Wong, L.-H., & Looi, C.-K. (2009). Adaptable Learning Pathway Generation with Ant Colony Optimization. *Educational Technology & Society*, *12* (3), 309–326.

Adaptable Learning Pathway Generation with Ant Colony Optimization

Lung-Hsiang Wong and Chee-Kit Looi
National Institute of Education, Singapore // lunghsiang.wong@nie.edu.sg, cheekit.looi@nie.edu.sg

ABSTRACT

One of the new major directions in research on web-based educational systems is the notion of adaptability: the educational system adapts itself to the learning profile, preferences and ability of the student. In this paper, we look into the issues of providing adaptability with respect to learning pathways. We explore the state of the art with respective to deriving the most apt learning pathway to recommend to the learner. Our proposal suggests a novel way of modeling learning pathways that combines rule-based prescriptive planning, which could be found in many of the classic Intelligent Tutoring Systems, and Ant Colony Optimization-based inductive planning, for recommending learning paths by stochastically computing past learners' traveled paths and their performances. A web-based prototype has been developed using C# and .NET technologies.

Keywords

Web-based learning, Learning pathway planning, Course sequencing, Personalized e-Learning, Swarm Intelligence (Self-Organizing Agents), Ant Colony Optimization, Learning on Demand

Introduction

With the proliferation of the usage of Internet technologies, many learner-centric e-learning portals have been launched. To cater for working adults' needs, the notion of "Learning on Demand" (LoD), defined by SRI Consulting Business Intelligence (n.d.) as "the process of using technology to enable and encourage workers, managers, and executives to learn and acquire new skills while resolving organization's problems", has emerged. For example, if an inexperienced Java programmer needs to implement a clock Applet, she need not study the entire Java programming guide. She only needs to learn about those few relevant Java library packages plus some other prerequisite knowledge. An advanced e-learning portal should provide such flexibility for the learners based on their learning goals.

We propose Dynamic Learning Path Advisor (DYLPA), a set of course sequencing algorithms that combine both prescriptive navigation rules and an inductive mechanism. Our research effort focuses on the latter, where we propose a mechanism that is inspired by a specific technique in an agent-based soft computing technology field called "swarm intelligence" or "self-organizing agents" (Bonabeau, Dorigo & Theraulaz, 1999; Engelbrecht, 2006). Swarm intelligence comes in various forms like Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), wasp task differentiations, and so on. In particular, a modified ACO mechanism has been developed for DYLPA. The prescriptive rules, proposed by earlier researchers, form the complementary component to the induction mechanism that we propose.

The learning path optimization process starts from the launch of a new course. At the early stage, without substantial amount of learner history, DYLPA recommends a learning path based on the prescriptive rules. However, DYLPA would not make the recommendation a strict prescription and instead allow the learner to explore an alternative path at her own will. The learner's performance in learning from each object will be measured and stored.

The learners' performance logs are thus gradually built up. By satisfying certain condition, the induction mechanism is triggered. When the next new learner logs on, the system will select up to a specific number of previous learners (or "alumni") who have similar profiles. The paths they have taken and their performances are analyzed by the swarm intelligence technique to "induce" a path for the new learner. However, the "prescriptive path" is not totally discarded. Instead, it is treated as a "virtual learner's" choice of a path that carries certain weight in the computation. Hence, the combination of rule-based prescription and stochastic computation in path selection is achieved. We believe that our approach is one of the first course sequencing techniques that explicitly combine prescriptive and inductive planning.

Literature Review

To facilitate Learning on Demand in the context of an e-learning portal, what comes into our mind is the "classic" Intelligent Tutoring System's (ITS) rule-based course sequencing techniques like Peachey & McCalla's (1986) curriculum planning techniques and Study Advisor (Arshard, 1989) where a learning path, i.e., a series of ordered learning objects, is chosen given a learning goal. One underlying weakness of the former, albeit its capability of generating learning path tailored to individual's current knowledge state, is that the navigation is based on a single rule: learn prerequisite knowledge first. The Study Advisor improves upon that by adding more metadata types (relationships between learning objects) to enable alternative path selection to suit learners with different learning styles, e.g., learners who prefer "generalization before specialization", or vice versa.

Some variations of such techniques in e-learning environments are reported in more recent literature (Karagiannidis, Sampson, & Cardinalli, 2001; Carchiolo, Longheu & Malgeri, 2002; Chen, Liu & Chang, 2006). Carchiolo *et al.*'s (2002) system, in particular, moves away from prior rule-based planning and instead assigns weights to individual arcs (each arc links a pair of learning objects in the course topology) according to how "suitable" individual paths are for a particular learner with respect to her profile, and subsequently makes use of the weights to recommend "the next learning object" upon the learner's completion of the current one.

However, prescriptive rules are typically designed based on commonsense or expert notions of how a learning path should be chosen, which are not always valid and could be relatively rigid. Brookfield (1985) suggests an alternative approach, "successful self-directed learners… place their learning within a social setting in which the advice, information, and skill modeling provided by other learners are crucial conditions for successful learning." The argument finds echoes in the information navigation literature, where the term "social navigation" (Höök & Benyon, 2003) has been coined to describe research reflecting the fact that "navigation is a social and frequently a collaborative process." (Dieberger, Höök, Svensson & Lönnqvist, 2001) In particular, "indirect social navigation" exploits traces of interactions left by others and can be used as the basis of a recommendation system (Shipman, Furuta, Brenner, Chung & Hsieh, 2000).

In the light of the alternative notion, Internet-based e-learning technology facilitates convenient collection and analysis of the activity logs, the performances and the results of a large pool of learners. That opens the door for researchers to look into data mining-like, constructive induction mechanisms where the historical data of previous learners' learning paths or actual performances can serve as a basis in selecting learning paths for new learners.

For example, the Web "tour guide" agent WebWatcher (Joachims, Freitag & Mitchell, 1997) accompanies thousands of users from page to page as they browse the web, and generate a "path", which is an ordered set of webpages, for each particular topic that maximizes the amount of relevant information encountered. Similarly, the Tutorlet agents (Niemczyk, 2000) employ Hidden Markov Models (HMMs) (MacDonald & Zucchini, 1997) to develop a statistical model of student behaviors based on past students' choices of learning paths. Such agents recommend the next learning media type (e.g., text, video, simulation, etc.) to a new student after she finishes her activity under the current media type. The e-learning portal Knowledge Sea (Farzan & Brusilovsky, 2005) works in the same way as WebWatcher except that a Time Spent Reading (TSR) (e.g., Claypool *et al.*, 2001) algorithm was used. Huang, Chen & Cheng (2007) proposes an improved approach that makes use of FP-tree technique to make multiple levels of abstract suggestions instead of single level frequent pattern mining results as its predecessors do.

Our proposed DYLPA algorithm is one of the earliest applications of the swarm intelligence techniques in course sequencing that make use of past learners' performances (not just the paths they have traveled) as the basis for path recommendations. After an extensive literature search for related work, we only managed to identify four such systems (Gutiérrez, Pardo & Kloos, 2006; Van den Berg, *et al.,* 2005; Vaigiani, *et al.*, 2005; Wang, Wang & Huang, 2008). We will compare our approach with theirs in a later section of this paper.

Ant Colony Optimization and its Applications

In the natural phenomenon of "Ant Colony Optimization" (ACO), ants construct networks of paths that connect their nests with available food sources. These networks form minimum spanning trees, minimizing the energy ants expend in bringing food into the nest. This globally optimal structure emerges from the simple actions of the individual ants

(Parunak, 1997; Dorigo & Stützle, 2004). Steels (1991) proposes a model where all ants share the same set of five rules to govern their actions:Avoid obstacles.

1. Wander randomly, in general direction of any nearby ***pheromones*** (scent markers that many insects generate).
2. If the ant is holding food, drop food pheromone while looking for and following a beacon, e.g., nest pheromone that leads in the general direction of nest. If the ant is not holding food, drop nest pheromone while looking for and following a food pheromone trail.
3. If the ant finds itself at food and is not holding any, pick the food up.
4. If the ant finds itself at the nest and is carrying food, drop the food.

Because only food-carrying ants drop food pheromone, all food pheromone paths lead to a food source. Once a full ant finds its way home, there will be nest pheromone paths that lead. The initial path will not be straight, but the tendency of ants to wander even in the presence of pheromones will generate shortcuts across initial meanders.

The ant path planning mechanism has inspired algorithms (e.g., Dorigo, 1992; Bonabeau, Dorigo & Theraulaz, 1999) for planning routes in the Travelling Salesman Problem (TSP), i.e., to find a tour of minimal length connecting *n* cities; each city must be visited once and only once. In the case where the number of cities is huge, soft computing techniques like genetic algorithms or ACO become better alternatives than conventional AI search techniques in solving such a NP-hard problem.

In an algorithm proposed by Dorigo (1992), for example, a huge set of ant-like agents are moving on the problem graph until each of them completes a tour. Initially, each agent selects the next connected city randomly when it reaches one city and maintains a memory of the cities it has visited and the distance of each edge. After completing a tour, it lays a quantity of pheromone Q/L on each edge that it has used, where Q is a parameter and L the total length of the tour.

With the pheromone being accumulated, each of the subsequent agents' selection of the next city to visit when it reaches one city will be influenced by the pheromone of each corresponding edge. There is a probability factor here – the greater the value of a particular edge's pheromone, the higher the probability the agent selects the edge. This is the key mechanism to "balance exploitation and exploration" of alternative solutions, the essence of swarm intelligence, rather than exploiting the presently known best solution all the way.

The method could not perform well without pheromone decay, or it would lead to the amplification of the initial random fluctuations, which very probably would not be optimal. Therefore, the quantity of pheromone that was "deposited" earlier should be decayed as time goes by.

Some variations of TSP ACO Algorithms have also been applied to a similar problem – communications network routing (e.g, Caro & Dorigo, 1998; Di Caro, 2004; Wedde & Farooq, 2006). Such algorithms offer better results than conventional AI search techniques in handling dynamic network conditions, e.g., the optimal path now may be congested later.

From Ant Colony Optimization to Learning Paths

As described before, "conventional" learning path planning can be viewed as finding a path in a course network that leads to one or more learning goal(s), which is analogous to the destination node in network routing. As our research objective is to derive a stochastic mechanism that involves the alumni's selection of paths and actual performances, the ACO metaphor has offered a plausible and perhaps powerful solution – the pheromones.

In adapting ACO for DYLPA, we consider a few issues:

- ***"Populating" the course network***. Whereas ACO Algorithms generate simulated agents to wander about the network and measure the cost (e.g., travel time) on each edge, DYLPA intends to "induce" good learning path from actual alumni's performances. Therefore, in DYLPA, each of the selected alumni is treated as an ant-like agent that moves on the course network and deposit pheromone along her way.
- ***Similarity of the ant-like agents.*** The ACO Algorithms always populate each search space with homogeneous agents. With this concern in mind, DYLPA needs alumni that are similar to the current learner to populate the course network so that the recommendation could tailor to individual learners' needs or preferences. However,

since learners with diversified background or preferences could be enrolled in the same course, DYLPA has to re-select "training cases" and re-compute the pheromones each time a new learner is enrolled.

- ***Where the pheromones are generated***. In many ACO-based applications like network routing, each node is only a point where agents drop by while the real performance measurement is the time spent on each edge. In a course network, however, a learner actually spends her time when attempting the nodes (learning objects) while an arc merely reflects the relationship between two nodes. Moreover, a learner's performance at a node is often influenced by the combination and the sequence of the nodes she had visited prior to that. Therefore, the pheromone generated at a node should reflect the "goodness" of the previous arc. For example, if a learner visits node *j* right after node *i*, the pheromone generated at node j should be attributed to arc (*i*, *j*) instead of any "outbound" arc of node *j*. However, at which node a pheromone value should be stored could become tricky when the system needs to make use of it for recommendations. We will address this issue in a later section.
- ***Whether the time-decaying feature of the pheromones should be incorporated into DYLPA***. The reasons such a feature is incorporated in the original ACO Algorithms are :- (a) To let the earlier agents' findings (which are essentially unguided since the pheromones have not been substantially accumulated yet) gradually evaporate; (b) To cope with the dynamic environment in certain applications (e.g., the communications network traffic).
- Reason (a) is relevant to DYLPA in an indirect sense that when a greater set of historical data is accumulated, the number of learners that are more "similar" to the current learner tends to increase, resulting in more "accurate" pheromones/recommendations. Reason (b) is also valid in DYLPA because of the possible updates of the course content such as addition or deletion of learning nodes, and amendment of the contents in the existing learning nodes – which is not unusual for e-learning portals in the 21st century. Such updates might affect the overall performances of "similar" learners who are enrolled in the course at different periods of time, making the previously deposited pheromones less accurate in predicting the performances of new learners.

Two important assumptions of DYLPA are:

- There are a lot of learners enrolled in an e-course or a set of overlapping e-courses at different periods of time to guarantee sufficient "training data"; and
- Many learners are adventurous enough not to always follow system recommendations - otherwise, DYLPA will keep on "exploiting" the prescriptive path instead of "exploring" alternative paths.

For assumption (2), the "rebellious mindset" is more likely to be found in adult learners than young students, especially if those who are attempting Learning on Demand are time pressed to accomplish their learning goals. Therefore, we predict that DYLPA would work better for adult learning.

Learner Profiles in a DYLPA-based System

A learner profile, the key resource for facilitating the DYLPA process, consists of two components: (1) learner attributes; (2) activity log.

"Learner attribute" is a way of quantifying a learner's relevant background, prior knowledge, learning preferences, and other learner information. The system needs it to compute the *similarity levels* between individual alumni and the current learner whom the system recommends a new learning path to. We refer to the latter as "target learner".

There are two ways to obtain such information: (1) each learner fills up a pre-enrolment questionnaire and/or go through a pre-assessment (for both the alumni and the new learner); (2) the system to analyze a learner's activity log and performances after she has finished the course (for the alumni only). The learner attributes and their grading schemes are:

- *Prior knowledge and skill set* (as specified by e-course designer): Each learner grades her competency on individual knowledge/skill in the scale of 1 (never learnt) to 5 (excellent), and/or go through a pre-assessment (the results will be normalized and rounded up to an integer between 1 and 5);
- *Learning Preference I – Concept-Task spectrum*: Each learner grades her preference, emphasis and competency in learning conceptual or practical knowledge (1 to 5);
- *Learning Preference II – Specialization-Generalization*: Each learner specifies whether she prefers to learn specialized before generalized knowledge (1), or vice versa (5);
- *Learning Preference III – Abstract-Concrete*: Each learner specifies whether she prefers to learn concrete before abstract knowledge (1), or vice versa (5);

- *Learning Preference IV – Free-Guided navigation*: Each learner grades her willingness to comply to the system's recommendations of the subsequent nodes to visit, or to explore the course network at her own will, in the scale of 1 (guided navigation) to 5 (free navigation), with 3 as no preference or "not sure";
- *Analytical skill competency:* Each learner grades her ability in understanding diagrams, flowcharts, etc., in the scale of 1 (poor) to 5 (excellent);
- *Language proficiency:* Each learner grades her proficiency of the language medium used in the e-course, in the scale of 1 (very poor) to 5 (excellent);
- *Similarity of occupations*: Since a learner's occupation might affect her expectation in what knowledge or skill to pick up in the course, and her learning styles, we will work out a semantic diagram that incorporates popular occupations and provides the similarity grade (1 to 5) between each pair of occupation.

The learner attribute set can be expanded to include more factors, e.g., other types of learning preferences, learner's educational qualifications, etc.

On the other hand, the learner's activity log records the learning path that she has visited and the assessment results (performances). To facilitate DYLPA pheromone computation, the learner's assessment results must be stored in node-by-node basis – see Table 1 for an example.

Table 1: An example of learner's activity log

Nodes	Performances
a	7 (out of 10)
c	9
e	6
b	5
g	6

The example shows that the learning path is a (performance = 7) → c (9) → e (6) → b (5) → g (6).

Overview of the DYLPA Process

The DYLPA learning path optimization process is an ongoing process. The entire mechanism can be summarized as the following:

- Recommend a learning path to a new target learner at the early stage of a new course:
  - The human course administrator specifies a "DYLPA training size" *n* and a "DYLPA training threshold" *t* (see below)
  - Without substantial amount of learner history, DYLPA recommends but does not reinforce learning paths based on prescriptive rules
  - Past learners' ("alumni") paths and performances are measured and stored
- Recommend a learning path to a new target learner when the number of alumni > 2*n*:
  Phase I: Pheromone computation (before the target learner starts to study the course)
  - Computes the *similarity levels* between individual alumni and the target learner
  - Selects up to (*n-1)* alumni whose similarity levels are higher than *t*
  - Generates ant-like agents corresponding to individual selected alumni to traverse the course network and deposit pheromones according to the alumni's performances
  - Generates an extra ant-like prescriptive agent to traverse the course network according to the learning pathway recommended by the prescriptive rules

  Phase II: Recommend the next node (after the target learner starts to study the course)
  - Each time when the target learner completes studying one node, the system selects and recommends the next node from all the next possible nodes; the higher the pheromones associated to a "next node", the higher probability that the node will be selected.

The detailed DYLPA recommendation process is elaborated as below.

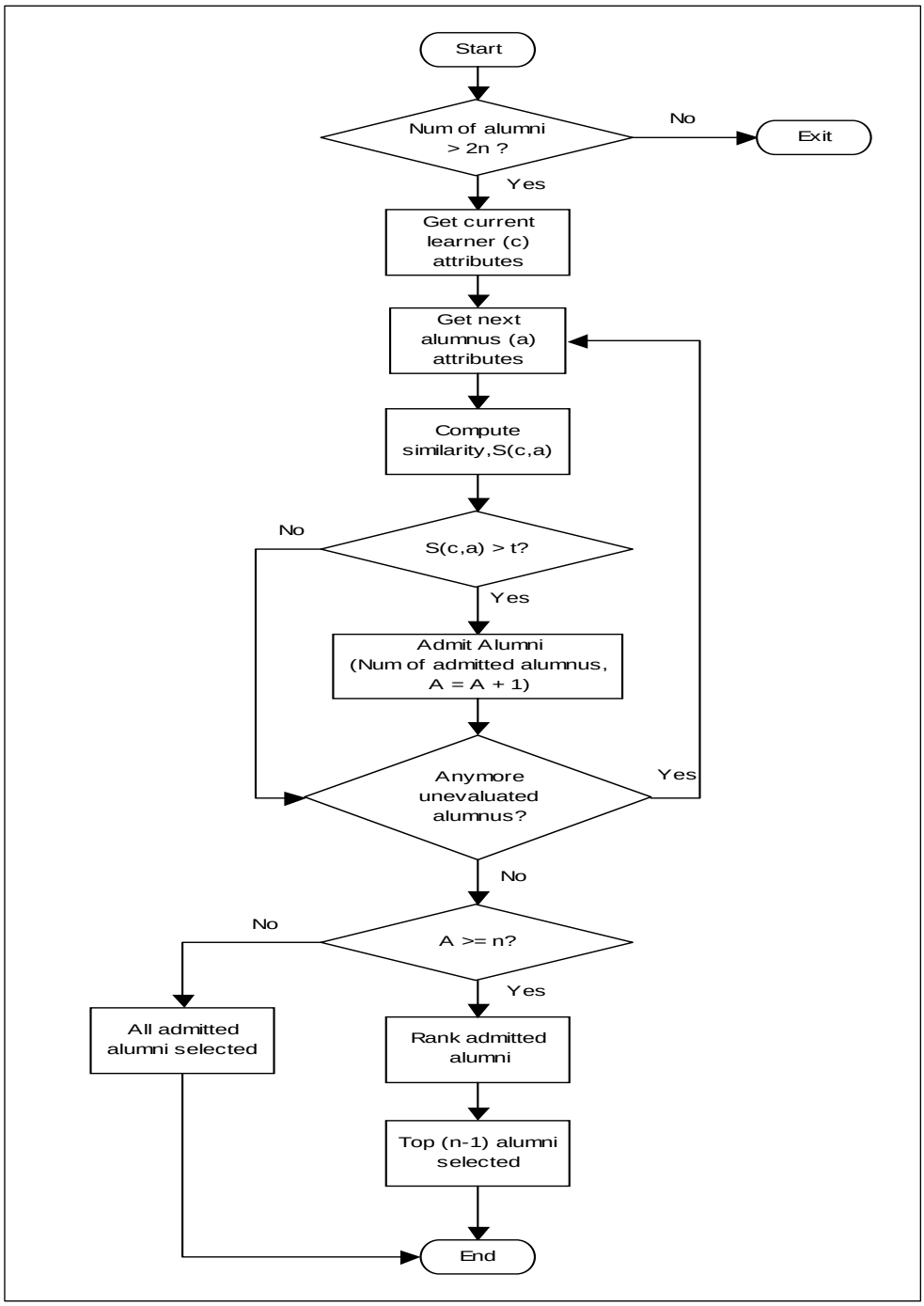

Figure 1: Control diagram of the process of selecting similar alumni

At the Early Stage of an e-Course

When a new course is launched, there is no learner history for DYLPA to compute the pheromones. Therefore, the system can only recommend a learning path based on built-in prescriptive rules. In principle, our system only recommends the next node after a learner has finished working on the current node, albeit disclosing the complete recommended path in one shot is also fine. However, since it is not mandatory for the learner to comply with the recommendation, the system may need to re-determine the shortest learning path upon her completion of the next node and to subsequently recommend the following node.

So far, the system works in exactly the same way as classic ITS content sequencing techniques. At the same time, learners' learning paths and performances in the course are logged and analyzed.

As time goes by, more and more learners would have "graduated" from the course. The DYLPA algorithm may then be triggered, depending on the number of the alumni and the DYLPA training size that the course administrator has preset. For example, if the DYPLA training size $n = 50$, the administrator may want to trigger the algorithm when the number of alumni surpasses $2n$ *(= 100)*, so that the algorithm would have better chances to identify the subset of alumni who have greater degree of similarity of each new learner.

Selecting Similar Alumni for Training

After the DYLPA algorithm has been triggered, the system will first select a set of similar alumni to compute the pheromones. Figure 1 depicts the control diagram of the mechanism.

The DYLPA's method of identifying similar alumni is inspired by the work by Tang & Chan (2002), which is a quantification technique to cluster a set of students, i.e., homogeneous student group forming for web-based collaborative learning. The technique has been looking into simplifying them to the set of learner attributes that will be input to the DYLPA unit for computing the similarity. However, for performance reason, especially that if there are thousands of alumni are available for comparisons, we adopt a simple grading scheme in the scale of 1 to 5 to quantify each learner attribute. The learner attributes in DYLPA have been elaborated in section 4.1 of this paper.

In essence, DYLPA computes the similarity level S(c, a) between the target learner (c) and a given alumnus (a) by the normalized weighted Euclidean distance of their corresponding learner attributes,

$$S(c,a) = \begin{cases} 1 & ,D = 0 \\ [\mathrm{Log}(D)]^{-0.5} & ,D > 0 \end{cases}$$

In the formula, x is the quantified value of each learner attribute (each type of prior knowledge or skill is treated as one learner attribute), and w is a weight assigned to each learner attribute, since there should be different degrees of importance for different learner attributes for the comparison. As this is the preliminary version of the DYLPA algorithm, we assign *ad-hoc* weights to the learner attributes, that is, $w = 1$ for each prior knowledge and skill type, and $w = 3$ (if the number of the prior knowledge/skill types is ≤ 5) or $w = 5$ (otherwise) for each of the rest of the learner attributes.

$w_t(t_c-t_a)^2$ is where we incorporate the time-decaying feature of pheromones. The time gap between the target learner (t_c) and an alumnus (t_a) in terms of number of months is/was enrolled in the e-course is squared. The *ad-hoc* weight of the attribute is 5.

Note that the formula itself is a variation of the typical formula of TSP ACO, that is, $S(c, a) = [\sum w_x(x_c-x_a)^2+w_t(t_c-t_a)^{-2}]^{-0.5}$ (where (t_c-t_a) is *inversed* and squared). In our formula, the Euclidean Distance for the time decay is squared (*without* being inversed!) before multiplied by its weight. This is to ensure that both the Euclidean distances of the attributes and time decay have the inversed proportional effects on the degree of the similarity. Next, as the computed values of D is generally high (say, > 100), by applying inverse and square root to the stated values will produce very small similarity values (say, < 0.1). Hence, an even spread of the similarity value between 0 (most similar) and 1 (least similar) could be achieved.

Therefore, instead of TSP ACO and communications network router's ongoing updates of the pheromones in the entire environment throughout the whole course of agent-based simulations where newly computed pheromone of each edge is added to a fraction of the original value (analogous to reinforcement learning), DYLPA handles time decaying while selecting the "ant-like agents" (i.e., the alumni).
After that, the alumni will be ranked in the order of individual's similarity with the target learner. Here, we define another parameter – DYLPA training threshold *t*, which is 0.5 or any smaller value if greater similarity is desired.

This serves as a filter for "agent" selection – alumni whose S < *t* will NOT be selected. The top-(*n*-1) (*n* = DYLPA training size) alumni whose S ≥ *t* will be selected as agents.

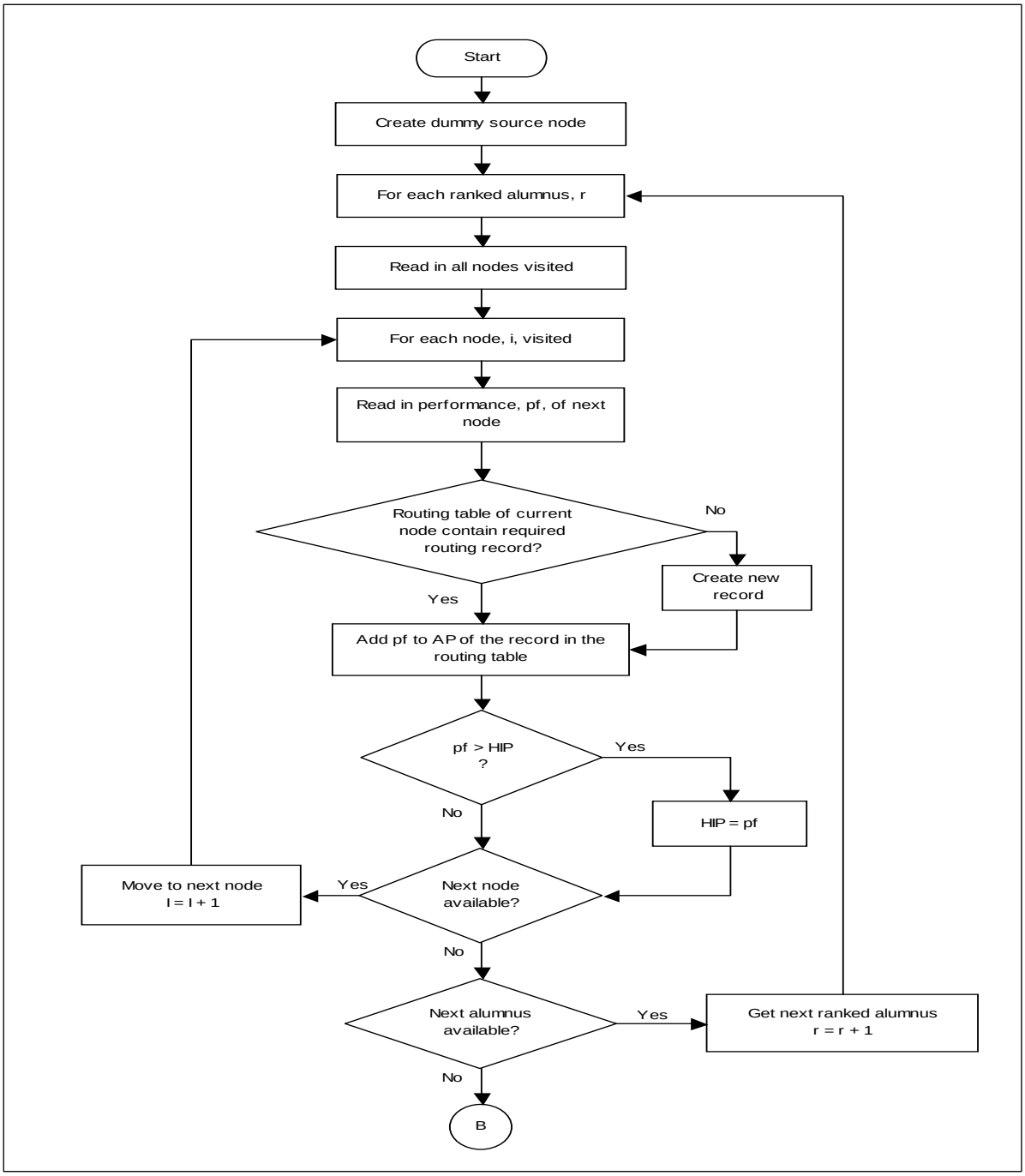

Figure 2a: Control diagram of DYLPA ant path planning

There is still one last "virtual alumnus" who complies with the "prescriptive path" and this will be treated as the $(n)^{th}$ agent. We refer to it as "prescriptive agent". In the case where less than *n* alumni (say, *k* number of them) satisfy the condition of S ≥ *t*, we will set the weight of the prescriptive agent's pheromones in the computation as (*n*-*k*). In the situation where there are sufficient number of alumni satisfies S ≥ *t*, the system will set k = 1.

In short, the pathway recommendation mechanism starts with having prescriptive rules dominating the planning process. As time goes by, the amount of alumni that satisfy the condition may also increase, resulting in gradual decrease of the significance (the weight) of the prescriptive rules in DYLPA planning.

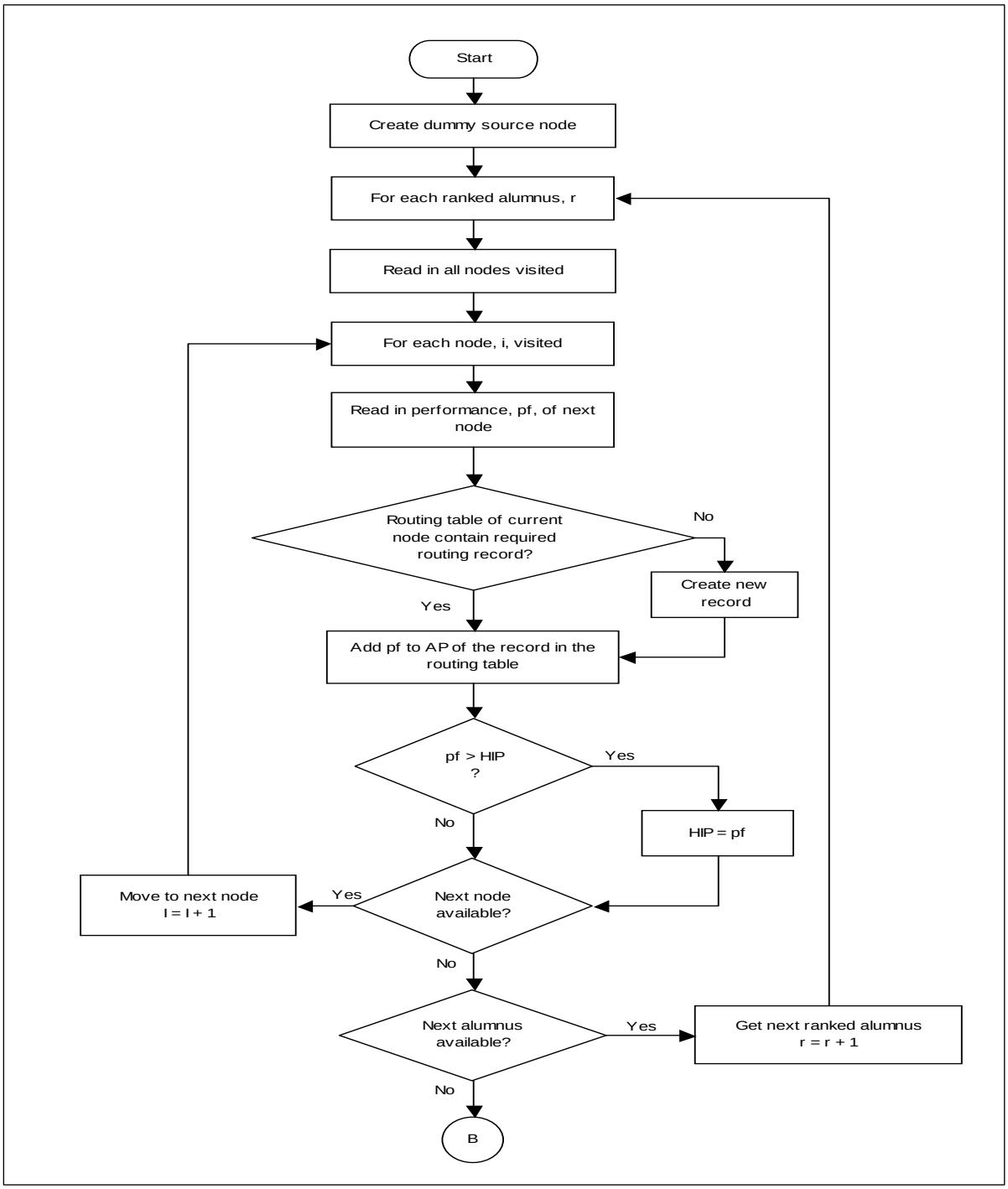

Figure 2b: Control diagram of DYLPA ant path planning

Computing the Pheromones

Now that the set of agents has been identified, DYLPA is ready to compute the pheromones. Figure 2a & 2b depict the control diagrams of the pheromone computation process.

Since the learners are given the flexibility to choose whatever next node they want to move to upon the completion of the current node, some "adjacent" pairs of nodes a learner has visited may not have "prescriptive relationship", e.g., prerequisite, abstract-concrete, etc., at all. There is no problem for DYLPA to handle such cases by assuming that the entire course network is fully connected, i.e., any given node could link to all other nodes in principle, and all arcs are bi-directional.

On the other hand, as stated before, the pheromone stored at a node should reflect the "goodness" of the previous arc in theory. However, when DYPLA makes use of the pheromones to plan the path, a more intuitive way is ACO's representation of "the probability or 'goodness' of traveling from current node *i* to destination node *d* via the next node *j*", which should be stored in *i* instead of *j*, even though the pheromone is actually a function of the learner's performance at *j*.

In essence, each node maintains a "local routing table" as shown in Table 2.

Table 2: Routing table of node *i* to learning goal node *d*

Via (neighboring node)	Accumulated pheromone value (AP)	Highest individual pheromone value (HIP)
J	…	…
K	…	…
M	…	…

The routing table is "local" because it only suggests what the "best" next node is, given the goal node of the course. After the learner moves to the next node, it is "none of the (current) node's business" anymore. On the contrary, conventional routing tables usually store the full path between each source-destination pair of nodes.

Prior to the pheromone computation process, the routing table of each node in the course network is blank. Then, DYLPA computes the pheromones based on selected alumni's history. For example, an alumnus has chosen node *j* after finishing *i*, and her performance measurement in *j* is *pf* – how to compute this will be described later. Therefore, the system will check whether the routing table of node *i* (NOT node *j*) has stored the pheromone of "to *d* via *j*". If so, *pf* will be added to the current pheromone value (that is, the Accumulated Pheromone Value or AP field) of this particular record. If not, a new record of "via *j*" will be created and hold the initial pheromone value of *pf*. The "highest individual pheromone value" (HIP) should store the highest *pf* value among the alumni who have visited j via i. This field will become useful later. The process continues until the entire alumni log has been read in and processed.

On the other hand, since the first node an alumnus visited has no "previous node", the system will create a "*dummy source node*" as the starting point of all the alumni.

After that, the "prescriptive agent's" path is incorporated into the pheromone computation. Since the path is supposed to be the "best" path in theory, the system "assigns" to the *pf* value on each node the prescriptive agent has visited by taking the HIP (given the previous node it has visited). However, in case "the prescriptive arc (e.g., from node *i* to node *j*)" has no pheromone value (i.e., has never been visited by any alumnus), the system will take the average value of all the HIP values of the other arcs on the same routing table as the *pf* value of the arc. Finally, each *pf* value of the prescriptive agent is multiplied by the weight of the prescriptive agent and then added to the AP on the corresponding routing table (or the system will create a new record if necessary). After that, the HIP fields on all the routing tables can be discarded to free the memory space as they are no longer needed.

How are alumni's performances measured? The performance measurement could consist of one or any combination of, but not restricted to, the following components, subjected to them being generated or made available:

1. Results of the assessments of individual nodes the alumni have visited;
2. Results of the post-assessment of the entire course: we need to split the results of individual questions to their corresponding nodes (However, if an assessment does not cover all the visited nodes, then this component should not be incorporated);
3. Learner' self-assessment at the individual nodes.

Each component is assigned a weight and all the weighted values will be added together to yield *pf*.

An alternative way of computing *pf* is to combine together the assessment result, if applicable, of and time spent on the same node by taking (weight * (result / time_spent)). There is still a weight component here to reflect different levels of significance of the node.

Recommending the Next Node

Figure 3 shows the control diagram of how the DYLPA algorithm recommends the next node upon a learner's completion of "learning" one node.

The pheromone computation with respect to a target learner has to be completed before she accesses the first node; and the pheromones will "follow" her until she finishes or quits the course. However, if she spends months on the course and other learners have completed the course along the way, the pheromones can be re-computed in the middle of her enrollment to yield better results because of potentially more "similar" alumni. The new pheromones will not affect the "validity" of what the learner has gone through prior to the re-computation due to the local planning nature of the technique.

At the beginning, the DYLPA System checks the routing table of the dummy source node to recommend the actual first node to the new learner. The relative pheromone value of each entry (i.e., the next node to go) in the table will determine the probability of the corresponding arc being chosen. For example (as in Table 3),

Table 3: Routing table of dummy node to learning goal node *d*:

Via (next node)	Accumulated pheromone value (AP)
J	13.5
K	5.1
M	40.3
N	2.2
P	7.6

Therefore, the probability of choosing node *m* is 40.3/(13.5+5.1+40.3+2.2+7.6) ≈ 0.604 or 60.4%

In other words, the DYLPA System constructs a roulette wheel that is weighted by AP values. It then spins the wheel to choose the next node to recommend. The learner will decide whether she will comply with the recommendation or make an alternative choice of her own. After she has completed working on the next node, the process starts all over again to recommend the following node, until the learner reaches the goal node.

There are a few potential variations of the process:

- Not all the "next nodes" in a routing table may "compete" for being recommended. Those with AP value that is below a pre-defined threshold (e.g., smaller than ¼ of the highest AP value) or a cutoff point (e.g., the "next nodes" are ranked in the order of their AP values and those at the bottom half will be discarded, unless there are less than 4 candidates) will not be included to the roulette wheel.
- Instead of constructing a roulette wheel, list all the "next node" options, in the order of their AP values, and displays their probability values to the learner, and let her decide which node to proceed to or even choose a node which is not in the list.
- After the learner has been recommended on the next node to visit (or picked her own node), she can request DYLPA to pre-compute and display the complete path to the goal node via that particular node by applying the pheromone-based mechanism (or firing the prescriptive rules, if the DYLPA algorithm has not been triggered yet). This will provide the learner a reference of what she can expect ahead.

Prototype Development and Testing

We have developed a software prototype of the DYLPA System. It is a web-based software running on Microsoft Windows 2000/XP that is interfaced with a simulated web learning portal. The core pathway planning mechanism is written in C#. The ASP .NET framework is used to develop the web interface so that the prototype is portable to real

e-learning portals that comply with the framework. The simulated web-learning portal contains a Microsoft Access database that stores test data sets, i.e., individual alumni's background and e-learning history. The overall architecture of the prototype is depicted in Figure 4.

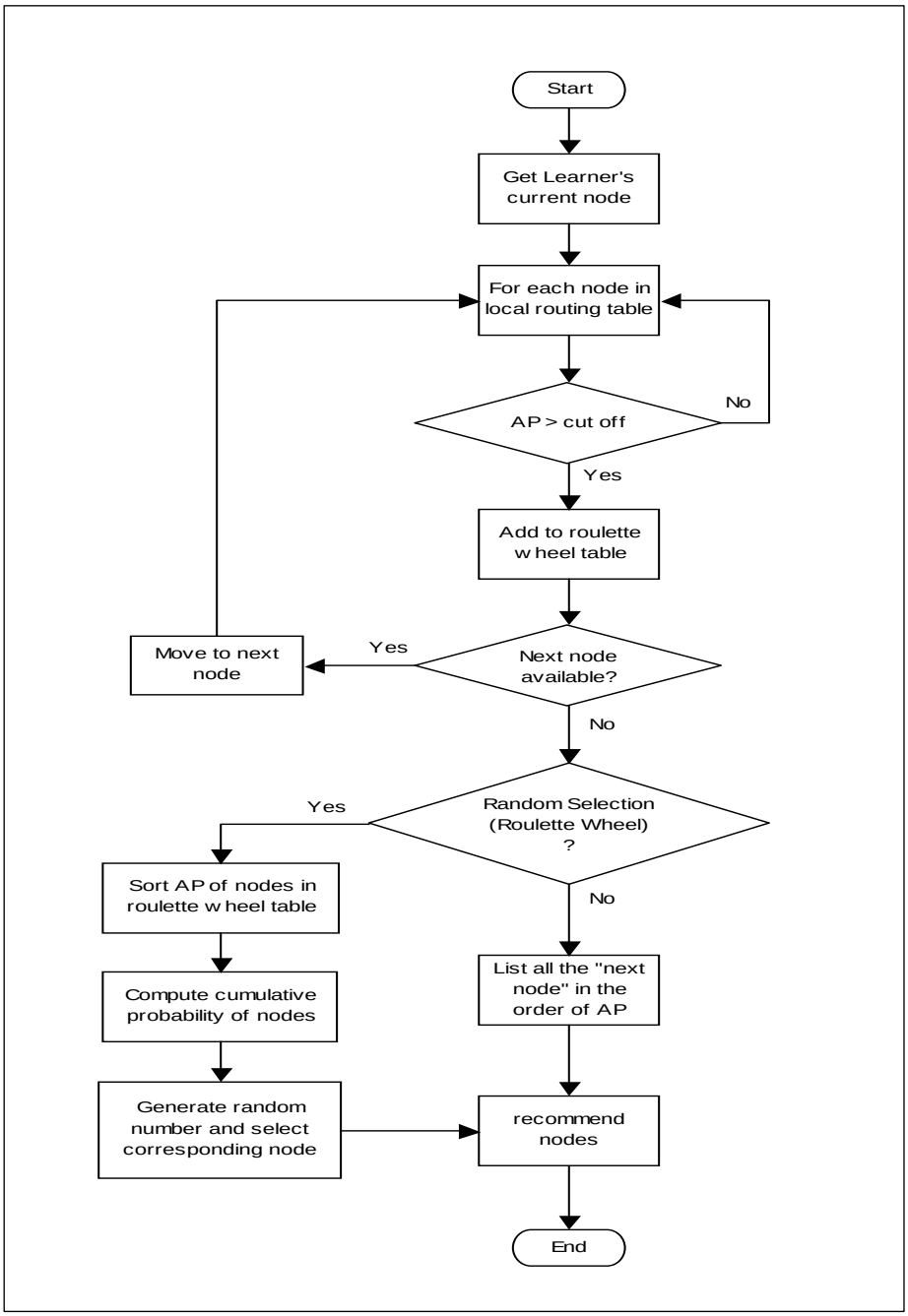

Figure 3: Control diagram of the process of recommending the next nodes

The simulator consists of an inference engine, a controller, an inference engine, a database and an e-learning Portal simulator. Driven by the DYLPA algorithm, the inference engine computes pheromones and recommends learning paths. The event-driven controller provides the functionality of data feeding, alumni selection and parameters control (training size, threshold values, etc.). The course data and the learner performance data are fed into the database which will be retrieved by the inference engine for further computations. The controller also acts as a communication hub among the other modules.

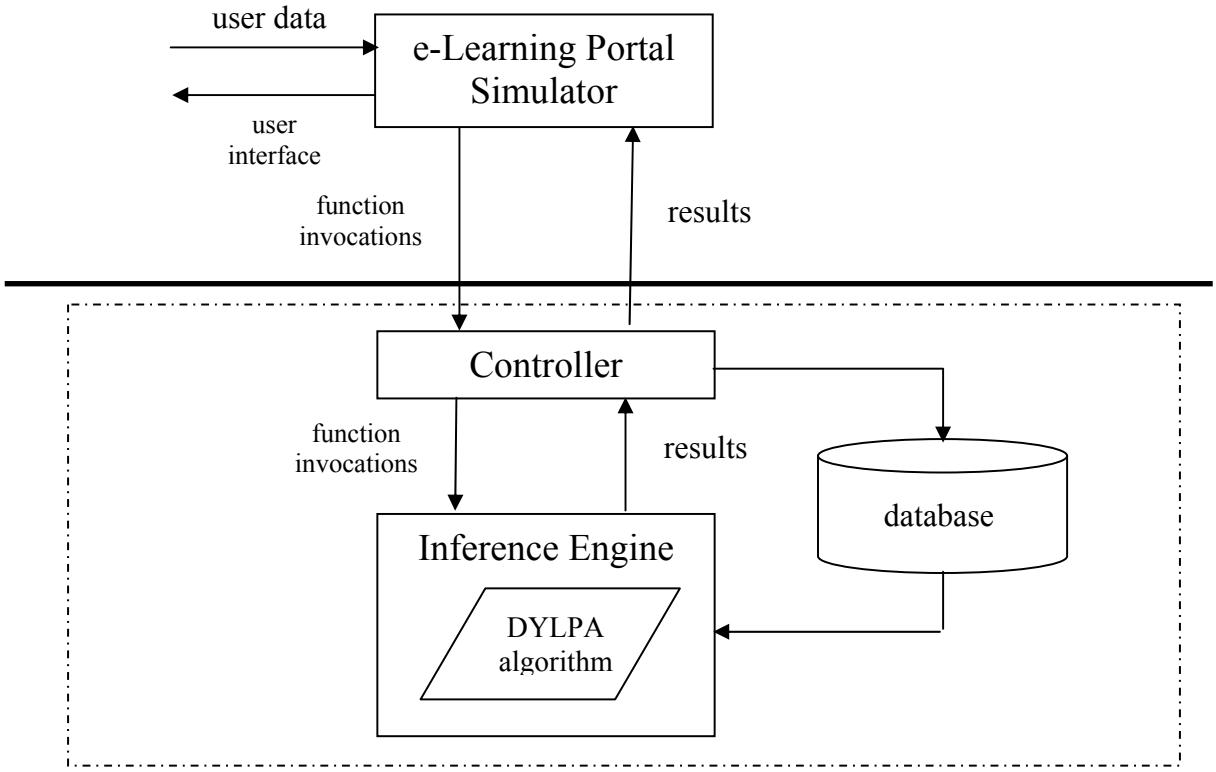

Figure 4: System architecture of DYLPA simulator

The e-learning portal simulator was designed with the purpose of simulating the integration between the e-learning system and the DYLPA. The simulator only feeds the test data into the inference engine, collects new learners' preferences and displays the recommended learning pathway or the next learning node. The test data consists of the course structure and alumni's performance records. It accepts the user input and would be redirected to DYPLA for recommending the best pathway.

Currently, we have validated and verified the prototype with simulated data. There are two different categories of test cases: (1) where number of "similar" (to the target learner) alumni << preset DYLPA training size; (2) where number of "similar" alumni > DYLPA training size. As an illustration, consider the following test cases that are subjected to the prescriptive and inductive pathways below,

- Prescriptive Pathway – learning object IDs (nodes): 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 10 → 11 → 12
- Inductive Pathway (what most "similar" simulated alumni went through, inductively, with minor variations among individuals, e.g., some might have skipped node 5 while some others might have visited node 8): 1 → 2 → 5 → 6 → 12

Test Case A (where number of "similar" alumni << training size)

- Training Size = 500
- Training threshold = 0.4
- Test Set = 300 "similar" alumni + 200 "random" alumni (whose background and histories are randomly generated – some of which may be "similar" to present student)
- Eventual recommended pathway : 1 → 2 → 3 → 4

When the DYLPA unit is activated, 303 alumni records were selected while the prescriptive agent carries a weight of 500-303=197. We would expect that the eventual recommended pathway is the same or similar to the prescriptive path due to the relatively heavy weight of the latter. Instead, the recommended pathway is terminated at node 4

because very few alumni have visited node 4 based on the simulated learner history (as reflected on the inductive pathway).

In real-life, when there is enough training size, such a "broken link" problem is very unlikely to occur. However, as it is a potential issue to DYLPA, we modified the algorithm by allowing the "routing" to proceed to the next node (in this case, from node 3 to, say, node 5) with much lower AP value as long as the student has yet to reach its learning goal (in this case, it is node 12). After the modification, the recommended pathway becomes:
1 → 2 → 3 → 5 → 6 → 7 → 9 → 10 → 11 → 12

where nodes 4 and 8 are skipped due to low visit rates, according to the simulated data. The new recommended path shows the predominant influence of the prescriptive path while the alumni's historical data still play a small part in varying it.

Test Case B (where number of "similar" alumni > training size)
- Training Size = 300
- Training threshold = 0.02
- Test Set = 300 "similar" alumni + 200 "random" alumni
- Actual recommended pathway: 1 → 2 → 5 → 6 → 12 → 14

In this case, with sufficient test set, the result can be predicted accurately as all the selected alumni are having the same background as the user. The random set does not cause obvious effect to the generated pathway because the Alumni Selector will select only the top 300. Therefore, the actual recommended pathway is identical to the "inductive pathway" that we preset for the simulator to generate test data. However, there is still 10% of test set which went through the course network in a random manner. Some simulated alumni have visited some other nodes and yielded "good" results. That explains why node 14 comes into the picture.

In summary, in type (1) cases, the DYLPA algorithm tends to recommend pathways that tally with the prescriptive pathways as the pheromones deposited by the prescriptive agent carries a greater weight. The greater the number of "similar" alumni increases (type (2) cases), the more the recommended pathways reflect the aggregate choices and performances of the alumni. More details of the validation and verification are reported in Lai, Lee, Chong & Ong (2004).

Related Work

In this section, we compare and contrast four other published similar ACO-based techniques with the DYLPA mechanism.

Gutiérrez *et al.* (2006) proposed a similar ACO-based approach. In their approach, sequencing graphs record the frequencies and the performances ("successful" or "failed") of various paths that other learners have been through. The information is presented to the target learner every time she finishes a learning unit (a "node" in our context), so she could choose the next unit by referring to how *all* her peers performed (the numbers of successful and failed peers) in the same situation.

Van den Berg *et al.* (2005) developed a simplified ACO algorithm that only keeps the records of the paths selected by the learners who have "successfully completed the course", determined by a post-course 5-question multiple-choice quiz, by maintaining a transition matrix that records the number of learner transitions between individual pairs of nodes on such paths. The full path will then be recommended to the next learner with the roulette wheel mechanism. Janssen *et al.* (2007) evaluated the approach by analyzing and comparing the online learning activity logs of more than 800 students who used or did not use this recommendation mechanism. He concluded that the approach were more effective in improving the performances, but were not necessarily more efficient in terms of individual students' total time spent in studying the online course.

Valigiani *et al.*'s (2005) Paraschool system (see also: Semet, Lutton & Collet, 2003) makes use of another modified ACO algorithm. The system requires the course designers to assign weights to individual arcs within the course network to reflect the "goodness" of the learners following particular outgoing arcs upon completion of a learning

node. The weights will be adjusted by real learners' performances later. Unlike DYLPA who allows gray-level performance measures for pheromone computation, Paraschool deposits two types of pheromones, namely, "success pheromones" and "failure pheromones", on the arcs for future fitness function-based (a concept borrowed from Genetic Algorithms) computations and recommendations.

Utilizing the ACO technique as well, the Style-based Ant Colony System (SACS) (Wang *et al.*, 2008) categorizes alumni into four learning styles (visual, audio, text or kinesthetic). In recommending the next node, the pheromones deposited by the alumni who fall into the same category with the target learner are favored in the computations. The pheromone computation is not based on alumni's performances but merely the number of times the alumni have traveled.

We argue that the four approaches have different combinations of pitfalls that our DYLPA algorithm has addressed (see Table 4).

Table 4: Comparison of the features of five ACO-based learning path recommendation mechanisms

	Pheromone computation based on alumni's paths or performances?	**Pheromones are time decaying?**	**Computation based on whose data?**	**Combining prescriptive and induction planning?**
Gutiérrez *et al.* (2006)	paths	no	All alumni	no
Van den Berg *et al.*(2005)	paths	no	Only alumni who have "successfully" completed the course	no
Semet, Lutton & Collet (2003); Valigiani, *et al.*, (2005)	paths & performances	yes	All alumni	No; but course designers are required to annotate weights on each arc in the course network as "prescribed preferences"
Wang, Wang & Huang (2008)	paths	yes	Categorizing alumni according to their learning styles; only alumni who fall into the same category with the target learner are selected	no
Wong & Looi (2002)	paths & performances	yes	Computing the similarity level between the attributes of the target learner and individual alumni; only alumni who have high similarity levels with the target learners are selected	Yes

First of all, all four approaches do not incorporate prescriptive planning as our approach does. With the notion of social navigation in mind, one may argue that a good course sequencing technique could rely solely on past learners' chosen paths and performances in recommending new paths or "the next nodes". The Paraschool system takes one step closer by requiring manual assignment of weights to individual arcs, which become "prescriptive preferences" of the course designers – this is similar to Carchiolo *et al.*'s (2002) system as described in the Literature Review section of this paper. However, we believe that the "classic" prescriptive planning techniques, having been validated by researchers in the past, should still have their place in the recommendation process, particularly during the early stage of a new course when there are very little or no alumni data to facilitate stochastic computations. Our novel approach of combining prescriptive and inductive planning provides a plausible and feasible solution to this issue.

Secondly, in Gutiérrez *et al.,* Paraschool and AACS' approaches, the chosen learning paths and the performances of all the past and/or present learners, regardless of their potentially diversified "learner attributes", are incorporated

into the computation, that is, they treat supposedly heterogeneous "agents" as homogeneous ones. Van den Berg *et al.*.'s approach does choose learners for pheromone computations – but only choosing "successful" alumni but ignore other "learner attributes". Hence, their system computes the same set of alumni data for all new learners who enroll to a course at the same time. SACS simply divide alumni into four categories as their attempt to address the typical ACO techniques' basic requirement of involving homogeneous agents. Compared to SACS, our approach employs a more fine-grained quantitative method in selecting similar alumni with respect to each new learner.

The time decaying feature of the pheromones is yet another element that the first two approaches lack. We have discussed the rationales behind this feature in section 3, i.e., (a) more "similar" alumni when a growing set of historical data is accumulated; (b) possible updates of the curriculum. Both reasons would make early pheromones gradually becoming less and less accurate for predicting future learners' performances. Therefore, we argue that time decaying is a significant feature that any stochastic mechanism for social navigation should not ignore, especially for domains that is dynamic and may change over time.

Conclusion

Although inductive course planning is not a new idea in learning technology field, DYLPA could still be considered novel as it is the first course sequencing technique that explicitly combines prescriptive and inductive planning Furthermore, given the adaptive capabilities of DYLPA, this algorithm may be more competitive in stochastic time-varying domains, in which solutions must be adapted online to changing conditions, than in static problems. The online learning domain may be seen as a domain with changing conditions. These changes may be due to curriculum updates, advancement in online instructional and learning tools, changes in the typical profile of the learners, and so on. The ability of the system to adapt to changing conditions is further enhanced by the inclusion of a time-decaying feature that favors more recent alumni than older ones.

As learning technologies are one of the ICT fields that are subjected to the full vicissitudes arising from considering the human factor, our novel technique is naturally bound to face some limitations. For future work, we would like to recommend further investigations on:

1. Weighting the agents

All the ACO-inspired algorithms require homogeneous agents. For DYLPA, we address the problem of "heterogeneous learners" by selecting the most closely matched alumni with the current learner. All the selected "agents" carry equal weights in pheromone computation. However, since the selected set of agents is not totally "homogeneous", we recommend further refinement of the algorithm so that the agents could carry different weights that are proportional to their respective similarity levels.

2. Improving the computation of the degree of similarity (including the heuristics)

Our current method of computing the degree of similarity and the choice of learner attributes for the computation are relatively *ad-hoc* and need to be revised. One potential solution is to incorporate other Artificial Intelligence techniques like Genetic Algorithms to learn the weight of the learner attribute settings. Another possibility is to adopt relevant techniques developed for other e-learning systems, e.g., the algorithm developed by Wang, Tsai, Lee & Chiu (2007).

3. Time-decaying factor

DYLPA incorporates the time-decaying factor of pheromones indirectly by considering it as one of the learner attributes that is taken into account in determining the degree of similarity. The current version proposes the difference of the time both learners are enrolled in the e-course. We propose the exploration of alternative ways of measurement, e.g., the degree of changes of the curriculum between the points when both learners are/were enrolled.

4. More elegant handling of backward learning

We have introduced a quick fix to handle the potential backward learning behaviors of the learners. Further investigation on the nature of such behaviors (especially that there may be other personal factors or reasons behind this) may shed light on whether our present fix is valid and if not, and whether the algorithm could be refined to handle such situations properly.

Finally, we need to validate the algorithm itself by testing it on real e-learning system with real alumni history and real e-learners as users.

In the long run, however, we believe such an inductive mechanism, guided by the past learners' aggregate performances, has the potential to become a formal validation technique of all prescriptive course sequencing and delivery planning techniques, which is similar to a technique that utilizes ACO for auditing of pedagogical planning as proposed by Guitérrez, Valigiani, Jamont, Collet, & Kloos (2007), or even for discovery of new prescriptive rules.

Acknowledgements

We would like to express our gratitude to Institute of Systems Science, National University of Singapore for their generous support of this study. Thanks also to Tong-Wah Lai, Patrick Lee, Conrad Chong and Pang-Yen Ong for their assistance in developing and testing the prototype.

References

Arshard, F.N. (1989). Knowledge based learning advisors, *Proceedings of International Conference on Technology and Education '89* (pp.213-217), Orlando, USA.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*, New York: Oxford University Press.

Brookfield, S. (1985). Self-directed learning: A critical review of research. In S. Brookfield (Ed.), *Self-Directed Learning: From Theory to Practice*, vol. 25, *New Directions for Continuing Education Series*. San Francisco, USA: Jossey-Bass.

Carchiolo, V., Longheu, A., & Malgeri, M. (2002). Adaptive formative paths in a web-based learning environment. *Educational Technology & Society*, *5*(4), 64-75.

Chen, C-M, Liu, C-Y, & Chang, M-H (2006). Personalized curriculum sequencing utilizing modified item response theory for web-based instruction. *Expert Systems with Applications*, *30*(2), 378-396, February 2006.

Claypool, M., Le, P., Waseda, M. & Brown, D. (2001). Implicit interest indicators. *Proceedings of ACM Intelligent User Interfaces Conference '01* (pp.33-40), Santa Fe, USA.

Di Caro, G., & Dorigo, M. (1998). AntNet: Distributed stigmergetic control for communications networks. *Art. Int. Res., 9*, 317-365.

Di Caro, G. (2004). Ant colony optimization and its application to adaptive routing in telecommunication networks. *Unpublished doctoral dissertation*. Faculte des Sciences Appliquees, Universite Libre de Bruxeles, Brussels, Belgium.

Dieberger, A., Höök, K., Svensson, M., & Lönnqvist, P. (2001). Social navigation research agenda. *Presented at: Computer-Human Interaction Conference '01*, Seattle, USA.

Dorigo, M. (1992). Ottimizzazione, apprendimento automatico, ed algoritmi basati su metafora naturale, *Unpublished doctoral dissertation*, Politecnico di Milano, Italy.

Dorigo, M., & Stützle, T. (2004). *Ant Colony Optimization*. Cambridge, MA: MIT Press.

Engelbrecht, A.P. (2007). *Computational Intelligence: An Introduction* (2nd Ed.), Hoboken, NJ: John Wiley & Sons.

Farzan, R., & Brusilovsky, P. (2005). Social navigation support in e-learning: What are the real footprints?. *Proceedings of Workshop on Intelligent Techniques for Web Personalization '05* (pp.49-56), Edinburgh, UK.

Gutiérrez, S., Pardo, A., & Kloos, C.D. (2006). Finding a learning path: toward a swarm intelligence approach. *Proceedings of International Conference on Web-based Education '06* (pp.94-99), Puerto Vallarta, Mexico.

Guitérrez, S., Valigiani, G., Jamont, Y., Collet, P., & Kloos, C.D. (2007). A swarm approach for automatic auditing of pedagogical planning. *Proceedings of International Conference on Advanced Learning Technology '07* (pp.136-138), Niigata, Japan.

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *Paper presented at International Conference on Management of Data '00*, Dallas, USA.

Höök, K., & Benyon, D. (2003). *Designing Information Spaces: The Social Navigation Approach*.London: Springer.

Huang, Y-M, Chen, J-N, & Cheng, S-C (2007). A method of cross-level frequent pattern mining for web-based instruction. *Educational Technology & Society*, *10*(3), 305-319.

Janssen, J., Tattersall, C., Waterink, W., Van den Berg, B., Van Es, R., Bolman, C., & Koper, R. (2007). Self-organizing navigational support in lifelong learning: How predecessors can lead the way. *Computers & Education*, 49, 781-793.

Joachims, T., Freitag, D., & Mitchell, T. (1997). WebWatcher: A tour guide for the World Wide Web. *Proceedings of International Joint Conferences on Artificial Intelligence '97*, (pp.770-775), Nagoya, Japan.

Karagiannidis, C., Sampson, D., & Cardinalli, F. (2001). Integrating adaptive educational content into different courses and curricula. *Educational Technology & Society*, *4*(3), 37-44.

Lai, T.W., Lee, P., Chong, C., & Ong, P.Y. (2004). Dynamic Learning Pathway Advisor (DYLPA) for Virtual Institute of ISS, *Unpublished internal report*, Institute of Systems Science, National Universiy of Singapore, Singapore.

Niemczyk, S. (2000). On pedagogical-aware navigation of educational media through virtual tutors. *Internal Document*, Department of Civil & Environmental Engineering, Massachusetts Inst. of Tech.

Parunak, H.V.D. (1997). "Go to the ant": Engineering principles from natural multi-agent systems. *Annals of Operations Research, 75*, 69-101.

Peachey, D.R., & McCalla, G.I. (1986). Using planning techniques in intelligent tutoring systems. *Man-Machine Studies*, *24*, 79-98.

Semet, Y., Lutton, E., & Collet, P. (2003). Ant colony optimization for e-learning: observing the emergence of pedagogic suggestions, *Proceedings of IEEE Swarm Intelligence Symposium '03* (pp.46-52), Indianapolis, USA.

Shipman, F., Furuta, R., Brenner, D., Chung, C., & Hsieh, H. (2000). Guided paths through web-based collections: Design, experiences, and adaptations. *American Society of Information Sciences*, *51*, 260-272.

SRI Consulting Business Intelligence (n.d.). *Welcome to SRIC-BI*. Retrieved March 14, 2008, from: http://future.sri.com/LOD/

Steels, L. (1991). Toward a theory of emergent functionality. *From Animal to Animats: Proceedings of the International Conference on the Simulation of Adaptive Behavior '91* (pp.451-461), Cambridge, MA: MIT Press.

Stützle, T., & Hoos, H. (1997). Improvements on the Ant System: Introducing MAX-MIN Ant System. *Proceedings of International Conference on Adaptive and Natural Computing Algorithms '97* (pp.245-249), Vienna, Austria: Springer-Verlag.

Tang, T., & Chan, K. (2002). Feature construction for student group forming based on their browsing behaviors in an educational system. *Proceedings of Pacific Rim International Conference on Aritificial Intelligence'02* (pp.512-521), Berlin/Heidelberg, Germany: Springer.

Valigiani, G., Biojout, R., Jamont, Y., Lutton, E., Bourgeois, C., & Collet, P. (2005). Experimenting with a real-size man-hill to optimize pedagogical paths. *Proceedings of the ACM Symposium on Applied computing '05* (pp.4-8), Santa Fe, USA.

Van den Berg, B., Van Es, R., Tattersall, C., Janssen, J., Manderveld, J., Brouns, F., Kurvers, H., & Koper, R. (2005). Swarm-based sequencing recommendations in e-learning. *Proceedings of International Conference on Intelligence Systems Design and Applications* '05 (pp.488-493), Wroclaw, Poland.

Wang, T-I, Tsai, K-H, Lee, M-C, & Chiu, T-K (2007). Personalized learning objects recommendation based on the semantic-aware discovery and the learner preference pattern. *Educational Technology & Society*, 10 (3), 84-105.

Wang, T-I, Wang, K-T, & Huang, Y-M (2008). Using a style-based ant colony system for adaptive learning. *Expert Systems with Applications*, *34*(4), 2449-2464.

Wedde, H.F., & Farooq, M. (2006). A comprehensive review of nature inspired routing algorithms for fixed telecommunication networks. *Systems Architecture*, *52*(8-9), 461-484.

Wong L.H., & Looi, C.K. (2002). Learning pathways & intelligent software agents. *Unpublished internal document*, Institute of Systems Science, National University of Singapore, Singapore.