Towards Interactive Learning by Concept Ordering

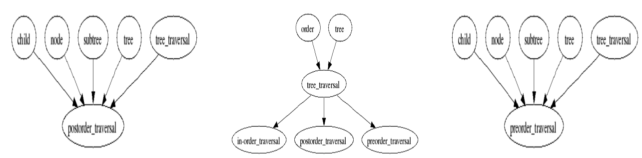
ABSTRACT

In this paper we present a visual education tool for efficient and effective learning. The toolkit is based on a simple premise: *simple concepts should be learned before the advanced ones.* We propose algorithms to automatically capture such pre-requisite dependence graphs among the concepts. Next, the concepts are arranged in a hierarchical structure and presented to the user through an visual interactive interface. The user, henceforth, guides the learning process, by selecting a target concept, exploring the associated learning graph, learning pre-requisite concepts, and repeating this process till the learning goal is reached. To measure usefulness and correctness of our approach, we conducted user study with 25 users. Overall, the feedback from users was encouraging. We believe that this effort is a positive step towards building user driven interactive learning systems.

1. SYNOPSIS OF THE APPROACH

Motivation: Traditionally, individuals have relied on books, newspapers and other printed material to learn. However, recent past has seen a radical shift in the learning paradigm with the advent of the Internet. In the presence of efficient and accurate search engines, an user searches a target concept on the web and learns it by following relevant (typically top 10) results. This method indeed looks attractive but it unfortunately lacks the structure provided by printed sources. For example, concepts in printed books are arranged in increasing order of difficulty and maturity. Any such ordering is missing from the Search–and–Learn paradigm. While reading about a concept using online resources, the user has to navigate back and forth between related, underlying or pre-requisite concepts. This navigation is ad-hoc without any principled ordering. However, online learning supports very effective browsing between concepts which is not possible in books. *In this paper, we develop a visual educational tool which aims to leverage the positive features of both learning approaches while mitigating the negative aspects of the individual approaches and helps the user learn effectively and efficiently.*

The tool is based on simple observations about learning i)learning any (fairly) advanced concept involves learning pre-requisite concepts and ii) outside of the classroom settings, individuals prefer focused learning and would not like to spend time on already known (redundant) or peripheral (non-relevant) concepts. We contend that both these criterion can be satisfied by arranging the concepts in hierarchical graph-like structure. To be more precise, there is a partial ordering between concepts, which corresponds to a directed acyclic graph (DAG). We employ graph theoretic algorithms to construct the graphs. The concept graphs are presented to the use in an interactive environment, where the user can see all the relevant concepts and can choose to explore non-redundant (previously unknown) concepts in detail, while ignoring known concepts.



(a) (b) (c)

Figure 1: Example showing the concept graphs for tree traversal algorithm

Before providing the brief summary of the algorithms, we present a concrete example. Assume that the user wants to learn post-order traversal. The corresponding learning graph is shown in Figure 1(a). However, by inspecting the graph, the user soon realizes that she needs to understand tree traversal. She clicks on the tree traversal node and the learning graph shown in Figure 1(b) is presented. The user learns tree-traversal and realizes that she can also learn pre-order and in-order traversal without learning any new concepts. Figure 1(c) is displayed when she selects pre-order traversal. This expository example illustrates the usefulness of our system, wherein the user needs an interactive environment to learn the target concept (post-order traversal) and at the same time also realizes that extra concepts can be learned without much effort.

Algorithm: Let C be a list of N concepts, and let $C[i]$ refer to the the i^{th} concept in the list. We assume the availability of a database of concept definitions for the given do-

Hypertext '07 Manchester, United Kingdom

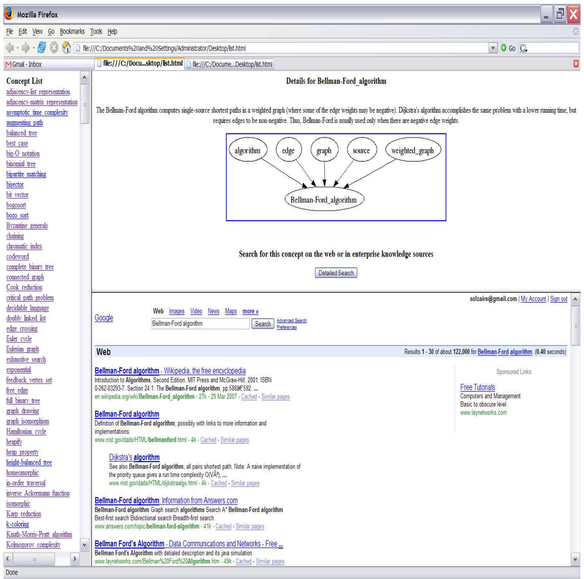


Figure 2: Snapshot of the web based visual tool

main[1]. Using this definition database, we construct the concept graph C_{DAG} for a particular domain. Our algorithm for constructing the concept DAG consists of two steps. In our first step, we construct a graph C_{graph} by making use of references between concepts in concept definitions. In the second step, we extract a concept DAG C_{DAG} from C_{graph}.

Step 1: We generate a weighted and directed concept graph $C_{graph} = (V, E)$ as follows. For every concept $C[i], 1 \leq i \leq N$, we create a vertex v_i in V. An edge is introduced from v_i to v_j iff concept $C[j]$ is mentioned in some definition of $C[i]$. We say that 'concept C_1 is mentioned in some text' *iff* the lemma form of C_1 matches the lemma form of some token in the text. The weight w_{ij} of an edge (v_i, v_j) is set to the number of mentions of $C[j]$ in the definitions of $C[i]$.

Step 2: Extracting a DAG C_{DAG} from C_{graph} requires the elimination of cycles from C_{graph}. There could be several sets of edges whose removal would lead to elimination of cycles. In our case of a weighted digraph, it makes sense to remove the set of edges with minimum total weight. The problem of removing minimum number of edges in a digraph is a standard problem studied in literature as the *minimum feedback arc-set problem* (FAS) [4]. Unfortunately, the problem is NP-Hard [3]. However, due to the wide applicability of the FAS problem, several heuristics have been proposed for it. For our task, we use a variant of a fast, effective and simple heuristic for FAS proposed by Eades, *et. al.* [1].

Interactive Visual Interface: Next, we outline the key components of our web-based interface. A snapshot of the interface is shown in figure 2. A ***concept list*** is the list of all concepts in a domain of interest. The list is displayed in the left pane of our tool. The user can select the concept of interest from the list and its associated learning graph is displayed on the right. ***Learning Graph*** shows the concept selected from the concept list and it's pre-requisite nodes. In addition to these ancestor nodes, we also show one level of descendants from our concept index. These are nodes which need the target concept as their pre-requisite. The learning graph is shown in top pane. The nodes of the graph are clickable and can also be used to select the target concept. The process is same as choosing a concept from concept list. of the tool. ***Concept Definition*** is the actual definitions of the selected concept. It is also shown in the top pane.

Overall Process: The concept list is obtained from either online glossaries or indexes present at back of book. Corresponding concept definition database is built. The list and database is used to derive the concept DAG. This process is performed in a offline fashion. The list is populated in the visual interface. The user selects a concept from the concept index and the associated learning graph is presented. Next, the user identifies the pre-requisite to be learned and clicks on the corresponding node. Now a new learning graph is displayed in real time. The key point to note here is that to maintain the context sensitivity of the learning graph the parent node (original target to be learned) is displayed. In our user study we found that this property helps the user stay focused and reduces navigational time. Once the user has learned all pre-requisites, she reads the definition of the target concept. Next, based on her interest, the user may search other data sources and refine or broaden her understanding.

Experiment: We designed and conducted an user study with 20 participants to evaluate the quality of our concept graphs. Each of the user was presented with learning graphs corresponding to 15 concepts and asked to mark redundant (false positive) and missing edges (false negatives). These numbers are used to calculate precision and recall. The average precision and recall were found to be .80 (standard deviation of .16) and .84 (.12) respectively.

Conclusions: In this paper, we presented a visual toolkit to facilitate the learning process by capturing the pre-requisite dependences between concepts in learning domains. These dependences are modeled hierarchically and presented to the user interactively. The user can recursively learn pre-requisite concepts to thoroughly understand target concepts of interest according to learning goals. We evaluated correctness and usefulness of our approach by conducting a moderately large user study. We used widely accepted evaluation metrics - precision and recall- to measure correctness and usefulness respectively. Overall, the users found the toolkit to be very useful for learning concepts in a principled and organized fashion. Currently, we are exploring the algorithms to incorporate each userŠs existing skill-set to set up a domain specific baseline will further result in focused learning and reduce learning time.

Due to lack of space we are not able to present the details of our approach. We will include the link to detailed technical report once the (blind) review process is over.

2. REFERENCES

[1] Peter Eades, Xuemin Lin, and W. F. Smyth. A fast and effective heuristic for the feedback arc set problem. *Information Processing Letters*, 47(6):319–323, 1993.

[2] Google. Advanced google search operators, 2002.

[3] R. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Communications*, pages 85–103, 1972.

[4] P. Slater. Inconsistencies in a schedule of paired comparisons. *Biometrika*, 48:303–312, 1961.

[1]We used "define:" query operator provided by the Google search engine [2] to build the such database