther from each other. It is important to know that the visualization is based on knowledge-based similarity. The distance between two examples is determined by comparing the set of programming concepts presented by these examples.

Each example bears an adaptive icon that shows the relevance of this example to the user's current knowledge and completion of progress through the example. If the user is not ready to access the example, due to a lack of prerequisite knowledge, a red X is displayed (Figure 2). If the user is ready to access an example, then a green bullet is shown. The fullness of the bullet approximates the user's progress within the example. An empty bullet shows a completely new, but ready-to-be explored example. A filled bullet denotes a fully explored example. The current icon is determined by the NavEx system, which compares the past history of the student's interaction with the example to the concept-level model of knowledge maintained by NavEx.

The spatial layout and adaptive annotations help the users to locate the most relevant examples. To start working with an example, the user double-clicks the example box, which causes a WebEx window to open for interactive exploration of the selected example.

To better explore the set of examples, users can manipulate the visualization: zooming in or out with a slide bar or panning the screen in four directions. In addition, users can display lines between similar documents and exact similarity values between pairs of documents.

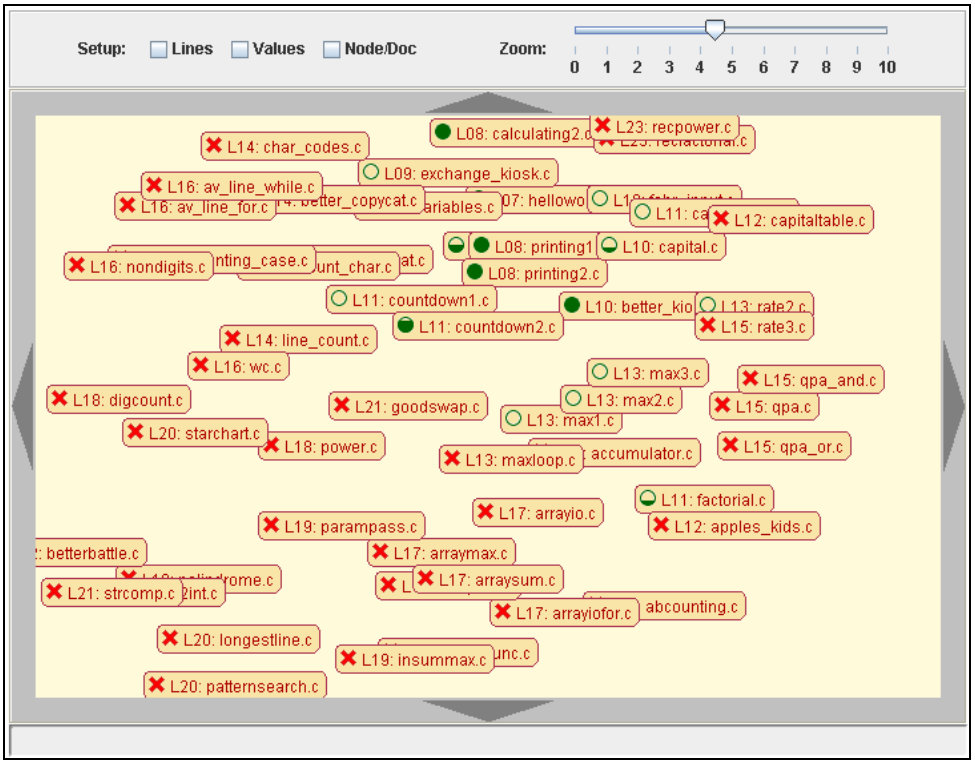

Figure 2 ADVISE 2D helps students choose the most relevant example by combining adaptive navigation support with spatial 2D visualization of the example space

These functionalities are important for dealing with an abundance of documents that may potentially be displayed in a relatively small window. Finally, the example names sometimes make the display too crowded to grasp the whole picture. In such situation, the rectangles with document titles can be reduced to small icons, as shown in Figure 3. In this view, the example name is only displayed when a user moves a mouse cursor over its icon.

3. Similarity-based visualization in ADVISE

The spatial visualization of examples in NavEx was produced with ADVISE 2D – one of the tools developed in our lab for the ADVISE project. ADVISE (ADaptive VISualization for Education) is a suite of web-based, personalized document-visualization systems for educational purposes. The goal of the ADVISE project is to help students find the most relevant educational materials, through personalized visualization. ADVISE suite includes three systems with different perspectives and approaches. ADVISE 2D and ADVISE 3D are similarity-based document space visualization systems for two- and three-dimensional spaces, respectively.

They provide spatial document maps where documents are distributed according to their similarity values. (We refer to every educational object in this system as a document.) ADVISE VIBE is an implementation of the VIBE [7] document visualization approach. It calculates similarities between documents and POIs (Point Of Interest) or concepts and determines document positions relative to the positions of the POIs. ADVISE tools are customizable to different application contexts and are available from the project home page (http://ir.exp.sis.pitt.edu/advise)

ADVISE 2D is a generic tool that provides personalized access to documents using a similarity-based visualization approach known as spring modeling (see section 3.4 for details). Icons which represent similar documents are positioned relatively closer to each other on a two dimensional map and while those which are different would be placed more distantly on the map. By consulting the distribution of the icons and the positions of each of them, users can make some general assumptions about the contents of a document even before actually opening it, understand the relationships between documents in terms of their contents, and see the overall picture of the documents in the corpus. The adaptive similarity-based visualization is produced in several steps: representing document contents as vectors, loading the vectors into the application, calculating similarities among document vectors, deciding document positions on a two dimensional space based on these similarity values, and finally presenting visual representation to users.

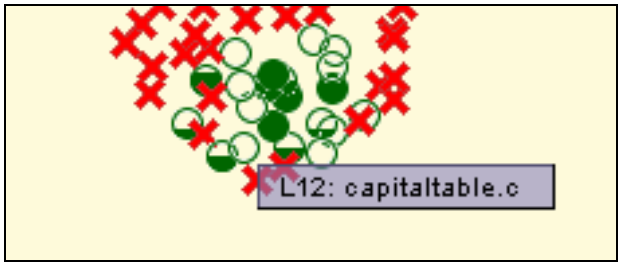

Figure 3 Simple display for documents

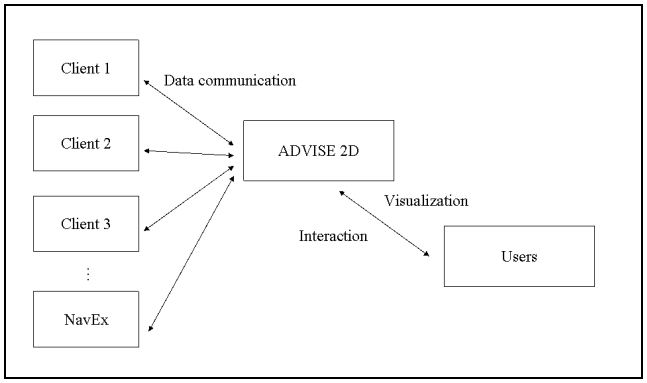

Figure 4 ADVISE 2D architecture

ADVISE 2D was designed as a context-independent Web-based visualization tool. It can visualize data that has been loaded from several different applications. Figure 4 is the architecture of ADVISE 2D. For the purpose of this project, ADVISE 2D was tuned to work with a repository of WebEx examples. The remaining part of this section describes ADVISE 2D and the next section presents use of the system for personalized access to examples.

3.1. Document representation

Documents in ADVISE 2D are represented as weighted term vectors in order to calculate the similarities and to place them in appropriate positions [8]. If a document is a full-text tutorial or a lecture slide, terms from the document are extracted and stored in a vector, which represents the document. If the document is a code example, language constructs are stored in the vector. Therefore, the whole corpus can be represented as a matrix with documents in its rows and terms in its columns. If the corpus contains a total of M documents and the total number of terms in it is N, an M by N matrix is constructed (Figure 5). In most cases, few documents would even come close to containing every term in the corpus, so the matrix tends to be very sparse, with a lot of 0's, which means that there is no occurrence of another corresponding term in the document.

The value of vector components can be binary or weighted. A well known TF (term frequency) or TF-IDF (term frequency multiplied by inverse document frequency) scheme can be applied for term weights. TF means the frequency of a term in a document and IDF is an inverse of the number of documents which have the corresponding term (Equation 1).

	$Term_1$	$Term_2$	$Term_3$	…	$Term_i$
Doc_1	w_{11}	w_{12}	w_{13}	…	w_{1j}
Doc_2	w_{21}	w_{22}	w_{23}	…	w_{2j}
…	…	…	…	…	
Doc_i	w_{i1}	w_{i2}	w_{i3}	…	w_{ij}

Figure 5 Document-term matrix

$$tfidf(d,t) = tf(d,t) \cdot \log\left(\frac{|D|}{df(t)}\right)$$

tf(d,t): number of occurrences of term *t* in document *d*
df(t): number of documents where term *t* appears
$|D|$: total number of documents in the corpus

Equation 1 TF-IDF weighting

3.2. Similarity calculation

The cosine similarity coefficient (Equation 2) was used for calculating inter-document similarities. This relates to the cosine angle of the two vectors *x* and *y,* and ranges from 0 (the two documents being completely dissimilar) to 1 (they are identical).

$$Sim(x,y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

Equation 2 Cosine similarity coefficient

3.3. Data communication

ADVISE 2D was designed to work seamlessly with web-based materials. It can spatially map the materials by their contents, explore the mapping, and open their contents by connecting to their URL's. For this last purpose, it was implemented as a Java applet and can launch within web browsers. It loads three types of data to make the visualization: a) document vectors with titles, b) list of terms, and c) document URL's. They are fed into the applet in two different ways: using <PARAM> tags in HTML- and XML-based communication with a server. The first method works in a static manner, which cannot dynamically update the contents being visualized but the second method makes it

possible for the applet to send a request to the server for new data, enabling the applet to update the current visualization. The applet and the server communicate in a predefined XML-formatted protocol.

3.4. Spatial mapping with the spring model

ADVISE 2D organizes and visualizes documents based on the spring modeling algorithm. The spring modeling algorithm or FDP (Force Directed Placement) is a heuristic approach to graph drawing, based on a hypothetical physical (mechanical) system in which the graph's edges are replaced by springs while the vertices (nodes) are replaced by rings. The springs attract the rings if they are too far apart and repel them if they are too close [1; 4]. It can be used to sort randomly placed nodes into a desirable layout that satisfies the aesthetics of visual presentation.

$$F_s = C_s \log(d / C_d)$$
$$F_r = C_r / d^2$$

Equation 3 The forces formula for the spring model

The forces acting on every node include spring force and repulsion force. The resultant of these forces can be calculated and, under the influence of spring force between connected nodes and repulsion force between unconnected nodes, the graph will automatically adjust itself until the system reaches a stable state. It calculates this resultant of forces by including the spring and repulsion forces that act on every node, in an iteration of the loop until the graph reaches a stable state [6]. Equation 3 shows the calculations of spring force and repulsion force in this spring model. *Cs*, *Cd*, *Cr* are constants such as spring length, spring stiffness, spring type, and initial configuration, which control the forces acting on nodes and their movements.

In ADVISE 2D, both spring forces and repulsion forces were considered to show their relationships in terms of similarities. When a system launches and reads in the data needed, it randomly places every document on the map. The spring algorithm begins from this state and iterates until it reaches a stable state. In this stable state, the documents are arranged according to their similarity values and are then finally visualized on the map.

4. Customizing ADVISE 2D to work with program examples

To achieve the goals of our current project, our context-independent system ADVISE 2D was customized to provide adaptive knowledge-based visualizations of annotated program examples. The customized system is referred to as NavEx ADVISE since both the spatial representation and the adaptive icons are generated using the knowledge-based representation of program examples produced by the NaxEx system [9]. As we mentioned in the introduction, NavEx was designed to provide personalized access to code examples through adaptive navigation support. NavEx provides guidance regarding relevant examples that students should or shouldn't explore by displaying an adaptive icon (Figure 7). To generate this icon dynamically for each example, NavEx takes into account the programming knowledge presented by each example, the current state of the student's knowledge, and his/her past interaction with the examples.

4.1 Knowledge-based example indexing

The key to this functionality is the knowledge-based indexing of each annotated example with a set of concepts from the C-programming domain. The concepts used are C language constructs such as decl_var, void, include, main_func, etc. The indexing is done automatically by a domain-specific parser. After all examples are indexed, concepts of each of the examples are split into prerequisites and outcomes. Outcomes are concepts that are illustrated by that example. Prerequisites are concepts that should be learned before exploring the example. The splitting is automatic, but it is driven by the teacher's individual approach to concept- sequencing within the course. To tune the indexing to a specific teaching approach, a teacher must provide a set of representative examples for each course lecture. More details about the indexing procedure can be found in [3]. The result of concept indexing and division is shown in Figure 6.

This knowledge-based example indexing was used by NavEx ADVISE to produce a spatial layout for the repository of examples. The concept index of each example was converted into a term vector as explained in section 3.1. To produce a course-independent layout, the indexing does not distinguish prerequisite and outcome concepts, but takes into account how frequently each concept is found in each example (and some concepts can present in several places within the same example). The resulting layout is based on deep knowledge-level similarity between the examples.

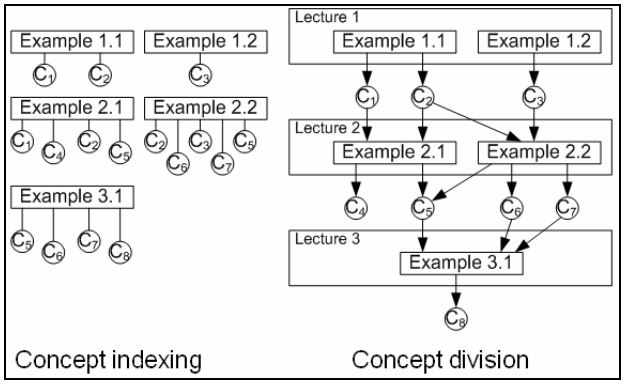

Figure 6 Concept indexing and division

4.2 The adaptive annotation of examples

The adaptive annotation of examples in NavEx ADVISE was produced using the same icons and algorithms as in the original NavEx system. Thus the same example in the NavEx list and on the NavEx ADVISE map bears an identical annotation at the same moment in time. Both views provide two kinds of annotation: progress-based and prerequisite-based. The progress-based annotation shown as a partially-filled green bullet (Figure 7) is calculated as simply the percentage of example code lines already explored by the student, compared to the total number of annotated lines in this example. Computation of prerequisite-based "readiness" for an example is based on a concept-level model of student knowledge. The example is considered ready for exploration if all of its prerequisite concepts are already known to the student to a specified extent. The system approximates the student's level of knowledge for each concept by monitoring student activities within the system (i.e., example exploration) and by consulting a centralized student model that collects evidences of student knowledge from multiple sources (i.e., reading a tutorial section or answering a quiz).

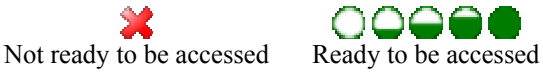

Figure 7 Annotation clues in NavEx

The adaptive annotations support a natural "flow" for exploration of concepts and examples. Once a student starts to work with examples, only those that are ready to be accessed will have no prerequisites. As a user explores an example and reviews a certain number of annotations (defined by a dynamic threshold), an example is considered to be completed (although their progress may remain below 100%). At the same time, all of the associated concepts are marked as learned. If there exist some examples whose prerequisites are now all marked learned, these open up to be explored and the flow continues on. For more details refer to [3].

Conclusions

This paper presents our attempts to apply adaptive, knowledge-based visualization to guide students of introductory programming courses to most of the items in a repository of educational examples. We presented the NavEx ADVISE visualization system, which combines a spatial similarity-based visualization with the adaptive annotation of educational objects. Both the spatial layout and the adaptive annotations are generated by using a knowledge-based indexing of program examples with programming concepts. The system was developed by the targeted customization of the generic ADVISE 2D visualization tool, which had previously been developed in our lab.

Both the NavEx and NavEx ADVISE systems are currently being used in an introductory programming course. Our earlier exploration of NavEx demonstrated that students appreciate the guidance provided by adaptive annotations and that it encourages students to explore significantly more examples [9]. Our current challenge is to determine the value of adaptive visualization in its role of supporting student access to educational examples.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 0447083.

References

[1] Battista, G. D., Eades, P., Tamassia, R., and Tollis, I. G., "Algorithms for drawing graphs: an annotated bibliography", *Computational Geometry: Theory and Applications* **4,** 1994, pp 235-282.

[2] Brusilovsky, P., WebEx: "Learning from examples in a programming course", In: Fowler, W. and Hasebrook, J. (eds.) *Proc. of WebNet'2001, World Conference of the WWW and Internet*, Orlando, FL, AACE, 2001, pp 124-129.

[3] Brusilovsky, P., Yudelson, M., and Sosnovsky, S., "An adaptive E-learning service for accessing Interactive examples", In: Nall, J. and Robson, R. (eds.) *Proc. of World Conference on E-Learning, E-Learn 2004*, Washington, DC, USA, AACE, 2004, pp 2556-2561.

[4] Eades, P., "A Heuristic for Graph Drawing". *Congressus Numerantium* **42**, 1984, pp 149-160.

[5] Leuski, A. and Allan, J., "Interactive information retrieval using clustering and spatial proximity", *User Modeling and User Adapted Interaction* **14**, 2-3, 2004, pp 259-288.

[6] Liu, X., Shizuki, B., and Tanaka, J., "Dynamic Parameter Spring Modeling Algorithm for Graph Drawing", In: *Proc. of International Symposium on Future Software Technology (ISFST2001)*, Zheng Zhou, China, 2001, pp 52-57.

[7] Olsen, K. A., Korfhage, R. R., Sochats, K. M., Spring, M. B., and Williams, J. G., "Visualisation of a document collection: The VIBE system", *Information Processing and Management* **29**, 1, 1993, pp 69-81.

[8] Salton, G., *Automatic Text Processing*, Addison-Wesley Publishing Co., Reading, MA.

[9] Yudelson, M. and Brusilovsky, P., "NavEx: Providing Navigation Support for Adaptive Browsing of Annotated Code Examples", In: Looi, C.-K., McCalla, G., Bredeweg, B. and Breuker, J. (eds.) *Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*. IOS Press, Amsterdam, 2005, pp 710-717.