LONET: An Interactive Search Network for Intelligent Lecture Path Generation

NEIL Y. YEN, Waseda University
TIMOTHY K. SHIH, National Central University
QUN JIN, Waseda University

Sharing resources and information on the Internet has become an important activity for education. In distance learning, instructors can benefit from resources, also known as Learning Objects (LOs), to create plenteous materials for specific learning purposes. Our repository (called the MINE Registry) has been developed for storing and sharing learning objects, around 22,000 in total, in the past few years. To enhance reusability, one significant concept named Reusability Tree was implemented to trace the process of changes. Also, weighting and ranking metrics have been proposed to enhance the searchability in the repository. Following the successful implementation, this study goes further to investigate the relationships between LOs from a perspective of social networks. The LONET (Learning Object Network), as an extension of Reusability Tree, is newly proposed and constructed to clarify the vague reuse scenario in the past, and to summarize collaborative intelligence through past interactive usage experiences. We define a social structure in our repository based on past usage experiences from instructors, by proposing a set of metrics to evaluate the interdependency such as prerequisites and references. The structure identifies usage experiences and can be graphed in terms of implicit and explicit relations among learning objects. As a practical contribution, an adaptive algorithm is proposed to mine the social structure in our repository. The algorithm generates adaptive routes, based on past usage experiences, by computing possible interactive input, such as search criteria and feedback from instructors, and assists them in generating specific lectures.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Information filtering, Search process; H.3.5 [**On-line Information Services**]: Web-Based Services; H.5.2 [**User Interface**]: User-Centered Design

General Terms: Management, Performance, Algorithms, Standardization, Human Factors

Additional Key Words and Phrases: Learning object network, lecture path, interactive search, social network analysis, ranking, repository, distance learning, SCORM

ACM Reference Format:
Yen, N. Y., Shih, T. K., and Jin, Q. 2013. LONET: An interactive search network for intelligent lecture path generation. ACM Trans. Intell. Syst. Technol. 4, 2, Article 30 (March 2013), 27 pages.
DOI = 10.1145/2438653.2438668 http://doi.acm.org/10.1145/2438653.2438668

This work was jointly supported by the National Science Council of R.O.C. under contract no. 99-2511-8-468-004, the Research Center for Science & Technology for Learning of the University System of Tajwan, 2010-2012 Waseda University Advanced Research Center for Human Sciences Project "Distributed Collaborative Knowledge Information Sharing Systems for Growing Campus" and the 2010-2011 RONPAKU Program of JSPS (Japan Socety for the Promotion of Science).

Authors' addresses: N. Y. Yen (corresponding author), The University of Aizu, Tsuruga, Ikki-machi, Aizu-Wakamatsu City, Fukushima, 965-8580, Japan; email: neil219@gmail.com; T. K. Shih, National Central University, no. 300, Jhongda Rd., Jhongli City, Taoyuan County 32001, Taiwan (R.O.C.); Q. Jin, Waseda University, 2-579-15 Mikajima, Tokorozawa-shi, Saitama 359-1192, Japan.

DOI 10.1145/2438653.2438668 http://doi.acm.org/10.1145/2438653.2438668

1. INTRODUCTION

Research has shown that social network analysis can be a promising approach to clarify the complex interactions among participants, human users, and their associated information and knowledge as well, by quantifying potential interactive processes to an interconnected relation graph or network [Tolsdorf 1976]. As a result, it not only makes it easier to enunciate interpersonal information, but, furthermore, enables researchers to discover those indirect relationships by mining information associated to common intersections [Palla et al. 2007]. The concept can also be applied to every domain in which the interactions may occur (e.g., links between Web pages, citations between research papers), by considering customized factors and metrics [Kumar et al. 2002; Raghavan 2002]. Hence, from a perspective of knowledge sharing, it raises emerging issues about patterning collective intelligence [O'Reilly 2007].

In distance learning, a repository identifies not only a distributed storage for LO (Learning Objects), any digital resources for educational purposes, but also a common platform for it to be shared and reused. Thanks to Web 2.0, a growing number of Internet resources can be selected, assigning corresponding metadata [Van Assche et al. 2003], for the use in specific learning activities. However, the situation also brings an open issue on how a huge number of LOs can be managed in an efficient manner [Koppi et al. 2005]. As a preliminary solution, CORDRA (Content Object Repository Discovery and Registration/Resolution Architecture) was proposed to build an interoperable federation for storing and sharing SCORM (Sharable Content Object Reference Model)-compliant LOs between heterogeneous underlying repository infrastructures [Vassiliadis et al. 2003].

From the viewpoint of instructors, repositories can be regarded as knowledge bases in which useful LOs can be obtained. Instructors can reassemble LOs in specific sequences for different objectives; and meanwhile, the original organization of LOs may be changed. On the other hand, the creation of a new lecture indicates a series of new connections. In an earlier work [Lin et al. 2009], a concept named Reusability Tree was proposed to track the organization alteration of LOs in a CORDRA-based repository (i.e., the MINE registry). This work not only recorded the changes but proposed quantitative metrics (i.e., similarity and diversity functions) to represent the difference degree between the newly created LO organizations and the original ones. As for the LO discovery, several approaches, such as ontological [Gasevic et al. 2007] and semantic [Tsai et al. 2006] representations, were adopted for facilitating the search process. Some metrics, such as ranking [Ochoa and Duval 2008] and inference rules [Khierbek et al. 2008] were also applied to develop customized search services. However, as a summary, there are two major challenges that we may face.

(1) *Time-Consuming Process of Lecture Generation*. A huge numbers of queries have to be sent by instructors to obtain useful LOs. Although a built-in search service can serve the initial situation, the cost of time in selecting related LOs is still necessary.
(2) *Uncertain Reuse Scenario of LO*. Though the alteration histories can be tracked and quantified, the use of this information has yet been addressed. In other words, past experiences, especially in generating lectures, cannot be referred or applied to the current scenario.

As a preliminary solution to the issues stated before, this article, based on a systematic reexamination of reuse scenarios, goes further to investigate the relations between LOs from a perspective of social networks. The relations are considered as external connections (e.g., in an Activity Tree), or internal attributes (e.g., prerequisite, inheritance, and reference) between LOs. The information is utilized to construct a social structure (i.e., LOs as nodes and relations as links) in our repository. A set of metrics inspired by PageRank with time-series factor [Bringmann et al. 2010] and Hebbian rule [Stent 1973] are proposed to calculate the interdependency and to mine the implicit information within such a structure. In addition, an interactive search method, revised from Swarm Intelligence [Chen et al. 2007] and Relevance Feedback [Rocchio 1971], is proposed to assist instructors in obtaining not only a specific LO but a path, organized by past usage experiences, related to it. For instance, a possible path "Introduction - Sorting - Hashing" may be returned if a result, labeling with "Introduction," is selected where the initial query goes "Data Structure + Introduction". In short, the aim of this work is to enhance the past search scenario for LOs, and to achieve optimization of its use on the basis of its past usage experiences.

1.1. Motivation and Contribution

Although SCORM and CORDRA have provided preliminary solutions for searching and reusing LOs, a few fundamental issues are still left unsolved. For instance, an LO must have certain relations with other LOs. That is, LOs do not exist independently; so does the LO retrieval process. Recalling the search service provided by repositories, it makes use of metrics, ranking, or recommendation to assist users in obtaining desired LOs. However, the obtained LOs were isolated. That is, one query was made only for one best-fit LO (and other independent LOs with lower priorities). If users would like to search for other related LOs, they have to start a new query and to filter out the irrelevant results. This may increase the burden for instructors when reusing LOs, and thus discourage LO sharing.

To cope with this situation, we have two major contributions. First, the concept of social network is adopted to enhance reusability and searchability in the MINE registry. The LONET (Learning Object Network), as an extension of Reusability Tree, is constructed to clarify the vague reuse scenario in the past, and to summarize collaborative intelligence through past interactive usage experiences. Once LOs are revised and reused for other specific objectives, relations among them can be graphed and quantified through their newly added interdependency. Second, as a practical contribution, an interactive search method is proposed to gather possible organization paths for lecture generation in accordance with the past usage experience in the LONET. The obtained lectures are based on an input query and can be revised upon instructors' immediate feedback. If there are newly added connections, they will be added to the LONET for further usage.

The organization of this article is given as follows. Related issues and works will be addressed in Section 2. The metrics to be used for quantification of implicit information among LOs will be discussed in Section 3. The interactive search method for mining possible patterns will be described in Section 4. The evaluation and experiment on this work will be given in Section 5. We will conclude this work and address possible future work in Section 6. And finally, the implementation results will be demonst rated in the Appendix.

2. RELATED WORK

Three major issues related to the research will be discussed in this section. First, the reuse scenario of LOs is introduced. Then, the strategies for mining social networks will be addressed. And, at last, the methods for information retrieval issues within social networks will be described.

2.1. Learning Object Reusability

2.1.1. The Reusability Tree. Each Learning Object (LO) is made for a specific instructional purpose. For example, an instructor can make an LO called "Hashing" as a supplemental reference for an existing LO which introduces “Data Structure.” The correlation can be established as soon as “Hashing” is produced. Similarly, links are constructed for all LOs in a repository. The history, such as creation and updating, can be recorded in the learning object repository. On the other hand, a content aggregation identifies a hierarchical organization of LOs; and each LO represents an instructionally relevant unit for learning. A content aggregation may consist of a launchable (i.e., initially executable) LO and several other LOs. As an extension, the concept of Reusability Tree [Lin et al. 2009] was proposed. It is conceptually similar to a version derivation tree applied to a content aggregation. It consists of nodes and links, where nodes stand for associated LOs and links represent the sequence of these LOs. A child LO thus contains properties inherited from its parent LO, plus its own additional properties. When an LO is being reused, several types of changes may be made. And the changes are captured in the Reusability Tree. The metrics, such as similarity and diversity, defined in the Reusability Tree, can represent the relative information (i.e., degree of changes) in the organization of LOs.

2.1.2. Methods for LO Retrieval and Recommendation. To facilitate LO searching, usage of techniques in Information Retrieval (IR) is necessary. IR is usually applied to specific domains for being a filter to achieve retrieval and recommendation. A method named Contextual Attention Metadata [Ochoa and Duval 2008] was proposed to rank LOs by obtaining related information, such as course context and relations between learners and authors, during the LOs’ lifecycle. As an extension, time-series methods are also applied as a factor for identifying the importance of LOs in different timescales [Yen et al. 2010]. For the recommendation aspect, IR can mainly be categorized into three types: (1) user-based filtering, which classifies users based on the similarity discriminating factors including users’ interests or hobbies, to generate the user model [Herlocker et al. 2002]. Moreover, the system can provide LOs adaptive to specific user groups; (2) item-based filtering, wherein with an increasing number of users, the user-based filtering method exposes the problem of excessive computation time. Sarwar et al. [2001] proposed a novel concept wherein the items that catch users’ interests must have certain similarity with the ones that users are interested in. It can provide more stable recommendations based on the similarity between items. However, the recommendation accuracy may be low because it does not take consideration of the difference of users’ information [Balabanovic and Shoham 1997]; (3) model-based filtering, which is a type of recommendation that focuses on providing solutions to the limitations, such as information scarcity and scalability, that may cause the provision of inaccurate results. In brief, this method adopts the advantage from the user profile, the same as the one in user-based filtering, and trains the following usage experiences to make prediction. To take a search service in the MINE registry, we generate a profile for each user and obtain feedback, like ratings, to specific items in the search result list. And then the system can try to remove and/or rerank some results [Yen et al. 2011] that users may not be interested in, to save time in selecting possible LOs.

2.2. Social Network Analysis

SNA (Social Network Analysis) emphasizes on mining the exchange of resources among participants ranging from individuals to organizations [Haythornthwalte 1996]. Briefly speaking, the architecture of social networks can be divided into two types, homogenous and heterogeneous, in accordance with their internal attributes. Graph Theory [Wellman and Berkowitz 1988] is often used to make a clear description of connections between participants within a specific network. The original resources can be traced back to the studies on human relationships [Wegener 1991] around twenty years ago. And nowadays, the composition of networks, not limited to analysis of human beings, can be constructed by integrating customized factors into original mining methodologies [Bates and Peacock 1989]. To obtain expected, often implicit, information from such networks, it is necessary to take some factors defined in graph theory into consideration by using the concept of data mining for implicit information and/or patterns discovery [Jensen and Neville 2002]. The methods for determining the significance of nodes and links are PageRank [Langville and Meyer 2006], HITS (Hyperlink-Induced Topic Search) [Kleinberg 1999], and EigenRumor [Fujimura et al. 2005] whose return values can be regarded as weights of measuring the degree of centrality [Freeman 1979]. The semantic methods [Weng et al. 2009; Wolf et al. 2009; Lin and Kao 2010] are also widely used for describing interdependency within the social structure. The semantics of a node is modeled using its surrounding labeled network structure, represented by the sequences of labels (i.e., paths) together with some statistical dependency measures associated with them. A scalable mining method [Albert and Barabashi 1999] is also applied to assist researchers in discovering and generating the possible connections. As a practical contribution, the work in Yang et al. [2010] addresses the importance of prioritization to specific interactivity such as exchanging email messages. The method to SNA, such as degree of centrality, training methodology, and SVM (Support Vector Machine), were adopted to cluster delivered email messages to specific groups, and to determine the order of email receipt in accordance with an assigned priority weight. This concept is also referred by the latest introduced service “Priority Inbox” provided by Google Gmail.

2.3. Searching in Social Networks

To make efficient use of the analysis results, in this section, we go further to discuss the methods that may be applied to discover those indirect relationships between participants within a social network. Research [Scott 1991; Wasserman and Faust 1994] indicates that the social network complies with small world theory and has a feature of heavy-tail distribution [Stanley 1967]. We put it differently to explain this phenomenon. That is, it is possible to identify the follow-up (i.e., path) when obtaining an expected node. In other words, how to navigate such a structure is an open issue. Generally speaking, the methods to search social networks can be categorized into (1) path oriented, and (2) efficiency oriented. The first one emphasizes on discovering shortest and complete paths to connect different nodes. The second one pays more attention to improving the overall search efficiency to reduce the possible time cost. For example, Kleinberg [2000, 2001] proved that a simple greedy strategy could achieve shortest path length, in $O((ln\ N)^2)$, in a scalable range (i.e., hierarchically nested groups) by converting the structure into specific dimensions of the space vector. To make improvement, several methods [Moody et al. 2005] were proposed to search the nonstructured social networks. NeuroGrid [Joseph 2002] utilized routing tables generated by past query histories to achieve intelligent route selection. The same method was applied to a study [Rowstron and Druschel 2001] which utilized tags to replace routing tables in order to improve the completeness. As another instance, SSON (Semantic Social Overlay Networks) [Lin and Chalupsky 2008; Loser et al. 2007] made use of response frequencies to obtain nodes that might respond to the requests, and utilized the LRU (Least Recently Used) to remove the redundant connections. As an extension, ESLP (Efficient Social-Like Peer-to-Peer Networks) [Lin and Chalupsky 2008] utilized query thresholds to determine the possibility of correct response to search results, and rank them for deciding how many query messages should be sent from a specific node. In addition, researchers [Ghanea-Hercock et al. 2006; Koo et al. 2006] posed the issue on neighborhood selection by assigning different weights to nodes. The Hebbian rule was applied in the weighted network to revise the swarm intelligence algorithm for optimal neighborhood selection within a social-like peer-to-peer network [Liu et al. 2009].

2.4. Section Summary

Research has pointed out the trends of social network development, which mainly emphasize how implicit information, such as neighborhood selection, associated to specific nodes is obtained within a specific network environment. This concept has also been applied in the scope of distance learning for learners to obtain appropriate models to follow. As for instructors' aspects, difficulties to lecture creation are one of the issues that discourage the popularity of distance learning. In this work, we adopt the concept of social network to construct the social structure, based on Reusability Tree of the MINE registry, of LOs to clarify usage experiences of social knowledge, and to utilize the relationship between LOs to achieve intelligent lecture generation in a systematic way.

3. THE LEARNING OBJECT NETWORK (LONET)

To generate the possible path of lectures, it is necessary to declare relations between LOs. We utilize the alteration histories and past experiences of lecture generation to construct a social structure, called LONET (Learning Object Network), in our repository. In Section 3.1, we will give a brief definition of LONET and address the possible correlations between LOs. In Sections 3.2 and 3.3, the construction of LONET and metrics for quantifying the correlations will be discussed as the factors to be used in lecture path generation. A concrete instance will be given in Section 3.4.

3.1. Basics of LONET

earning Object Network (LONET) is derived from the concept of social network. It consists of nodes and links, where nodes represent the main participants (or LOs) and the links stand for the correlations associated to two connected nodes. It is used to clarify the relationships between LOs, especially those used for assembling lectures for specific objectives, and is constructed by the possible alterations recorded by the Reusability Tree while generating lectures. The main purpose of LONET is to meet the following needs.

—*To represent relationships effectively.* The matter for composing LOs to lectures varies in accordance with different teaching objectives. That is, the relationships among LOs are thought to be intricate. Once the organizations of the lecture change, the correlations between related LOs will change (e.g., remove old or establish new LOs). The repeated alteration processes are difficult to trace if an isolated hierarchical structure is the only method used. Thus, how to effectively represent the relations among a huge number of LOs is the first priority.

—*To facilitate lecture generation.* Search engine can be regarded as a tool to assist instructors in retrieving resources from the repository. However, a lot of queries are required to obtain the desired LOs. Fortunately, the past experiences that instructors utilized to generate lectures for specific objectives can be tracked. Therefore, we can provide an interactive search method to mine possible patterns for lecture generation from such networks. The pattern is composed of several consecutive LOs, and can be revised based on instructors' feedback. This may assist instructors in lecture generation.

3.2. Construction of LONET

In this section, an example will be addressed first to illustrate the shift of structure from Reusability Tree (past experiences) to LONET (social structure). Following this example, we will go further to give a general expression to formulize the constructed LONET.

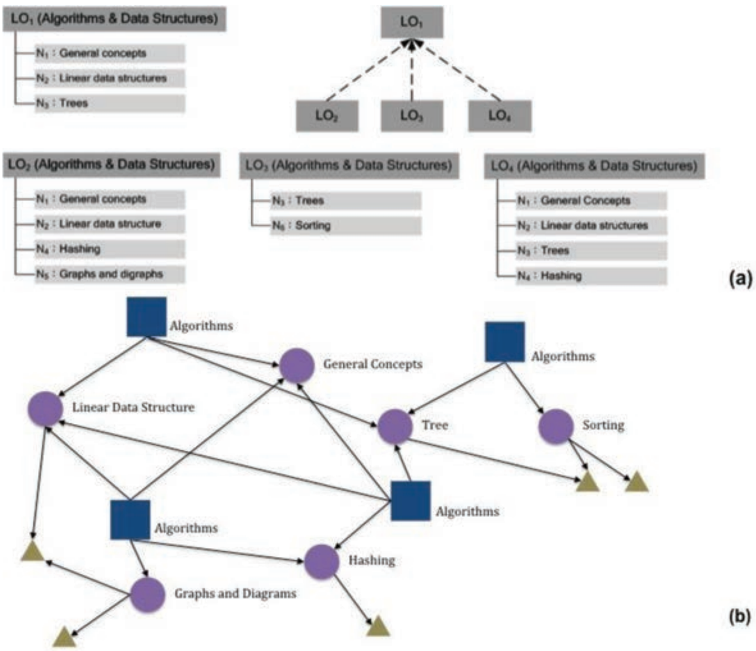


Fig. 1. Illustration of Reusability Tree and LONET (partially selected).

3.2.1. From Reusability Tree to LONET. As mentioned before, past usage experiences are necessary for constructing the LONET. A simple example from the MINE registry is shown in Figure 1(a). Once a user sends a query, the system will return results in the form of a tree-based structure that represents the relations, similarity, and diversity between LOs from a macro point of view. Hundreds of such results will be shown to users for further usage[1], but it also takes time for filtering out irrelevant ones. Considering different compositions of the activity tree in LOs, they can be taken as ways for generating specific objectives. Thus, to make an improvement, we take each node activity tree as an independent LO by giving corresponding metadata, and use a directed graph to describe the correlations as shown in Figure 1(b). In this situation, LOs not connected originally can be connected in accordance with the same objectives. This practice may strengthen the degree of binding between LOs, and may achieve a continuous process of LOs discovery that can assist instructors in lecture generation. In this figure, the elements for conversion from Reusability Tree to LONET are shown: the box is a subjective node; the circle is a common node; and the triangle is indicating a reference node, also known as the resource that can support the objectives, referred by specific LOs in LONET. The subjective node is often used to describe a specific topic, and the common node is the node that has actual learning content.

3.2.2. Definition of LONET. With the relations from the alteration history inherited from the Reusability Tree, the LONET is defined as a weighted digraph expressed by

$$G_{LONET} = (V, E, W), \tag{1}$$

where $V = \{v_1, v_2, \ldots, v_n\}$ represents the set of nodes (i.e., LOs) used for a specific lecture generation, $E = \{e_{ij}/\text{ if correlation exists between } v_i \text{ and } v_j\}$ represents the link

[1] The selected LOs may be reorganized through an authoring tool. The revision will be uploaded to the repository and added to a specific Reusability Tree according to the degree of modification.

between two nodes; and the element w_{ij} belongs to W and stands for the weight (or strength) of the links between vertex v_i and vertex v_j. Its value, ranging from -1 to 1, decides the direction between v_i and v_j. If w_{ij} is positive, the direction is from v_i to v_j; and on the contrary, is from v_j to v_i. For the case of 0, it represents no direct correlation between LOs; that is, other nodes will be needed to make the connection. The value of w_{ij} will be determined by the link(s) connecting v_i and v_j. The possible correlation (or interdependency) that forms the link(s) between nodes can be categorized into four major types shown as follows.

—*Prerequisite Correlation.* This relation is a mandatory dependency. That is, some LOs have to be utilized before accessing the LO with perquisite attributes. If not, it will be considered as an invalid access. For instance, the LO "Linked List" shalbe taught before LO "Tree" while performing a lecture "Data Structure."

—*Inheritance Correlation.* This relation is a weak dependency. That is, it is optional. It indicates a specific concept, usually a core concept that can be applied to other LOs for further usage. For instance, the LO introducing "Sorting" is related to the LO introducing "Sorting in Tree."

—*Reference Correlation.* This relation does not possess dependency, that is, it can exist independently. It is the same as an inheritance relation as an optional item. For example, it can be regarded as LOs that contain several videos or figures for describing the LO introduced as "Tree."

—*Peer Correlation.* This relation establishes through minor modification of specific LOs. It can be regarded as connections among similar nodes within the networks because of their slight differences from each other. The difference between them can be calculated through a similarity coefficient [Cost and Salzberg 1993]. For example, several LOs might have been generated by different instructors for introducing the concept "Sorting."

LONET can be constructed in accordance with these correlations between LOs. Generally speaking, nearly all of the information pertaining to correlations can be obtained from the past usage experiences of lecture generation. However, it is still insufficient to determine the "Prerequisite Correlation". For instance, there may exist lots of possibility if there is a direction from v_i to v_j. Thus, in LONET construction, the LOM-based metadata is adopted to determine the predefined correlation (i.e., prerequisite), and the Reusability Tree is used to track for others (i.e., inheritance, reference, and peer).

To quantify the weight of links, two major factors (correlation type and usage frequency) have been taken into consideration. The first is the correlation properties of LOs and the second is the actual frequency of usages to specific links. Through requirements, w_{ij} can be quantified by

$$w_{ij} = \frac{\sum \left(C\left(e_{ij}\right) \times (1 + H_{coe})\right)}{m}. \tag{2}$$

where $C(e_{ij}) = \{C_1|C_2\}$, Here

$$C_1 = \begin{cases} 1, & \text{if Prerequisite relation exists between } v_i \text{ and } v_j \\ -1, & \text{if Prerequisite relation exists between } v_j \text{ and } v_i \\ 0, & \text{otherwise} \end{cases}$$

$C_2 = F(e_{ij})$, if other correlations exist between v_i and v_j.

For correlation type $C(e_{ij})$, it can be categorized into two (i.e., C_1 and C_2) in accordance with its necessary independence. For instance, an LO with mandatory correlation indicates an irremovable connection, and must comply with the direction in accessing LOs. Therefore, the LO shall have a higher weight than others. Thus, the correlation belongs to “Prerequisite” and “No Relation” is categorized in C_1 where the value to “Prerequisite” equals 1 or -1, and 0 for “No Relation”. For the rest (i.e., inheritance, reference, and peer), variable $F(e_{ij})$ is used to assign weights. Its range will be located between -1 to 1 but not equal both of them. In the past experiences to lecture generation, lots of correlations can be used to connect the LOs. For instance, the LO “Sorting in Tree” is inherited from LO “Sorting”; and the LO “Sorting in Tree” can also be a reference to introduce its parent LO. In other words, several links can exist between nodes in LONET. The variable m indicates the number of links between connected nodes. After confirming the types of correlation, the actual frequency of usage will be addressed.

The frequency of usage in LONET shall be interpreted as the patterns, from v_i to v_j, adopted to generate lectures. Considering the revised Hebbian rule [Stent 1973], the weight of links shall be raised if they are triggered in accordance with existing patterns. On the other hand, it shall be reduced if the pattern is changed (i.e., only one of them is triggered). This approach is applied dynamically to adjust the weight of the link and to highlight the patterns often used. Thus, in Eq. (2), the additional weight, considering frequency of usage, is added to w_{ij} by H_{coe} which can be expressed as

$$H_{coe} = \varphi \sum \overrightarrow{e_{ij}}\overrightarrow{e_{ij}}^{T}, \tag{3}$$

where $\varphi = \{z/-1 \leq z \leq 1, z \neq 0\}$.

The link between v_i and v_j will be converted to vector $\overrightarrow{e_{ij}}$, and its length will increase with the frequency of usage. In order to avoid an extreme value, the vector $\overrightarrow{e_{ij}}$ will be projected onto a virtual coordinate space to obtain an additional increment or reduction to w_{ij} under normalization. That is, the obtained value will be relative to every other node connected to the datum node. An equilibrium coefficient φ is set in accordance with the directional relation (or usage direction) between nodes. Its value will be within -1 and 1, but will not be equal to 0 since the existing pattern identifies that there is at least one link between nodes. In LONET, the value of H_{coe} can also be regarded as the credibility of specific patterns.

3.3. Metrics of LONET Analysis

The fundamental issue of a social network is how nodes within a specific network can be described and quantified, to produce the corresponding positional importance. In this section, the metrics to mine the implicit information, as factors to achieve lecture generation in LONET, are addressed. The role that a node takes can be quantified through several indexes described in this section.

3.3.1. Degree Centrality. The *degree centrality* identifies the links, incoming and outgoing, which a specific node v_i may have within LONET. It can be quantified through Eq. (4). Its value reflects the importance of nodes within LONET. That is, it explains why a node is needed by other nodes or needs other nodes to support its existence. From the perspective of lecture generation, the path can be produced through the connection of links which identify the possible sequence of construction of specific objectives associated to nodes. Here

$$DegCent\,(v_i) = \frac{InDeg\,(v_i) + OutDeg\,(v_i)}{2}, \tag{4}$$

where $InDeg(v_i) \neq 0$ or $OutDeg(v_i) \neq 0$.

For a single node v_i, its degree centrality will be the average of every incoming link ($InDeg(v_i)$) and outgoing link ($OutDeg(v_i)$). Note that the degree centrality of a specific node v_i will not be 0 since every node was generated for specific objective(s), so that it must have at least one incoming or outgoing link. The metrics for obtaining the degree of incoming and outgoing links will be discussed as follows.

In LONET, the number of incoming links indicates the popularity (or degree of attention) a node has. From the perspective of lecture generation, the node with many incoming links can be regarded as a component often used for specific objectives. The PageRank algorithm is adopted as the basis for quantifying the incoming links. In addition, the time-series issue [Wassermann and Faust 1994] is also a factor that highlights the importance during different timescales. The formula can be simplified as in Eq. (5). We have

$$InDeg(v_i) = \delta \frac{\sum T\left(v_{ji}\right) R_{in}\left(T_S, v_{ji}\right)}{R_{in}\left(v_{ji}\right)}, \tag{5}$$

where $R_{in}(Ts, v_{ji})$ represents the incoming links to node v_i in a selected timescale (Ts). The least measurement unit for Ts is assumed to be one day, but can be adjusted by an administrator or instructor. The variable δ (default value is 1) is used to balance the final outcome value if the extreme value happens. The variable $T(v_{ji})$ indicates the additional weight that shall be assigned to incoming links to node v_i. We revised the Tilt-Time Window Model [Chen et al. 2002] and Time-Fading Model [Chang and Lee 2003] to calculate the weight for a specific timescale by Eq. (6).

$$T\left(v_{ij}\right)_l = \frac{D_{n-l+1}}{\sum D_l} \tag{6}$$

The $T(v_{ij})_l$ indicates the weight of every other node v_j to node v_i in a specific timescale D_l. And n is the number of units with which we separate the total length of time for obtaining different weights of links from v_j to v_i. For instance, the weight of v_{12} in D_1 will be $T(v_{12})_1$, and the weight of v_{12} in D_2 will be $T(v_{12})_2$.

Similarly, for nodes used to generate lectures, they are bound to have outgoing links in accordance with the definition in the IMS Simple Sequencing [Yang et al. 2004]. The exception can only happen in the starting node, which may have lots of outgoing links and no incoming link. For the end nodes, the types of missing links are opposite. The outgoing link, from v_i to v_j, can be calculated through Eq. (7).

$$OutDeg\left(v_i\right) = \delta \frac{\sum T\left(v_{ij}\right) R_{out}\left(Ts, v_{ij}\right)}{R_{out}\left(v_{ij}\right)} \tag{7}$$

The major difference between Eqs. (6) and (7) is the direction of links, and the weighted values for the links in different timescales.

3.3.2. Eigenvalue. *Eigenvalue* is used to find out the contributing source within LONET by assigning relative values to each network participant. It is originated from the eigenvector. From another viewpoint, a node with high eigenvalue must have lots of incoming links or has been linked by nodes with higher InDegree (or authority). This value can be used to rank the nodes within LONET by

$$EiVal\left(v_i\right) = InDeg\left(v_i\right) + \frac{\left|w_{ij}\right| \cdot InDeg\left(v_j\right)}{\sum \left|w_{ij}\right|}. \tag{8}$$

In Eq. (8), we only consider the direct interaction between two connected nodes when calculating the eigenvalue. That is, there is no existing intermediate node. With this definition, the eigenvalue of node v_i will be the sum of part of the InDegree value, which depends on the percentage of w_{ij} in $\sum |\mathrm{w}_{ij}/$, from every connected node v_j. In this study, the eigenvalue can help to find the entry node when receiving the query, since a node with a high eigenvalue is considered to have more links and can be the basis for deciding a search direction.

3.3.3. Betweenness Centrality. In general, there shall be one shortest path for connecting two nodes within LONET. The *betweenness centrality* is used to quantify how many times a specific node appears in the shortest paths connecting all other nodes. A node with high betweenness centrality could be an important node within the network, and can be regarded as a bridge that connects two clusters that have no direct link. If this node is removed from the network, it may cause fracture. The value can be obtained by

$$BetCent\,(v_i) = \frac{\delta}{|R|^2} \sum\nolimits_{j \neq i}^{|R|} \sum\nolimits_{l \neq i, l \neq i}^{|R|} \frac{\sigma_{li}\,(v_i)}{\sigma_{li}}, \tag{9}$$

where σ_{lj} indicates the sum of all pairs of shortest distances going from v_l to v_j, and $\sigma_{lj}(v_i)$ represents the sum of shortest distances from v_l to v_j via v_i. And $|R|$, which is equal to $|R_{in} + R_{out}|$, indicates the sum of links that v_i has. To obtain the average path lengths, we utilize $|R|^2$ as a denominator for normalization. The size of $BetCent(v_i)$ will affect the composition of lecture path generation. This situation often occurs in arrangement of nodes used to describe the same subject. For instance, it is how the sequence between "Linked List" and "Queue" can be arranged when introducing the "Data Structure."

3.3.4. Closeness Centrality. The nodes are connected through links that can also be quantified to indicate the distance between nodes. The *closeness centrality* indicates a node that has the shortest path to access every other node within the network. In LONET, only the nodes in a specific area, close to a target cluster, will be calculated to obtain their closeness centrality. It will lead to a relative value for selecting alternative nodes when instructors make changes to the original lecture generation. The closeness centrality of a node can be obtained through

$$CloseCent(v_i) = \sum\nolimits_{i \neq j} \left[d_{ij}\right]^{-1}, \tag{10}$$

where d_{ij} represents the distance between v_i and v_j, and can be formed by

$$d_{ij} = \sum \left|\vec{v_i} - \vec{v_j}\right|. \tag{11}$$

The closeness centrality of a node is defined by the inverse of the average length of the shortest paths to/from all the other nodes in LONET. The value will be taken as the first visit neighborhood to reproduce the following path if the changes have been made. For instance, an instructor would like to change the original sequence, from "Linked List" to "Queue," when generating a lecture introducing "Data Structure." In this situation, every node connected by "Queue" shall be rearranged (or the path of "Linked List" will be removed). The detailed algorithm to achieve this goal will be discussed in Section 4.3.

3.4. An Example

We utilize five existing nodes, $v_1(LO_1)$, $v_2(LO_2)$, $v_3(LO_3)$, $v_5(LO_5)$, and $v_6(LO_6)$, which form a small group in LONET as an example to describe how the proposed metrics work. The relations between them are shown in Figure 2. We take v_1 as the basis. The incoming link of v_1, $R_{in}(v_{31})$, represents the partial link from v_3 to v_1 while the outgoing link of v_1, $R_{out}(v_{12})$, is the link from v_1 to v_2.

First, we assume that the age, the existing timescale, of v_1 is three months, 30 days per month, and 12.5 days, say 102.5 days in total. The incoming/outgoing links of v_1 are 55/30 times in the first three months, 25/20 times in following twelve days, and 20/15 times in the last half-day. And the sum of outgoing links from v_3 to v_2 is 130. In this situation, we can separate the timescale into three parts

$$D_1 = 0.5, D_2 = 12, D_3 = 90, n = 3$$

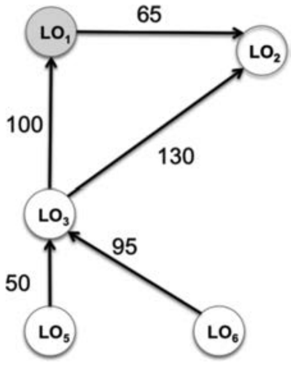


Fig. 2. Illustration of existing nodes.

and the weight of each incoming/outgoing timescale can be obtained through Eq. (6).

$$T\ (v_{31})_1 = T\ (v_{12})_1 = \frac{D_{n-l+1}}{\sum D_l} = \frac{D_{3-1+1}}{D_1 + D_2 + D_3} = \frac{90}{102.5} = 0.878$$

The weights of the rest of the timescales, 0.117 and 0.004 for the second and the third timescale, respectively, can be obtained through the same equation. Thus, we can calculate InDegree and OutDegree of v_1. The detailed process, with default value of variable δ, is given as follows.

$$\mathit{InDeg}\,(v_1) = \delta \frac{\sum T\ (v_{31})\, R_{in}\,(Ts,\, v_{31})}{R_{in}\,(v_{31})} = \frac{(0.878 \cdot 20 + 0.117 \cdot 25 + 0.004 \cdot 55)}{20 + 25 + 55} = 0.207$$

$$\mathit{OutDeg}\,(v_1) = \delta \frac{\sum T\ (v_{12})\, R_{out}\,(Ts,\, v_{12})}{R_{out}\,(v_{12})} = \frac{(0.878 \cdot 15 + 0.117 \cdot 20 + 0.004 \cdot 30)}{15 + 20 + 30} = 0.240$$

$$\mathit{DegCent}\,(v_1) = \frac{0.207 + 0.240}{2} = 0.224$$

After obtaining the degree centrality, including InDegree and OutDegree, we go further to calculate the eigenvalue of v_1. The concept of eigenvalue is similar to the ranking algorithm used in search engines. The authority of a source node will affect the value. Thus, to start to calculate the eigenvalue of v_1, the InDegree of v_3 is necessary. We assume the nodes, v_3, v_5, and v_6 form another group as well in LONET, and the relations between them are shown in Figure 2.

We do the same calculation to obtain the InDegree of v_3. The age of v_5 is 1 month and 5 days while the age of v_6 is 15.5 days. The incoming links from v_5 to v_3 are 40 in the first month and 10 in the following 5 days, and the incoming links from v_6 to v_3 are 60 in the first half-month (15 days) and 35 in the following 0.5 days. Through the definition, we can calculate the weight of each timescale through Eq. (6) and obtain that $T(v_{53})_1$ is 0.857, $T(v_{53})_2$ is 0.143, and $T(v_{63})_1$ is 0.968, $T(v_{63})_2$ is 0.033. Then we utilize the Eq. (5) again to obtain the InDegree of v_3 as follows.

$$\begin{aligned} \mathit{InDeg}\,(v_1) &= \frac{\sum T\ (v_{53})\, R_{in}\,(Ts,\, v_{53})}{R_{in}\,(v_{53})} + \frac{\sum T\ (v_{63})\, R_{in}\,(Ts,\, v_{63})}{R_{in}\,(v_{63})} \\ &= \frac{(0.857 \cdot 10 + 0.143 \cdot 40)}{40 + 10} + \frac{(0.968 \cdot 35 + 0.033 \cdot 60)}{35 + 60} = 0.663 \end{aligned}$$

Table I. Corresponding Metadata Elements of Nodes v_1 and v_2

Metadata of v_1 (Partially Selected)	Metadata of v_2 (Partially Selected)
... <general> <title>Algorithm & Data Structures</title> <language>eng</language> <keyword>data structure, intro., overview</keyword> <coverage>Higher education</coverage> </general> ... <technical> <format>text/.html</format> <size>29234</size> <location> 2097/e4a921.html</location> </technical> <educational> <learningresType>lecture</learningresType> <IntendRole >University</IntendRole > <typageRange>19-22</typageRange> <difficulty>medium</difficulty> <typTime>3 hour/hours</typTime> </educational> ... <annotation> <date> <datetime>2008/10/1</datetime> </date> <description> <string language = “eng”> web content resource </string> </description> </annotation>	... <general> <title>Algorithm & Data Structures</title> <language>eng, jpn, chi</language> <keyword> data structure, intro., hashing, overview</keyword> <coverage>Higher education</coverage> </general> ... <technical> <format>text/.html</format> <size>30173</size> <location> 2097/ef349.html</location> </technical> <educational> <learningresType>lecture</learningresType> <typageRange>22-25</typageRange> <difficulty>medium</difficulty> <typTime>2.5 hour/hours</typTime> </educational> ... <annotation> <date> <datetime>2008/12/21</datetime> </date> <description> <string language = “eng”>This is the web content resource from MINE Registry</string> </description> </annotation> ...

The value of InDegree of v_3 will be divided into two according to the portion of its outgoing links. Thus, through Eq. (8), we can obtain the one that will be added to v_1:

$$EiVal\,(v_1) = InDeg\,(v_1) + \frac{|w_{13}| \cdot InDeg\,(v_3)}{|w_{13}| + |w_{23}|} = 0.207 + \frac{100 \cdot 0.663}{100 + 130} = 0.495$$

Then, we consider the betweenness centrality and closeness centrality of v_1. In accordance with the definition of Eq. (9), there are two kinds of links that connect v_2 and v_3. One goes directly between v_2 and v_3, while another one goes via v_1. In this situation, we can obtain $\sigma_{23}(v_1)$ equals to 165, the sum of $R_{out}(v_{12})$ and $R_{in}(v_{31})$, and σ_{23} equals to 130, the sum of $R_{out}(v_{32})$. Then, the value of betweenness centrality of v_1 can be obtained by

$$BetCent\,(v_1) = \frac{1}{|165|^2} \cdot \frac{165}{130} = 0.007.$$

And, lastly, the closeness centrality depends on the distance between connected nodes. In distance learning, the optimal way to calculate the distance between objects is to adopt the corresponding metadata elements. We assume the metadata descriptions of v_1 and v_2 are as shown in Table I.

We defined each matched term in metadata as equaling “1” [Wu et al. 2008]. Thus, the similarity within category “General” between v_1 and v_2 will be

$$d_{12\_General} = 1 + \frac{1}{3} + \frac{3}{4} + \frac{1}{4} = 0.771.$$

With the same process, we can also obtain the similarity in the of the rest categories of metadata. They will be “0”, “0”, “1”, “0.567”, “0”, “0”, “0.556”, and “0”, respectively. Thus the distance between v_1 and v_2 shall be

$$d_{12} = 0.771 + 0 + 0 + 1 + 0.567 + 0 + 0 + 0.556 + 0 = 2.894$$

and the closeness centrality of v_1 will be

$$CloseCent\,(v_1) = {d_{12}}^{-1} = (2.894)^{-1} = 0.346.$$

After the calculation, the results of attributes of each participant node within LONET can be applied to the search process addressed in the following section.

4. INTERACTIVE SEARCH IN LONET

In this section, issues related to lecture generation in accordance with LONET and attributes of its participating nodes will be discussed. Three parts are involved in this process. First, the search criteria that assist instructors in obtaining possible LOs will be defined. The possible lecture paths generated by analyzing the input query will then be addressed. And finally, the algorithm that will reorganize the path by dynamic changes from instructors will be described.

4.1. Definition of Search Criteria

The elements of the IEEE LOM have been used to assist users in obtaining LOs in our repository. Based on the elements of the IEEE LOM, the search criteria can be classified into the following groups: precise criteria, incremental criteria, precedence criteria, time/duration criteria, and single/multiple-choice criteria, which have already been addressed in Lin et al. [2009]. In these classifications, five selected elements [Rivera et al. 2004] have been adopted to strengthen the educational usage. There are two items, keyword and coverage, under the general category, and three others, “LearningResourceType,” “Difficulty,” and “TypicalLearningTime,” under the educational category.

4.2. Lecture Path Generation

A lecture path can be considered as the possible recomposition of LOs used to generate lectures for specific objectives. This concept is used to reduce time and effort that instructors may spend in generating specific lectures. In our past experiences of searching for LOs, search results consists of lots of isolated LOs. In this article, in addition to providing LOs and their related recommendations, structures (or templates) with specific sequences are also generated. Instructors can choose to follow or make further revision of the structures. The revision will be taken as a new composition of a lecture. This may simplify the tedious lecture creation process encountered in the past through collaborative intelligence from users.

The lecture path is produced according to possible sequences within LONET. To achieve this, the first step is to analyze the queries from instructors, and to produce the entry nodes. The entry nodes here can be considered as a list of LOs returned by searching services from repository. Each of the nodes is considered as a center in a lecture web, and can be used to determine the direction to lecture path generation in accordance with its implicit properties. This iterative retrieval process will be kept in a temporary array to achieve a continuous path. Inspired by the swarm intelligence

Table II. Algorithm for Step 1

```
Input: users' query $\overrightarrow{Q}$ with search criteria $Ci$
Output: corresponding results array, $V_{ent}[v_m]$, ranked by similarity
1: initialize query vector $\overrightarrow{Q}$
2: let $L_q := $ IEEE LOM elements, where $\overrightarrow{Q} \subseteq L_q$
3: for each input criteria $Ci$ in $\overrightarrow{Q}$
  3.1 initialize query attribute array $Aq[r_w][s_x]$ for $Ci$, and node attribute array
      $Av[r_y][s_z]$ for $v_i$ in $V$, where $r_w, r_y \in${LOM category}, and
      $s_x, s_z \in${sub elements in corresponding category},
  3.2 calculate similarity degree between $Aq$ and $Av$
      for each $i$ with $i < |V|$
        $Sim_{coe}(Aq, Av_i) = \frac{\overrightarrow{Aq}\,\overrightarrow{Av_i}}{\left|\overrightarrow{Aq}\right|\left|\overrightarrow{Av_i}\right|}$
  3.3 initialize entry nodes array, $V_{ent}[v_m]$, where $V_{ent} \in V$
      add top-N nodes with high $Sim_{coe}$ to $V_{ent}[v_m]$
4 : return $V_{ent}[v_m]$
```

Table III. Algorithm for Step 2

```
Input: ranked results array $V_{ent}[v_m]$
Output: entry array of candidate nodes $V_{ent}'[v_m']$
1: reset $V_{ent}$ by neighbor node of $v_m$ with higher OutDegree
  1.1 for each node $v_{(m,j)}$ in $V_{ent}$, find nearest node $v_n$ with highest OutDegree
      for $l$-th neighbor nodes $v^l_{(m,j)}$, $l = \{0,1,\ldots,n\}$ of $v_{(m,j)}$
        if ($InDeg(v^l_{(m,j)}) < InDeg\ (v_m)$ AND
          $OutDeg(v^l_{(m,j)}) > OutDeg(v_m)$)
            $v_m \leftarrow v_m' = v^l_{(m,j)}$
        else continue
  1.2: reorder the $V_{ent}'[v_m']$ based on OutDeg($v_m'$)
2 : return $V_{ent}'[v_m']$
```

algorithm, which is used to discover the shortest path, we developed an algorithm to obtain the nearest-neighbor nodes when determining the entry nodes.

As for lecture generation, the shortest path is not the primary means to assemble the lecture, but rather the frequency of usage from users. The higher frequency of usage represents higher credibility, and can also be served as the countable reference for lecture path generation. Thus, common user experiences are used. The generation of a lecture path can be categorized into the following steps.

—*Step* 1. In the beginning, the attribute array $Aq[r_w][s_x]$ will be generated by converting the query from the user in accordance with its corresponding metadata description. Then, to return the top-N items, $V_{ent}[v_m]$, the matched objects in LONET will be ranked, from high to low, on the basis of their relative similarity. It could refer to the algorithm shown in Table II.

—*Step* 2. The swarm algorithm will be revised to be the basis for visiting every node that has connections with node v_m in V_{ent}. The nodes with a shorter path, with less nodes between v_m and $v_{(m,j)}$, and a higher OutDegree value, will be the priority obtained. The main reason is that the node with a higher OutDegree identifies that more nodes can be accessed. In other words, the node offers more choices for the system to select. Thus, we set up a rule that if the visited node $v_{(m,j)}$ has a higher OutDegree value, a shorter distance than v_m will replace the original one. And a new entry node array $V_{ent}'[v_m']$ will be generated. The process is shown in Table III.

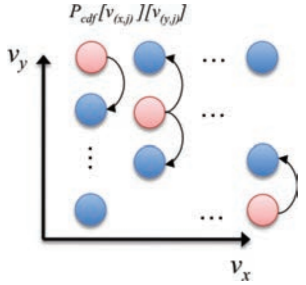


Fig. 3. Illustration of lecture path generation.

Table IV. Algorithm for Step 3

```
Input: revised candidate node array $V_{ent}'[v_m']$
Output: initialized vector space for lecture generation
1: initialize path array $P_{cdt}[v_x][v_y]$ where $v_x \in V_{ent}$, $v_y \in V$ and has direct edge $e_{xy}$
    convert $v_m'$ in $V_{ent}$ to $v_x$ in $P_{cdt}$
      for each $j$ with $j < |v_m'|$
        $v_{(x,j)} \leftarrow v_{(m,j)}'$
        $v_{(y,j)} \leftarrow null$
```

—*Step* 3. The returned entry array, $V_{ent}'[v_m']$, will be converted to a two-dimensional vector space $P_{cdt}[v_x][v_y]$, where v_x represents the entry nodes from v_m' and v_y indicates the possible path to v_x as shown in Figure 3. The circle with light color represents the possible position of an entry node where the others are default set to *null* for expansion. Three possible positions are formed in the path generation process. In the left-hand side of Figure 3, the position identifies a start node regarded as the first object of a lecture (e.g., introduction). The middle position stands for the middle node that may achieve two-way expansions or be recognized as a middle object of a lecture. The right-hand side represents the end node opposite from the start node (e.g., conclusion). The algorithm shown in Table IV was applied.

—*Step* 4. As shown in Figure 3, nodes used to generate the lecture path are supposed to have correlations with a specific sequence. But it is difficult for us to know if the entry node, the first node to generate the lecture, can be determined by a user's query. What we can do is to obtain an appropriate entry node, with a high OutDegree, in accordance with the user's query. As for the entry node, it may be located at any order in the path. Hence, the proposed metrics, the weight of the edge between two connected nodes and eigenValue, are utilized to determine the possible direction of the path for those correlations that do not belong to "Prerequisite." The possible directions can be tracked backward or forward. As stated in Section 3.2.2, the correlations can be categorized into four types, and there is only one mandatory interdependency (the other three are optional). That is, the correlations except "Prerequisite" will not affect the generation rule of the lecture path. Thus, we only consider the edge with the prerequisite correlation. If such a kind of correlation exists, it will be added to the path without calculating the eigenValue, and the following direction will be determined. The detailed algorithm is listed in Table V.

—*Step* 5. To avoid duplications (i.e., two same nodes appear in a same path), the mechanism named "Duplicated Nodes Removal" has been utilized to make cross-comparison to all the nodes within $P_{cdt}[v_x][v_y]$. If duplication exists, the node being added later will be removed directly, and the process will go back to the previous

Table V. Algorithm for Step 4

```
Input: revised candidate node array $V_{ent}$'[$v_m$']
Output: filled path array $P_{cdt}[v_x][v_y]$
1 : for each $v_x$
        let its tolerable threshold $\kappa$ for candidate neighbor
        nodes selection with direction $dir$
2 : for $\forall v_i \in V$ that has not been visited
        for $k$-th neighbor nodes $v^k_{(x,j)}$ $k = \{0,1,\ldots,n\}$
          if $v^k_{(x,j)} \notin P_{cdt}$
            calculate highest weight $|w_{(x,j)i}|$ from $v_{(x,j)}$ to $v_i$
            if $e_{(x,j)i}$ contains no Prerequisite Information
              if $EiVal(v_i) \geq \kappa$
                add $v_i$ into $P_{cdt}$
                if $w_{(x,j)i} > 0$
                  $y = y - 1$
                  let $v_{(y,j)} \leftarrow v_i$
                  $dir = backward$
                else
                  $y = y + 1$
                  let $v_{(y,j)} \leftarrow v_i$
                  $dir = forward$
              else continue
            else if
              add $v_i$ connected by $e_{(x,j)i}$
              $y = y - 1$
              $v_{(y,j)} \leftarrow v_i$
              go to 2
3 : return $P_{cdt}[v_x][v_y]$
```

Table VI. Algorithm for Step 5

```
Input: $P_{cdt}[v_x][v_y]$
Output: $P_{cdt}$'$[v_x][v_y]$
1: check for duplicated nodes $v_d$ in same column of $P_{cdt}$
    for each j with j < $|v_j|$
      compare $v_{(y,j)}$ with rest nodes in $v_{(y,j+1)}$
      if ($v_{(y,j+1)}$ is duplicated from $v_{(y,j)}$)
        then remove $v_{(y,j+1)}$
        go to Step 4 (2)
      else continue
2 : return $P_{cdt}$'$[v_x][v_y]$
```

one to look for alternative nodes. If there is no existence of a duplicated node, the produced candidate array $P_{cdt}{}'[v_x][v_y]$ will be converted to a list of single paths that will be the result responding to a user's query. Then decisions can be made by users to make further modifications. The detailed algorithm is shown in Table VI.

4.3. Path Reorganization

Once a user has received possible lecture paths from our repository, he/she can make further modifications. That is, the paths offered by the generation tool are regarded as templates for reference to generate specific lectures, not the final paths that must be followed. We make use of the metrics, betweenness centrality and closeness centrality,

Table VII. Algorithm for Candidate Path Reorganization

```
PathReorganization()
Input: User's Request Req()
Output: Revised Path $P_{rev}[v_x][v_y]$
1: check the status of input request
   if $Req() = $ "Add"
     do 2
   else if $Req() = $ "Removal"
     do 3
   else return $P_{rev}[v_x][v_y]$
2: check the attribute of added node $v_a$
   2.1 check $Sim_{coe}(v_a, {}_{v(y,j)})$ between $v_a$ and $v_{(y,j)}$
        find appropriate position to add node $v_a$
        $P_{rev}[v_x][v_{(y,j+1)}] \leftarrow P_{cdt}[v_x][v_{(y,j)}] + v_a$
   2.2 if $v_a \in V$
           add $v_a$ to $V$
        else continue
   2.3 add correlation $(e_{a(j+1),}\ e_{a(j-1)})$ to nodes associated to $v_a$
3: check position of removal node $v_r$ in $v_{(x,j)}$
   for each $j$ with $j < |v_j|$
     if $v_r = v_{(y,j)}$
        for nodes set $V'$ connected to $v_r$
           compute $CloseCent(v_z)$, where $v_z \in V'$
        convert to $N_c[v_z]$, where $N_c \in V'$
        for each $p$ with $p < |v_z|$
           if $e_{(y,j)(z,p)}$ exists
             $v_{(y,j)} \leftarrow v_{(z,p)}$
           else
             $v_{(y,j)} \leftarrow v_{(z,p+1)}$
     else skip
4: return $P_{rev}[v_x][v_y]$
```

to develop a relevance feedback algorithm to achieve this purpose, which is shown in Table VII.

The processes to path modification can be mainly categorized into two aspects. The first is to add nodes into a path, while the second is to remove nodes from the candidate path. In the adding process, we will first calculate the possible position of the nodes, which may be added, within the selected path through comparison of similarity between v_a and $v_{(y,j)}$. Then the node v_a will be directly added to the path, and the newly constructed correlations (i.e., $e_{a(j+1)}$, $e_{a(j-1)}$) will also be returned to LONET. Compared to the adding process, the removal process will be more complicated. When the system receives the instruction to remove a specific node v_r, it will check which node (i.e., $v_{(y,j)}$) shall be removed. Then, the system will calculate the nodes (possible alternative nodes) that have a higher closeness centrality to $v_{(y,j)}$, and place them in a temporary set V'. After that, the correlations associated with v_r will be determined. If the correlation does not belong to "Prerequisite," the corresponding node $v_{(y,j)}$ in the path will be removed directly. Otherwise, if correlation belongs to "Prerequisite," two of them will be removed together. And the alternative nodes will be selected again through the same process to reorganize the revised path. The algorithms seem complicated. However, in the following section, we will demonstrate the feasibility. And an experimental system is illustrated in the Appendix.

Table VIII. Experiment Results from TREC Evaluation

			Raw Data		Non-Interpolated Data	
LO Retrieved	Precision	Relevant Retrieved	Recall	Precision	Recall	Precision
100	1.0000	100	0.1190	1.0000	0.0	1.0000
200	0.9200	184	0.2143	0.9200	0.1	1.0000
400	0.9475	379	0.3333	0.9475	0.2	1.0000
600	0.8317	499	0.3810	0.8317	0.3	0.9200
2000	0.7675	1535	0.5476	0.7675	0.4	0.8095
4000	0.3705	1482	0.8810	0.3705	0.5	0.7857
10000	0.2001	2001	0.9286	0.2001	0.6	0.7568
20000	0.0500	1000	0.9524	0.0500	0.7	0.6977
					0.8	0.4789
					0.9	0.3585
					1.0	0.0000

5. EXPERIMENTS AND EVALUATION

5.1. The Dataset

Our repository system, the MINE registry, has stored around 22,000 LOs in the past few years. Most of the learning objects were collected from open-access training programs, such as Open University, UK and ADL (Advanced Distributed Learning), USA, and from the local universities. The LOs in the MINE registry can mainly be categorized into four major types: Photoshop, Algorithms and Data Structure, Computer Science, and others; each containing about 6000, 9000, 5000, and 2500 LOs, respectively. And these are the nodes we used to construct LONET.

5.2. Overall Performance of Search Services in MINE Registry

To achieve lecture path generation, the fundamental issue is to determine the entry nodes in accordance with queries. In this situation, the performance of the search engine, especially the retrieved items and the ranked results, becomes important. Two experiments were conducted to obtain the performance of our proposed search service. In the beginning, the TREC [Buckley and Voorhees 2004] was adopted to evaluate the LOs stored in the MINE registry without further data processing. The metrics [Jarvelin and Kekalainen 2000] for estimating the performance are shown as follows, where P′ represents the precision value and R′ indicates the recall value. The nIAP′, non-Interpolated Precision, was used to calculate the precision of a retrieved LO within a ranked result. In nIAP′, the variable r_j is the sum of queries sent to the system, the $LO_j(i)$ is the order of retrieved LO in a list, and the i is the order of query. To perform nIAP′, the cut-off, number of documents in a result list, needs to be calculated. In this research, it was set to 10 by default. The results can be found in Table VIII.

$$\text{P}' = \text{number of relevant LOs retrieved/number of LOs retrieved} \tag{12}$$

$$\text{R}' = \text{number of relevant LOs retrieved/number of relevant LOs} \tag{13}$$

$$nIAP' = \frac{\sum_{i=1}^{r_j} i/LO_j(i)}{r_j}. \tag{14}$$

In Table VIII, two kinds of data are examined. The raw data represents the results of common LOs in the MINE registry while the noninterpolated data is for the ranked results. Through the preceding results, the precision-recall curve for both raw data and non-interpolated data are shown in Figure 4.

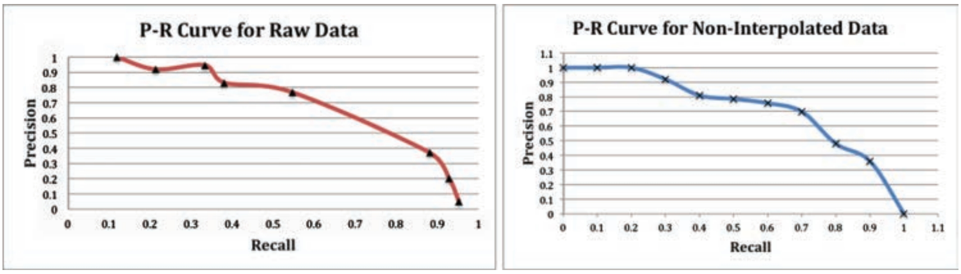


Fig. 4. P-R curve for performance of search service in MINE registry.

Table IX. Precision-Recall Evaluation for Classified LOs

	Q_1	Q_2	Q_3
Total LOs in Repository		22141	
Relevant LOs in Repository	5962	8713	4948
Retrieved LOs	6235	9214	5123
Relevant LOs Retrieved	5602	8313	4601
Precision	89.85%	90.22%	89.81%
Recall	93.96%	95.41%	92.99%

In addition, to enhance the searchability for specific categories in the MINE registry, the k-nearest method was adopted to make classification of the LOs in accordance with the corresponding metadata. As stated, three classifications, Photoshop, Algorithms and Data Structure, and Computer Science, were used in precision-recall evaluation. In this experiment, three queries, based on predefined search criteria (keyword as example), were used to perform the experiments. The first query (Q_1) is to obtain related LOs from the major type "Photoshop" with a query term "photoshop." The second query (Q_2) is for the major type "Algorithms and Data Structure" with term "algorithm + intro". The third query (Q_3) is with term "cs + agenda" to query the major type "Computer Science". The corresponding results to these queries are shown in Table IX.

According to the results, it is worth mentioning that the precision and recall value reaches, in average, 90% and 94%, respectively, which reveales that our search mechanism can assist users in obtaining related learning objects in an efficient way. And this result also shows that the entry nodes retrieved by our search service have certain creditability for being the basis of generated lecture paths.

As for lecture path review, another experiment on the accuracy evaluation of selected nodes in the generation process of the lecture path was conducted based on the same method. The selected nodes to generate the lecture path were utilized to compare with the ones we obtained in candidate array. This relation is used to generate a precision-recall curve. We involved two P-R curves in the previous experiment as the based line, which will be compared with the one generated according to the proposed interactive search mechanism. In Figure 5, the curved with circled dots on it is the result while using interactive search. The linear dashed line going across these three curves is the average trend. It is obvious that there is no prominent part in this curve through comparison with the baselines. But, by and large, the proposed search method can reach average accuracy.

5.3. Experiments on Usability of Search Scenario

Another experiment was conducted to evaluate the usability of the proposed interactive search process. In this experiment, we focus on the implicit feedback from users. We

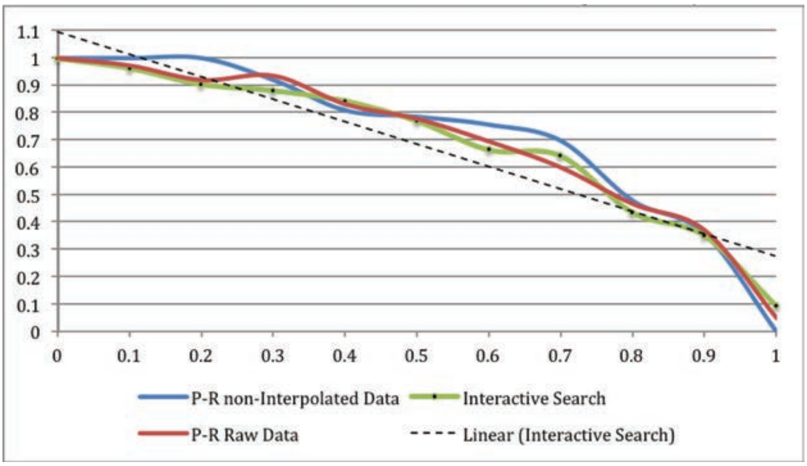


Fig. 5. Precision-recall curve of selected nodes in lecture paths.

Table X. Results for Usage Scenario in Search Service

(*Unit:Item*)

	Basic Search			Interactive Search		
No. of Terms	Visit/Page	Selection	Revision	Visit/Page	Selection	Revision
1	5.07	3.16	2.12	7.18	3.60	2.06
2	5.66	3.32	3.67	6.33	4.13	1.54
3	3.53	1.85	3.39	4.11	2.33	1.37
4	2.24	1.10	1.43	1.97	0.79	0.47
5	1.51	0.76	0.48	1.01	0.34	0.27

collect the usage data from 25 instructors that includes 10 lecturers and 15 teacher assistants. Three major factors were collected while accessing the search service. The first one is the "number of visits per page" to the item listed in the result list. In our search engine, the returned items within a result page are 10. The second one is the "number of selections" to the returned items. The third one is the "number of query revisions" that an instructor will send to obtain an expected item. The results are shown in Table X.

The results reveal that the interactive search can reduce the number of query revisions to at least 1.08, in average ((5.71 – 11.09)/5), comparing with the basic search that follows the general core of the search engine. Also, the interactive search can raise the actual use of returned items within the result list because the average selection is up to 0.2 ((11.19 – 10.19)/5) in average. Through this experiment, we can also observe that a preferable number of keywords, in this case, shall be in the range from 2 to 3.

In addition, the number of nodes in a lecture path is also recorded. The selected items in the result list are utilized to generate the initial path, and we collect the revision scenario of the generated paths. The length of a lecture path is, by default, set to be between 3 and 7. In this experiment, 200 paths, equal to 200 selected LOs, were utilized to reach the results. There are 40 in length 3, 50 in length 4, 35 in length 5, 35 in length 6, and 40 in length 7. We can observe that the preferable length of generated path is around 6, by rounding off the numbers in this experiment. The detailed results can be found in Table XI.

Table XI. Results to the Use of Lecture Path Generation

Initial Length	Final Length	Number of Revision	
		Add	Remove
3	4.47	3.13	0.19
4	5.13	2.79	0.54
5	5.46	1.68	0.77
6	5.57	0.46	1.32
7	5.96	0.13	2.21

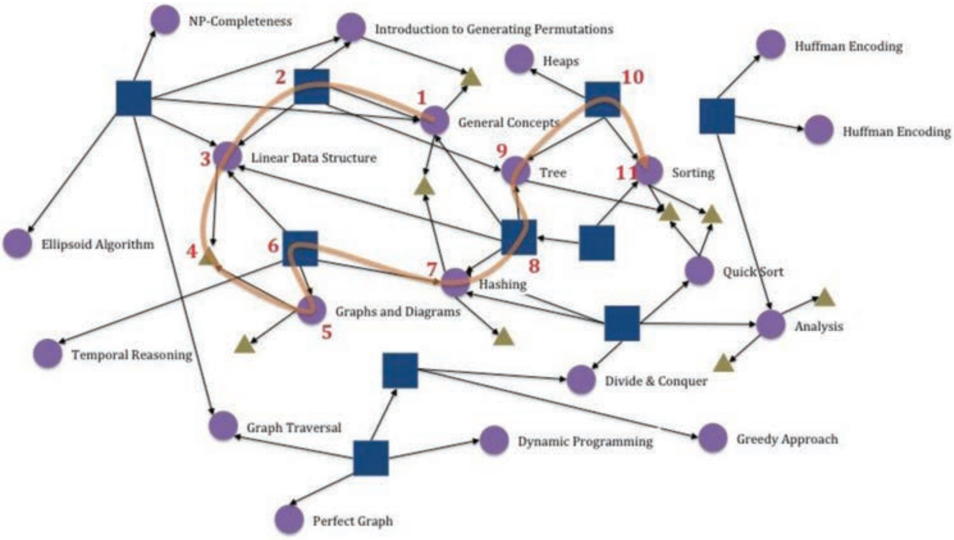


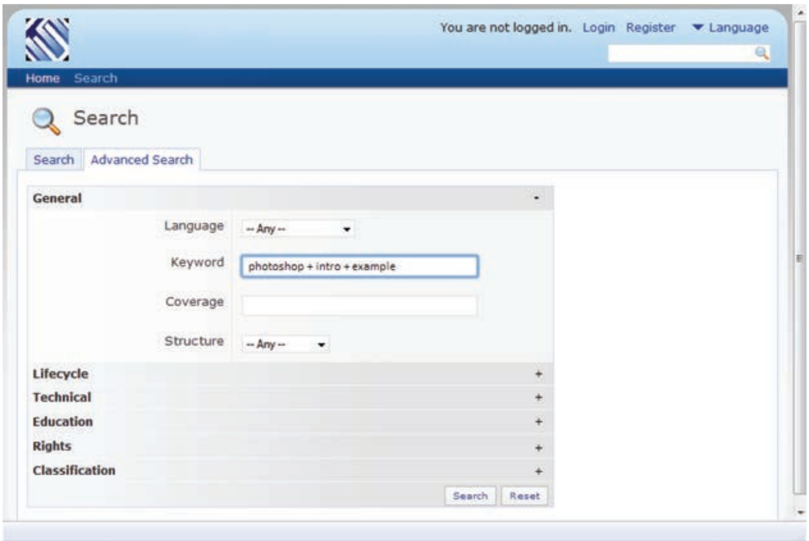
Fig. 6. An example to lecture path generation with query Q_2.

5.4. Instance of Lecture Path

Figure 6 shows a concrete example for the lecture path generation in accordance with query Q_2. In this example, we only list one part, the “Algorithms and Data Structure cluster”, of the overall LONET. With the query, the entry node “1” can be obtained through similarity calculation. Then, we look for the neighbor nodes associated to node “1.” Through this figure, it is obvious that node “2” has the highest OutDegree through the number of outgoing links to node “1.” Thus, the direction, compared to node “1”, is headed to the left. Then, we go further to check the eigenValue and the correlations between following candidate nodes in accordance with the processes addressed in Section 4.2. A preliminary path to generate a lecture corresponding to the query Q_2 is constructed through the interactive iterative search process. Through the query, the generated path has 11 nodes, and contains three different kinds of attributes. That is, there are four subjective nodes (blue color), one reference node (gray color), and six common nodes (purple color). The detailed definition has been given in Section 3.2.1. We only put emphasis on the common nodes. So the lecture path corresponding to the input query Q_2 shall be “1→3→5→7→9→11”. The generation process will be ended when the next obtained common nodes have no direct usage history.

6. CONCLUSIONS

The issues to learning object reuse have been widely discussed in the scope of distance learning literature. Although a simple approach to realize reusability relies on finding



Language -- Any --

Keyword photoshop + intro + example

Coverage

Structure -- Any --

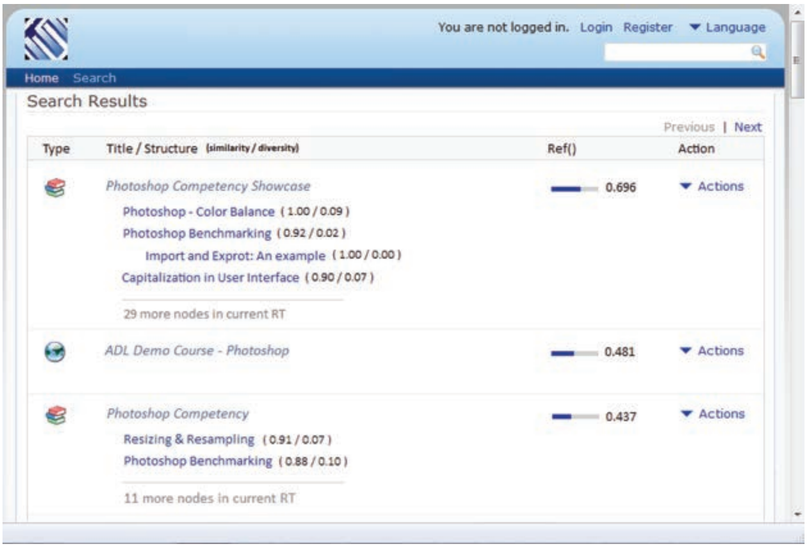
Fig. 7. Advanced search interface.

learning objects in a repository, the generation of lecture paths based on user experiences could lead to another dimension of reusability. In this article, we have presented an integrated approach to utilize user practical experiences in using a Reusability Tree, and proposed an associated searching mechanism. Based on the modification of LOs in a repository, a network structure among LOs is constructed. The structure is then used in the automatic generation of lecture paths.

The major contributions of this article can be summarized as follows. First, based on concepts from a social network, we proposed LONET (Learning Object Network). The extra weights are considered to highlight the commonly used links among LOs based on the revised Hebbian rule. Then, inspired by the swarm intelligence algorithm, two algorithms were proposed and developed to generate candidate paths and allow users to reorganize preference paths. By using the prerequisite relation among LOs, as well as the attributes from these LOs, candidate paths can be generated. The generated paths are templates which can be reorganized by the users themselves, and can be stored back to the repository for further reference and reuse. The implemented system has demonstrated high usability and user satisfaction. We believe that the proposed mechanism can make LOs in a repository reach high reusability, by providing an automatic mechanism to reduce the load of lecture creation.

However, there are some limitations. First, metadata of LOs are required to be carefully specified by the creators. With less precise metadata, the system returns less accurate results. Second, due to the limitation of user experiences, only a limited prototype was demonstrated in this study.

In the future, in addition to solving the preceding limitations, an enhanced mechanism, based on instructional objectives, for lecture path generation is considered in a progressive way. Although educational theories tell us how lectures are designed to fit instructional purposes, a quantitative approach to measure effectiveness in the generation of learning activities is still difficult. We will try to give explicit description to LONET, and/or its subnetwork, to achieve specific instructional purposes, and continuously provide solutions to the challenges mentioned before.



You are not logged in. Login Register ▾ Language

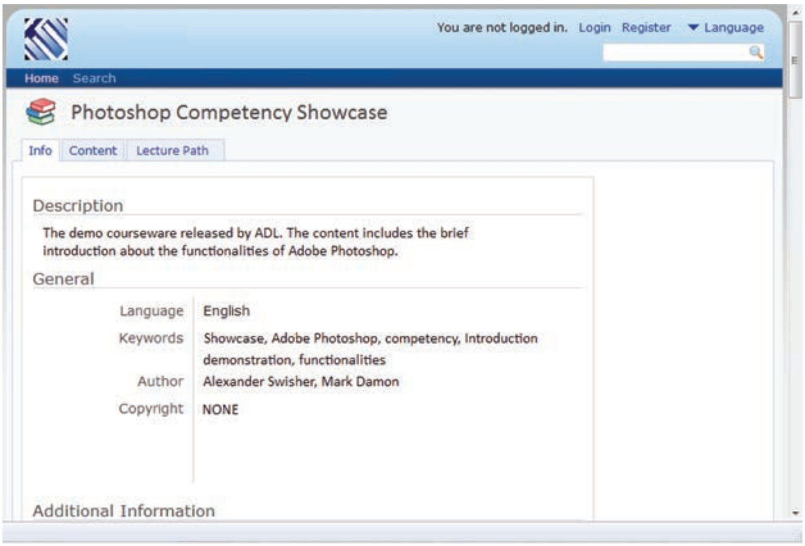
Home Search

Search Results

Previous | Next

Type	Title / Structure (similarity / diversity)	Ref()	Action
	Photoshop Competency Showcase Photoshop - Color Balance (1.00 / 0.09) Photoshop Benchmarking (0.92 / 0.02) Import and Exprot: An example (1.00 / 0.00) Capitalization in User Interface (0.90 / 0.07) 29 more nodes in current RT	0.696	▾ Actions
	ADL Demo Course - Photoshop	0.481	▾ Actions
	Photoshop Competency Resizing & Resampling (0.91 / 0.07) Photoshop Benchmarking (0.88 / 0.10) 11 more nodes in current RT	0.437	▾ Actions

Fig. 8. Search results in reusability tree.



You are not logged in. Login Register ▾ Language

Home Search

Photoshop Competency Showcase

Info Content Lecture Path

Description

The demo courseware released by ADL. The content includes the brief introduction about the functionalities of Adobe Photoshop.

General

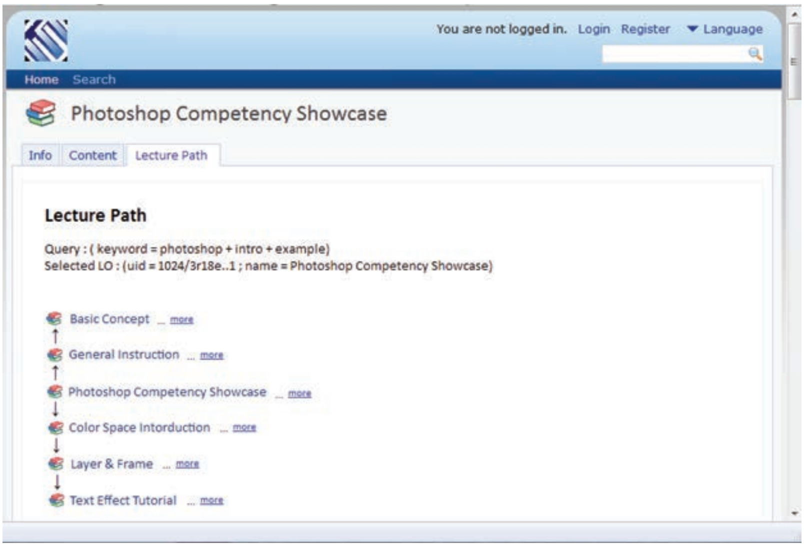
Language	English
Keywords	Showcase, Adobe Photoshop, competency, Introduction demonstration, functionalities
Author	Alexander Swisher, Mark Damon
Copyright	NONE

Additional Information

Fig. 9. Detailed information to selected learning object.

APPENDIX

In this appendix, a concrete example is addressed to give a demonstration in accessing our repository, known as the MINE registry. Most learning objects stored in our repository were collected from online training programs, such as Open University and MIT OpenCourseWare, and have been reused for generating lectures corresponding to specific topics. Here, we utilize learning objects in the major type "Photoshop" for demonstrating an example. We focus on the search service that instructors may confront and show the corresponding system interfaces. Interested readers are welcome to visit our official Web site at http://www.mine.tku.edu.tw/ for more detailed information.



Photoshop Competency Showcase

Lecture Path

Query : (keyword = photoshop + intro + example)
Selected LO : (uid = 1024/3r18e..1 ; name = Photoshop Competency Showcase)

Basic Concept ... more
↑
General Instruction ... more
↑
Photoshop Competency Showcase ... more
↓
Color Space Intorduction ... more
↓
Layer & Frame ... more
↓
Text Effect Tutorial ... more

Fig. 10. The lecture path based on selected LOs.

The elements described in LOM-based metadata are adopted as the basis of search criteria. In this example, we make a query by using a combination of keywords, "photoshop + intro + example."

The results are ranked in accordance with the ranking value "*Ref()*," and the similarity and diversity that assist instructors in realizing the degree of derivation are also shown next to each result.

Here we assume that the instructor selects the first result "Photoshop Competency Showcase." The detailed information includes the metadata description, its original content, and the possible lecture path for the related objective.

Once the instructor clicks the "Lecture Path" button in the upper menu, the possible composition lecture path is generated. The instructor can take this as a template, and make further modification by clicking the "more" button.

REFERENCES

ALBERT, R. AND BARABASHI, A. L. 1999. Emergence of scaling in random networks. *Science. 286,* 509–512.

BALABANOVIC, M. AND SHOHAM, Y. 1997. Fab: Content-Based, collaborative recommendation. *Comm. ACM 40*, 3, 66–72.

BATES, F. L. AND PEACOCK, W. G. 1989. Conceptualizing social structure: The misuse of classification in structural modeling. *Amer. Sociol. Rev. 54*, 565–577.

BRINGMANN, B., BERLINGERIO, M., BONCHI, F., AND GIONIS, A. 2010. Learning and predicting the evolution of social networks. *IEEE Intell. Syst. 25*, 4, 26–35.

BUCKLEY, C. AND VOORHEES, E. M. 2004. Retrieval evaluation with incomplete information. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. 25–32.

CHANG, J. S. AND LEE, W. S. 2003. Finding recent frequent item sets adaptively over online data streams. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*. 487–492.

CHEN, Y., DONG, G., HAN, J., WAH, B. W., AND WANG, J. 2002. Multidimensional regression analysis of time-series data streams. In *Proceedings of the International Conference on Very Large Databases*. 323–324.

CHEN, Y. P., PENG, W. C., AND JIAN, M. C. 2007. Recombination and dynamic linkage discovery. *IEEE Trans. Syst. Man Cybernet. 37*, 6, 1460–1470.

COST, S. AND SALZBERG, S. 1993. A weighted nearest neighbor algorithm for learning with symbolic features. *Mach. Learn. 10*, 57–78.

FREEMAN, L. C. 1979. Centrality in social networks: Conceptual clarification. *Social Netw. 1,* 3, 215–239.

FUJIMURA, K., INOUE, T., AND SUGISAKI, M. 2005. The eigenrumor algorithm for ranking blogs. In *Proceedings of the International Conference on World Wide Web*.

GASEVIC, D., JOVANOVIC, J., AND DEVEDZIC, V. 2007. Ontology-Based annotation of learning object content. *Interact. Learn. Environ. 15*, 1, 1–26.

GHANEA-HERCOCK, R. A., WANG, F., AND SUN, Y. 2006. Self-Organizing and adaptive peer-to-peer network. *IEEE Trans. Syst. Man Cybernet. 36*, 6,1230–1236.

HAYTHORNTHWALTE, C. 1996. Social network analysis: An approach and technique for the study of information exchange. *Library Inf. Sci. Res.18,* 4, 323–342.

HERLOCKER, J., KONSTAN, J., AND RIEDL, J. 2002. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Inf. Retr. 5*, 4, 287–310.

JARVELIN, K. AND KEKALAINEN, J. 2000. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the ACM International Conference on Information Retrieval*. 41–48.

JENSEN, D. AND NEVILLE, J. 2002. Data mining in social networks. In *Proceedings of the National Academy of Sciences Symposium on Dynamic Social Network Analysis*.

JOSEPH, S. 2002. . NeuroGrid: Semantically routing queries in peer-to-peer networks. In *Proceedings of the International Workshop on Peer-to-Peer Computing*. 202–214.

KHIERBEK, A., SALLOUM, S., AND TANNOUS, A. 2008. An inference network model for retrieving reusable learning objects. In *Proceedings of the International Conference on Information and Communication Technologies*. 1–5.

KLEINBERG, J. 1999. Authoritative sources in a hyperlinked environment. *J. ACM 46*, 5, 604–632.

KLEINBERG, J. 2000. Navigation in a small world. *Nature 406,* 845.

KLEINBERG, J. 2001. Small-World phenomena and the dynamics of information. *Adv. Neural Inf. Process. Syst. 14,* 1–14.

KOO, S. G., KANNAN, K., AND LEE, C. S. 2006. A genetic-algorithm-based neighbor-selection strategy for hybrid peer-to-peer networks. *Future Generat. Comput. Syst. 22,* 732–741.

KOPPI, T., BOGLE, L., AND BOGLE, M. 2005. Learning objects, repositories, sharing and reusability. *Open Learn. J. Open Dist. Learn. 20*, 1, 83–91.

KUMAR, R., RAGBAVAN, P., RAJAGOPALAN, S., AND TOMKINS, A. 2002. The web and social networks. *IEEE Comput. 35*, 11, 32–36.

LANGVILLE, A. N. AND MEYER, C. D. 2006. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press.

LIN, C. L. AND KAO, H. Y. 2010. Blog popularity mining using social interconnection analysis. *IEEE Internet Comput. 14*, 4, 41–49.

LIN, F. H., SHIH, T. K., AND KIM, W. 2009. An implementation of the CORDRA architecture enhanced for systematic reuse of learning objects. *IEEE Trans. Knowl. Data Engin. 21*, 6, 925–938.

LIN, S. AND CHALUPSKY, H. 2008. Discovering and explaining abnormal nodes in semantic graphs. *IEEE Trans. Knowl. Data Engin. 20*, 8, 1039–1052.

LIU, H., ABRAHAM, A., AND BADR, Y. 2009. Neighbor selection in peer-to-peer overlay networks: A swarm intelligence approach. *Comput. Comm. Netw. 4*, 405–431.

LOSER, A., STAAB, S., AND TEMPICH, C. 2007. Semantic social overlay networks. *IEEE J. Select. Areas Comm. 25*, 1, 5–14.

MOODY, J., MCFARLAND, D., AND BENDER-DEMOLL, S. 2005. Dynamic network visualization. *Amer. J. Sociol. 110*, 4, 1206–1241.

O'REILLY, T. 2007. What is web 2.0: Design patterns and business models for the next generation of software. *Int. J. Digit. Econ. 65,* 17–37.

OCHOA, X. AND DUVAL, E. 2008. Relevance ranking metrics for learning objects. *IEEE Trans. Learn. Technol. 1,* 1, 34–48.

PALLA, G., BARABASI, A. L., AND VICSEK, T. 2007. Quantifying social group evolution. *Nature 446,* 664–667.

RAGHAVAN, P. 2002. Social networks: From the web to the enterprise. *IEEE Internet Comput. 6,* 1, 91–94.

RIVERA, G. M., SIMON, B., QUEMADA, J., AND SALVACHUA, J. 2004. Improving LOM-based interoperability of learning repositories. In *On The Move to Meaningful Internet Systems*. 690–699.

ROCCHIO, J. 1971. Relevance feedback information retrieval. In *The Smart Retrieval System - Experiments in Automatic Document Processing,* G. Salton, Ed. 313–323.

ROWSTRON, A. AND DRUSCHEL, P. 2001. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms*. 329–350.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. 2001. Item-Based collaborative filtering recommendation algorithms. In *Proceedings of the International Conference on World Wide Web.* 285–295.

Scott, J. 1991. *Social Network Analysis: A Handbook*. Sage Publications.

Stanley, M. 1967. The small world problem. *Psychol. Today 1*, 1, 60–67.

Stent, G. S. 1973. A physiological mechanism for Hebb's postulate of learning. *Proc. Nat. Acad. Sci. 70,* 4, 997–1001.

Tolsdorf, C. C. 1976. Social networks, support, and coping: An exploratory study. *Fam. Process*. *15*, 4, 407–417.

Tsai, K. H., Chiu, T. K., Lee, M. C., and Wang, T. I. 2006. A learning objects recommendation model based on the preference and ontological approaches. In *Proceedings of the International Conference on Advanced Learning Technologies*. 36–40.

van Assche, F., Campbell, L. M., Rifon, L. A., and Willem, M. 2003. Semantic interoperability: Use of vocabularies with learning object metadata. In *Proceedings of the IEEE International Conference on Advanced Learning Technologies*. 511–514.

Vassiliadis, N., Kokoras, F., Vlahavas, I., and Sampson, D. 2003. *An Intelligent Educational Metadata Repository. Databases and Learning Systems*, CRC Press, Boca Raton, FL.

Wasserman, S. and Faust, K. 1994. *Social Network Analysis: Methods and Applications*. Cambridge University Press.

Wegener, B. 1991. Job mobility and social ties: Social resources, prior job, and status attainment. *Amer. Sociol. Rev. 56*, 60–71.

Wellman, B. and Berkowitz, S. D., Eds. 1988. *Social Structures: A Network Approach*. Cambridge University Press.

Weng, C. Y., Chu, W. T., and Wu, J. L. 2009. RoleNet: Movie analysis from the perspective of social netowrks. *IEEE Trans. Multimedia 11*, 2, 256–271.

Wolf, T., Schroter, A., Damian, D., Panjer, L. D., and Nguyen, T. H. D. 2009. Mining task-based social networks to explore collaboration in software teams. *IEEE Softw. 26,* 1, 58–66.

Wu, H. C., Luk, R. W. P., Wong, K. F., and Kwok, K. L. 2008. Interpreting TF-IDF term weights as making relevance decisions. *ACM Trans. Inf. Syst. 26*, 3, 13–49.

Yang, T. D., Chiu, C. H., Tsai, C. Y., and Wu, T. H. 2004. Visualized online simple sequencing authoring tool for SCORM-compliant content package. In *Proceedings of the IEEE International Conference on Advanced Learning Technologies*. 609–613.

Yang, Y., Yoo, S. J., Lin, F., and Moon, I. C. 2010. Personalized email prioritization based on content and social network analysis. *IEEE Intell. Syst. 25*, 4, 12–18.

Yen, N. Y., Shih, T. K., Chao, L. R., and Jin, Q. 2010. Ranking metrics and search guidance for learning object repository. *IEEE Trans. Learn. Technol. 3*, 3, 250–264.

Yen, N. Y., Shih, T. K., Jin, Q., Hung, J. C., Zhou, Q., and Chao, L. R. 2011. Adaptive object re-ranking mechanism for ubiquitous learning environment. *J. Multimedia 6,* 2, 129–138.

Received October 2010; revised January 2011; accepted April 2011