DEFINING ADAPTIVE LEARNING PATHS FOR COMPETENCE-ORIENTED LEARNING

Jose Manuel Marquez
Telvent

Juan Antonio Ortega, Luis Gonzalez-Abril, Francisco Velasco
Universidad de Sevilla.

ABSTRACT

This paper presents a way to describe educational itineraries in a competence-oriented learning system in order to solve the problem of sequencing several independent courses. The main objective is to extract adaptive learning paths composed by the subset of needed courses passed in the right order. This approach improves the courses' re-usability allowing courses to be included in different itineraries, improving the re-usability of the courses, and making possible the definition of mechanisms to adapt the learning path to the learner's needs in execution time.

KEYWORDS

Ant Colony Optimization, adaptive learning path, curriculum, long life learning

1. INTRODUCTION

Nowadays, existing specifications do not take into account how to manage several courses in an ordered learning sequence. IMS-CP [1], the most widely used specification in the e-learning world, offers a mechanism of packaging educational in a .zip compressed file. IMS-CP is the corner stone of more complex specifications like SCORM [2] or Learning Design [3], but defining learning path composed by several packages is not of concern for any of them. These specifications are focused in the e-learning process involved in just one packaged course, but not in more complex situations that are presented in this paper. Some authors [4] have worked in content adaptability, giving good approaches for incorporating adaptive rules embedded in the SCORM course's descriptor. Sequencing of the course packages are usually carried out by web-based systems and adaptability mechanisms have to be taken into account in the server side. That is an easy solution when the learning path, defined as the ordered set of course packages that a learner has to pass, is fixed, but it requires a great effort in order to get a dynamically adaptive learning path. We think that the learning itinerary can be personalized based on the student's learning capacities. For this act, several learning speeds (associated to a score in a concrete course) and learning paths are considered which are optimized depending on the student's capacities. This learning system will allow a better attendance to the student's diversity.

Sometimes, a bunch of courses in the learning system have to be played in a fixed order when, just before starting one course, it is necessary to have achieved all the knowledge offered by a previous one. However, looking for a stronger learning personalization, the pedagogical team can design different courses to ensure the right knowledge acquisition by the student, in his own way, taking into account his/her personal needs: language, capacity, complexity, comprehension level, etc. In this way, the pedagogical team will be able to define several personalized paths to the same competence or goal. This is the first conceptual problem that needs to be solved: it is necessary to differentiate between learning itinerary or curriculum and all its learning paths that a student, in his/her own way, may choose to achieve all the competences and knowledge offered by the learning itinerary.

2. SCENARIO

Let us suppose that an enterprise wants to prepare workers to employ them in some different vacancies of higher responsibility. These vacancies require a set of competences being owned by the applicants, and the way to achieve these competences is to pass some specific courses.

The proposed platform is developed as an e-learning system built on top of an OSGi framework in which the educational contents, packaged as the IMS-CP specification defines, are delivered as OSGi bundles. Both specifications are 100% compatible [5] and it is possible to build a graph describing the learning itinerary needed to achieve one concrete competence. The achievement of this competence implies that a set of courses have been successfully passed, each one delivered as one atomic bundle. However, this package is not unique: courses and the pedagogical complexity of each one may vary from one user to another, depending on the user's learning profile. The enterprise may wish that courses vary dynamically reacting to the learning process, making the total length of the learning itinerary as long as needed by each user helping them to achieve its knowledge at their own capacity, defining special paths for students of various learning methods and styles.

The smart LMS could recommend or impose the next course, taking into account the most recent results of each user in his/her passed courses and the most successful path taken by most of the users.

With this goal, the company's pedagogical team has designed the following graph with several alternative learning paths (Figure 1) in which all the available learning paths a learner may follow is given. Therefore, nodes represent courses and arcs are transitions between courses. Arcs rounded with a curved arrow labeled with an ampersand (&) indicate that their children have to be crossed in the order expressed by the arrow (e.g. deep-first and from left to right) and arcs rounded with a curved arrow labeled with a plus sign means that one of the children paths has to be chosen. Each node contains educational contents and exercises, usually embedded in web pages, and an evaluation test, which will send the final qualification of the student to the LMS.

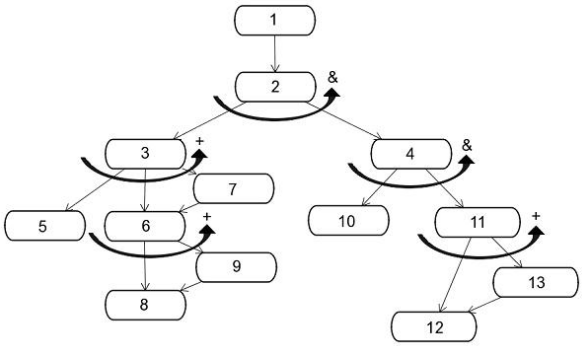

Figure 1. Sequencing graph with alternatives learning paths

The objective is to describe in a formal way this learning graph in order to be able of predicting the best learning path for each student, taking into account the user's profile and the paths followed by the rest of the students.

3. FORMAL DESCRIPTION USING XML

The learning graph shown in Figure 1 looks like a kind of tree, but it is not a real tree:

-Its starting point (root) and its end points (each leaf) cannot be the same node, but internal cycles are allowed.

-It contains implicit arcs that has not been drawn for clarity purposes. E. g.: After having completed the fifth course, the learner will start the course labeled with 4. The same thing occurs when a learner finishes the eighth course.

-The learning graph also introduce us the concept of conditional arc. The resultant path of taking a conditional arc is an adaptive decision that the smart LMS has to choose based on the recent past of the involved student and the actions done by the rest of equals (students using the same learning graph).

A declarative description of the Figure 1 using XML is:

```
<?xml version="1.0" encoding="UTF-8"?>
<learning-graph>
 <node-list>
  <node id="1" root="true">
    <name>Introduction to graphs</name>
    <url>http://myserver.com/course?id="1"
    </url>
  </node>
  <node id="2">
    <name>Serializing graphs using XML
    </name>
    <url>http://myserver.com/course?id="1"
    </url>
  </node>
 </node-list>

...

<arcs-list>
  <arcs from="1">
   <condition type="FIXED">
    <arc to="2" weight="1" order="1"/>
   <condition>
  </arcs>
  <arcs from="2">
   <condition type="AND">
    <arc to="3" weight="1" order="1" />
    <arc to="4" weight="1" order="2" />
   <condition>
  </arcs>
  <arcs from="3">
   <condition type="OR">
    <arc to="5" weight="1" order="1" />
    <arc to="6" weight="1" order="1" />
    <arc to="7" weight="1" order="1" />
   <condition>
  </arcs>
  <arcs from="4">
   <condition type="AND">
    <arc to="10" weight="1" order="1"/>
    <arc to="11" weight="1" order="1"/>
   </condition>
  </arcs>
  <arcs from="6">
   <condition type="OR">
    <arc to="8" weight="1" order="1"/>
    <arc to="9" weight="1" order="1"/>
   </condition>
  </arcs>

 ...

 </arcs-list>
</learning-graph>
```

The XML descriptor shown describes the learning graph like a list of nodes (tag node-list) and a list of arcs (tag arcs-list). The node-list contains the description of each node: id, name, url where the course package can be obtained, etc. By the other hand, the arc-list describes the possible paths that a learner may take depending of the outgoing arc assigned by the system. Note that the arcs of the list are conditioned,

using three types of conditions: AND, OR and FIXED (without conditions). Each arc also indicates its pedagogical weight and its position in the sequence with the rest of arcs of its same level. The order attribute makes sense only in case of AND condition. For simplicity purposes, other serialized fields related with the graphical representation like position, objects' class of the swing components that render nodes and arcs have been omitted.

This is a descriptive way of defining the learning graph in a human readable form and will help the system to initialize the data structures needed to store the current state of each user. However, it is already quite complex to be processed by a simple algorithm.

4. CREATING THE LEARNING TREE

In order to process the learning graph in a easier way by a simpler algorithm, it may be transformed in a pure equivalent tree (Figure 2).

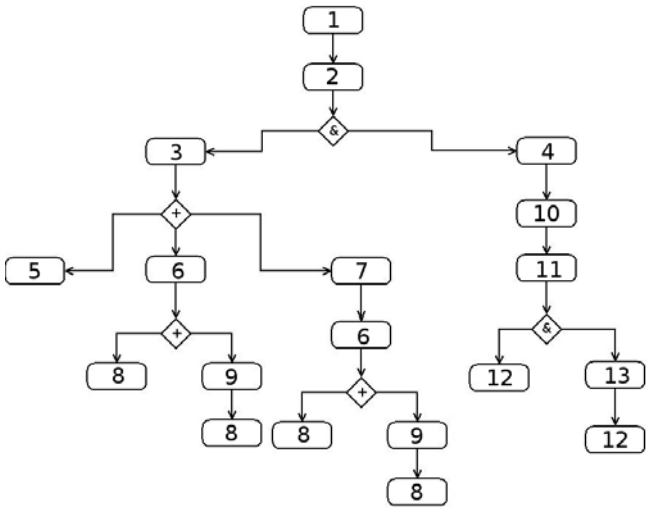

Figure 2. Learning Tree

The learning tree eliminates local cycles creating all the available learning paths. Note that, if it is necessary, some nodes are repeated in different branches.

The serialization process of the learning tree is based on the learning path concept: a learning path is composed by a root node following by one or more conditional paths. Obviously, the minimal path is just one node (a leaf). Consecutive branches will be serialized in order, indicating the order attribute when AND condition occurs. Applying a deep-traversal crossing algorithm the learning tree is serialized as follows:

```
<?xml version="1.0" enconding="UTF-8"?>
<learning-tree>
<node-list>
 <node id="1">
  <name>Introduction to graphs</name>
  <url>http://myserver.com/course?id="1"   </url>
 </node>
 <node id="2">
  <name>Serializing graphs using XML</name>
   <url>http://myserver.com/course?id="1"
   </url>
  </node>
  ...
 </node-list>

 <path root="1">
  <condition type="FIXED">
```

```
   <path root="2">
    <condition type="AND">
     <path root="3"  order="1">
      <condition type="OR">
        <path root="5" />
        <path root="6">
         <condition type="OR">
          <path root="8" />
          <path root="9">
           <condition type="FIXED">
            <path root="8" />
           </condition>
          </path>
         </condition>
        </path>
        <path root="7">
        <path root="6">
         <condition type="OR">
           <path root="8" />
           <path root="9">
            <condition type="FIXED">
              <path root="8" />
            </condition>
           </path>
          </condition>
         </path>
        </path>
       </condition>
      </path>
      <path root="4" order="2">
       <condition type="AND">
        <path root="10" order="1"/>
        <path root="11" order="2">
         <condition type="OR">
          <path root="12" />
          <path root="13">
           <condition type="FIXED">
            <path root="12" />
           </condition>
          </path>
         </condition>
        </path>
       </condition>
      </path>
     </condition>
    </path>
  </condition>
 </path>
</learning-tree>
```

Note that learning paths are defined recursively. The result is a longer XML serialization. On the other hand, a simpler algorithm for processing the complete learning tree and its internal learning paths is needed, making possible to apply adaptive mechanism to it.

5. PROVIDING ADAPTABILITY TO THE LEARNING PROCESS

The smart LMS will choose the next course taking into account the most recent results of the user in his/her passed courses and the most successful path taken by most of the users. This artificial intelligence based on the collective behavior of decentralized, self-organized systems has been named “swarm intelligence”, an expression introduced in the context of cellular robotic systems [6] by Gerardo Beni and Jing Wang, inspired in the social behavior of animals [7].

In order to solve the problem of calculating the best path for each user, taking the feedback from the rest of the users, we will implement a service based on Ant Colony Optimization [8]. In our system, a student represents an ant, moving freely in the learning tree, dropping pheromones everywhere it goes. Besides, the amount of dropped pheromones is modified as a function of the achieved success.

Let us consider as acceptable a qualification higher than *m* over 10, with 0<*m*<10, and to define the amount of pheromone to be dropped in the path of the late evaluated node as

$$\Phi_{ij}(\varepsilon, m) = \begin{cases} -(1-\varepsilon)\varphi - \omega_3 \dfrac{m-\varepsilon}{10}, if\ \varepsilon < \dfrac{m}{10} \\ \varepsilon\varphi + \omega_3 \dfrac{\varepsilon - m}{10}, if\ \varepsilon \geq \dfrac{m}{10} \end{cases} \quad (1)$$

where φ represents a constant of the pheromone value and ε is the score obtained by the learner in the late evaluated node, with $0 \leq \varepsilon \leq 1$. In order to empower the effect of the recently dropped pheromones, the last three arcs taken by the student in the learning tree, are reinforced as well with φ/2, φ/3 and φ/4 as it is done in [9]. ω_3 is a constant which will help to calibrate the system, letting us check the impact of the minimum required score in the amount of pheromones dropped. Therefore, the amount of pheromones dropped in a arc depends on the qualification of any student who has been taken the arc. Obviously, if no student have taken the arc, no pheromones will be thrown.

As it has been shown before while explaining the serialization of the learning graph, the pedagogical team may assign a pedagogical weight to each arc. This weight may represent the pedagogical complexity associated to the arc *ij* or the pedagogical load of each one compared with the total load of the whole path [10]. They are static values very useful to define a fitness function which gives us the value of how well the arc *ij* fits to the user needs:

$$f_{ij} = \omega_1 \cdot W_{ij} \cdot P_{ij} + \omega_2 \cdot \Phi_{ij} \quad (2)$$

This function will help to decide the next node and will help to build dynamically the learning path of each student. Therefore, it must involve both the pedagogical weight and pheromones.

In (2), W_{ij} denoted the pedagogical weight associated to the arc ij, Φ_{ij} is the amount of pheromones dropped in the arc ij, and ω_1 and ω_2 are constants to calibrate the system. P_{ij} is the suitability factor of the arc ij, calculated a priori based on the previous results, influencing the impact of the pedagogical weight. The first addend in (2) is completed with P_{ij} which is the suitability factor of the arc ij. This factor could be given by the pedagogical team as a static value or could be dynamically calculated as it is explained on [10], using bayesian networks. The first addend in (2) is key to avoid that the system is blocked in a local minimum caused by the effect of the second addend, the pheromones, just like it was tested in [11]. This would always lead the students by the same branch of the learning tree, commonly optimized as the way required by the minimum effort.

The fitness function may be dynamically calculated taking the score (ε), the minimum score desired (m) and the time[1] (t). In order to calculate it, we define the amount of pheromones in function of the time as:

$$\Phi_{ij}(\varepsilon, m)^t = \rho \Phi_{ij}^{t-1} + \lambda(i, j)\Phi_{ij}(\varepsilon, m) \quad (3)$$

In (3) ρ is the evaporation rate of the pheromones. In order to avoid a great impact in the fitness function, pheromones, like in real life, will partially evaporate with the passing of time. Because e-learning systems' users may interrupt their sessions whenever they want and they are able to continue the course from the same point they left it, the time has to be understood in the same way. So, the evaporation rate will be simply a constant less than 1 that the LMS will multiply by the current amount of pheromones existing in the edge each time a user chooses an edge (independently of the edge chosen). Initially no edge has pheromones. The value of the total amount of pheromones dropped in an edge is calculated once the student has finished the course and needs to choose the next course. A value of ρ = 0.9 is a typical evaporation rate [11], but in the

[1] Iteration counter

calibration process of the system any positive value less than 1 could be selected. In (3) the evaporation rate is multiplying the amount of pheromones existing in the arc in the time just before. The second addend of (3) is composed of (1) by $\lambda(i,j)$ which returns 1 if the node i was the latest evaluated course of the student and node j is going to be next course, or zero in any other case. Then the amount of pheromones for each edge at any given time, with $\lambda(i,j) = 1$, is:

$$\Phi_{ij}^{t}(\varepsilon,m)=\left\{\begin{array}{l}\rho\Phi_{ij}^{t-1}-(1-\varepsilon)\rho-\omega_3\dfrac{m-\varepsilon}{10},if\ \varepsilon<\dfrac{m}{10}\\ \rho\Phi_{ij}^{t-1}+\varepsilon\rho+\omega_3\dfrac{\varepsilon-m}{10},if\ \varepsilon\geq\dfrac{m}{10}\end{array}\right\}= \tag{4}$$

$$\left\{\begin{array}{l}\Phi_1^x+\rho^t\left(\Phi_0-\Phi_1^x\right),if\ \varepsilon<\dfrac{m}{10}\\ \Phi_2^x+\rho^t\left(\Phi_0-\Phi_2^x\right),if\ \varepsilon\geq\dfrac{m}{10}\end{array}\right\}$$

where $\Phi_1^x=\dfrac{-(1-\varepsilon)\varphi-\omega_3\dfrac{m-\varepsilon}{10}}{1-\rho}$, $\Phi_2^x=\dfrac{\varepsilon\varphi+\omega_3\dfrac{m-\varepsilon}{10}}{1-\rho}$ and Φ_0 is the initial value. Note that if $|\rho|<1$, then $\underset{t\to\infty}{Lim}\Phi_{ij}^{t}=\left\{\begin{array}{l}\Phi_1^x,if\ \varepsilon<\dfrac{m}{10}\\ \Phi_2^x,if\ \varepsilon\geq\dfrac{m}{10}\end{array}\right\}$, that is, Φ_i^x is stable.

6. CONCLUSIONS AND FURTHER WORK

In this paper it has been shown how to describe a learning graph and how should be generated the equivalent learning tree. An approach for providing adaptability to the learning system using Ant Colony Optimization has been introduced as well.

This project was initially born to solve the problem of adapting OSGi bundles to user's profiles in digital TV residential gateways, and is currently evolving to become a complex e-learning platform using the architecture and middleware resulting from an European R&D project and continues currently in development.

This project has allowed us to solve several problems related with different phases involved in distributed systems: deployment of courses packaged like OSGi[2] bundles, integrating heterogeneous learning platforms using Web Services or mixing digital TV techniques to develop an educational platform based on interactive digital television in collaboration with the results acquired by previous R&D projects.

Next steps in our work are focused on dynamic learning paths and tools for making easier the definition of learning trees based on the formal serialization described in this work.

[2] www.osgi.org

REFERENCES

IMS Global Learning Consortium. *IMS Content Package. Final Specification v.1.2*, November 2005. http://www.imsglobal.org

Advanced Distributed Learning. *SCORM 2004 Specification*. http://www.adlnet.org

IMS Global Learning consortium. *IMS Learning Design. Final Specification v.1.0*, January 2003. http://www.imsglobal.org

Rey-López, M.; Fernández-Vilas, A.; Díaz-Redondo, R. and Pazos-Arias, J., 2006. *Providing SCORM with adaptivity*. In Proceedings of the 15th International Conference on World Wide Web (Edinburgh, Scotland, May 23 - 26, 2006). WWW '06. ACM Press, New York, NY, 981-982

Márquez, J.M., 2007. *Estado del arte del eLearning. Ideas para la definición de una plataforma universal*. Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla. Spain.

Beni, G. and Wang, J., 1989. *Swarm Intelligence in Cellular Robotic Systems*. In Proceedings of NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy, June 26–30.

Miller, P. 2007. *Swarm Theory*. National Geographic Magazine. July 2007

Dorigo, M., Di Caro, G. and Gambardella, L.M., 1999. *Ant Algorithms for Discrete Optimization*. Artificial Life, 5 (2): 137–172

Gutierrez, S.; Valigiani, G.; Jamont, Y.; Collet, P.; Delgado Kloos, C., 2007. *A Swarm Approach for Automatic Auditing of Pedagogical Planning*. In Proceedings of Seventh IEEE International Conference on Advanced Learning Technologies, 2007. ICALT 2007. 18-20 July 2007 Page(s):136 – 138.

Márquez, J. M.; Ortega, J.A.; González-Abril, L. and Velasco, F., 2008. *Creating adaptive learning paths using Ant colony Optimization and Bayesian Networks*. International Joint Conference on Neural Networks (IJCNN). IEEE World Congress on Computational Intelligence. Hong Kong, June 1-6, 2008. (*Accepted paper, pending of being published in the proceedings*).

Gutiérrez, S., 2007. *PhD Thesis: Secuenciamiento de actividades educativas orientado a la reutilización y la auto-organización en tutoría inteligente*. Departamento de Ingeniería Telemática. Universidad Carlos III de Madrid.