Learning from a network of peers via peer-driven adjustment of a corpus

John Champaign and Robin Cohen

School of Comp. Sci.; Univ. Waterloo; Waterloo, ON Canada N2L 3G1
{jchampai, rcohen}@uwaterloo.ca

Abstract. In this research, we explore the opportunity for a community of e-learners to network together in a Web 3.0 environment in order to improve the educational experience of each student. In particular, we outline a procedure for each student to be able to propose the creation of subdivisions of existing learning objects (text, video, etc.) of a predefined corpus, in order to adjust the set of learning objects from which subsequent students will learn. We provide an algorithm that specifies the recommended content sequencing of the newly-created corpus for each new student, using a probabilistic approach based on measures of similarity with previous students and on the success of previous learning. We demonstrate the value of this approach through simulations of student learning experiences. In short, we provide a peer-driven personalized learning experience for each student.

Keywords: Personalization and recommendations, Community modeling

1 Introduction

The paradigm of Social Semantic Web, Web 3.0, explores the big data challenges of dependencies between data and the application that created it and the challenge of filtering massive amounts of content to find relevant, personalized information for a particular user within a community of fellow users. This work details novel technologies and methodologies for creating and managing e-learning systems that address the questions of "how adaptation and personalization methodologies can improve Web 3.0 environments" and "what models, techniques, tools are the most adequate to support Web 3.0 users".

In our research, we are exploring how the previous learning experiences of other students with an online repository of learning objects (texts, videos, articles, etc.) can be used to leverage the learning of current students in an intelligent tutoring environment. Inspired by the collaborative filtering approach of recommender systems[1], we aim to first of all identify like-minded students as the basis of the social network that is assisting the student. With this pool of students, we then aim to recommend to students the objects that have offered the most significant benefit to those previous peers. In contrast with other researchers, we are interested in considering previous learning experiences that may have already ended, not restricting ourselves to the networking of students who are currently engaged in learning, for our peer-based approach.

Our work takes as a starting point McCalla's proposed ecological approach [2] for the design of peer-based intelligent tutoring systems and first of all introduces an algorithm for selecting appropriate content (learning objects) to present to a student, based on previous learning experiences of peers. From here, our primary focus is on the process of enabling peers to augment the corpus of learning objects, proposing subdivisions of existing objects as valuable for guiding the learning of subsequent students.

1.1 Our Approach

Our proposed Collaborative Learning Algorithm (CLA) for determining which learning objects to present to students is presented in Algorithm 1. The anticipated benefit of a specific learning object l, for the active user, a, under consideration would be (Adapted from [1]):

$$p[a,l] = \kappa \sum_{j=1}^{n} w(a,j)v(j,l) \tag{1}$$

where $v[j,l]$ is determined by assessing the student before and after the interaction, and the difference in knowledge is the benefit, $w(a,j)$ reflects the similarity $\in$ (0,1], based on a comparison of assessments of each student's knowledge (with peer experiences reflected in the object's history), between each user j and the active user, a, and κ is a normalizing factor.

Algorithm 1 CLA

```
Input current-student-assessment (CSA)
for each learning object (LO): do
  Initialize currentBenefit to zero
  Initialize sumOfBenefits to zero
  Input interactions for students and LO
  for all prior interactions on LO: do
    {interactionInitialAssessment IIA}
    similarity = calcSim(CSA, IIA)
    {interaction-final-assessment IFA}
    benefit = calcBen(IIA,IFA)
    {sumOfBenefits SOB}
    SOB = SOB + sim * ben
  end for
  currBen = SOB / numOfPrevInter
  if bestObject.ben < currentBen then
    bestObject = currentObject
  end if
end for
if bestObject.ben < 0 then
  bestObject = randomObject
end if
```

Algorithm 2 Expanding Corpus

```
Input: Repository of LO, set of students
for each time unit of instruction do
  for each student (S) do
    if student is available then
      if not S's first LO assignment then
        do post-test assessment of S
        attach interaction with S to LO
        update student similarities
        {based on new assessment}
        {allow S to divide the LO}
        if S creates a new LO then
          generate LO
          {based on S and original LO}
          add new LO to repository
        end if
      end if
      assign S to a LO{using CLA}
    end if
  end for
end for
Return: Repository of LOs
{includes old repository plus new objects}
```

2 Corpus Algorithm

Our proposed approach [3] to maintaining a corpus of learning objects is to provide students with tools to (optionally) divide non-atomic lessons, turning a single learning object into multiple learning objects. For example, if a student were assigned a book as a learning object, after reviewing it she might feel that only 3 chapters were worthwhile for what she was trying to learn. Extracting these from the book as a whole allows subsequent students to benefit from the original student's experience and quickly targets the part of the original learning object that was most worthwhile.

Since as each student experiences a learning object new objects may be created, these need to become new learning objects in the repository. The initial interaction history for each new learning object is assigned the interaction of the original learning object. The motivation for doing this is to enable the new learning object to be as likely as the parent to be recommended to new students; this serves to prime the new learning object for its place in the repository[1]. The newly created learning objects are then available to be assigned to students using the ITS.

The algorithm that creates a new learning object, based on student selection of the most worthwhile parts, operates as follows. Once the student has selected a portion, a new learning object is created using only those parts. As mentioned, all data about previous interactions between students and the parent learning object are copied to the newly created learning object. Once new learning objects are introduced, the corpus grows to include divided objects as possible candidates for the *bestObject* selected for a student as in the CLA. The procedure to assign a learning object to a student is outlined in Algorithm 2. Over time, each student will either be currently interacting with a learning object or available to experience a new learning object. This is reflected in the pseudo-code description of Algorithm 2.

Note that we choose to allow repeated viewing of learning objects by students. This is consistent with the theories of psychologists such as Pimsleur [4] and is reinforced by the "Blue's Clues" case of school children viewing repeated lessons mentioned in [5]. Extensions to this work could certainly be made contrasting this approach to allowing each learning object to only be experienced once by a particular student. This would result in more complex reasoning about showing divided learning objects and their parents.

3 Experimental Method

In order to simulate the learning achieved by students, the following overall approach was used. Let *LOK[l,k]* represent some learning object l's target in-

[1] The introduction of a new learning object part way through the course of instruction creates a cold start problem. Inheriting the interaction history is, necessarily, an imperfect approach, as the entire point of creating a new learning object is for it to be different from the original in some way. However, the perspective was taken that this inherited history would be better than no history.

struction level of knowledge k, such that $LOK[l,k] \in [0,1]$. Here 1 represents the highest levels of knowledge a student can have, and 0 represents the lowest. For example, the target instruction level might be 0.68 for a 90 minute lab on recursion, since students have completed previous learning but are still gaining an understanding of nuances. Each object is assumed to be comprise of a small fixed set of different knowledge K (where $k \in K$).

Learning objects also have an impact, which can be positive or negative[2]. Let $I[l,k] \in \mathcal{R}$, represent the impact of learning from learning object l on the knowledge k, that is, in the optimal case how much the learning object can adjust a student's knowledge k. The impact can be thought of as, for a student at the target level, what is the expected learning benefit of the object. This is information used by our approach to simulate the learning that is occurring.

Let $UK[j,k]$ represent user j's knowledge of $k \in K$, such that $UK[j,k] \in [0,1]$. An example from computer science would be a knowledge of recursion recorded to be at 0.33. This would be interpreted as the student has an understanding of 33% of the course content dealing with recursion.

After an interaction with an object l, a user j's knowledge of k is changed by:

$$\Delta UK[j,k] = \frac{I[l,k]}{1 + (UK[j,k] - LOK[l,k])^2} \tag{2}$$

This has the implication that the impact of a lesson is at a maximum when the student's knowledge level matches the target level of the learning object. As the two values differ, the impact of the lesson exponentially decreases.

Based on this change, the user's knowledge in that area is updated as:

$$UK'[j,k] = UK[j,k] + \Delta UK[j,k] \tag{3}$$

The user's average knowledge can then be calculated as:

$$\overline{UK}[j] = \frac{1}{|K|} \sum_{k \in K} UK[j,k] \tag{4}$$

In order to plot learning curves, the average knowledge ($\in [0,1]$) of all students is plotted against their progress in the course of study. Algorithms perform well when the average knowledge attained by students is high.

3.1 Experimental Design Decisions

In our results, the x-axis represents the units of time that have passed so far for the instruction and the y-axis is the mean knowledge of the class as a whole, ranging from 0 (complete ignorance) to 1 (complete mastery), where an individual student's knowledge is calculated as the average over all $k \in K$.[3]

[2] The negative impact was introduced to simulate the possibility of misinformation from a poor quality learning object or a learning object that does a good job teaching one concept, while undermining the understanding of another concept.

[3] In this simulation we used 6 k values.

We incorporate reasoning about the length of time it takes to complete an interaction with a learning object. Clearly, in real learning situations, learning events can take variable amounts of time. Watching a recording of a lecture might take 76 minutes, while attending a day long seminar might take 8 hours. Rather than making the simplifying assumption that each interaction with a learning object will take an equivalent length of time, we can incorporate this concept into our reasoning.

Calculate Benefit (calcBen) of Algorithm 1 then needs to incorporate time. Rather than consider the absolute benefit of the learning object, we can think of the proportionate benefit, that is, how much benefit it provides per minute of instruction (assuming a repository where each learning object's average time to completion is recorded). This can be calculated by dividing the benefit of the learning object by the new length of time it takes to complete the interaction for the average student.

For example, if one learning object, LO_{37}, took 100 minutes to raise a student from a B to a B+ and another learning object on the same subject, LO_{42}, took 20 minutes to raise a student from a B to a B+, we could say that LO_{42} provides 5 times the benefit of LO_{37} (even though, in an absolute sense, they raised the student's grade the same amount).

Note that when a peer-generated division creates a new learning object, the divided object begins with an interaction history inherited from its (longer) parent; the predicted benefit of the parent object is thus also carried over (and there is no adjustment in the calculation due to the divided object's shorter length). Once new students begin to interact with the divided object, the benefit calculation then does include division by the length of object.

Algorithm 3 Function divide learning object simulation

```
Input: Learning object lo, student s
{After student s completes interaction with learning object lo}
if lo.generation < 3 then {Allow a max of 3 divisions}
  if Random.prob(divide-value) < 0.20 then {20% chance a new division is proposed}
    newLO = lo
    newLO.generation = lo.generation + 1
    newLO.length = lo.length / 2 {division always halves an object}
    for each lo.impact do
      newLO.impact[i] = lo.impact[i] / 2 {inherits half the impact of the parent}
      if Random.prob(author-value) < s.authorship then
        newLO.impact[i] = newLO.impact[i] * 1.1 {this models improved impact}
      else
        newLO.impact[i] = newLO.impact[i] / 1.1 {this models decreased impact}
      end if
    end for
    {achieve personalization}
    for each lo.targetLevel do
      diff = (s.knowledgeLevel[i] - lo.targetLevel[i]
      newLO.targetLevel[i] = lo.targetLevel[i] + 0.1 × diff
    end for
    for each lo.interaction do
      newLO.interaction[i] = lo.interaction[i]
    end for
  end if
end if
Return: newly created learning object, newLO
```

In our simulation (detailed in Algorithm 3) a new lesson is created 20% of the time when a student interacts with a learning object, representing a group of students who are highly motivated to contribute to the system. After division, the length of the newly created learning object is set to half of the original learning object [4]. The impact of the newly created learning object was set to be proportionate to the new length, halved in this case[5]. This impact was then adjusted to be slightly higher or lower, depending on the skill of the student who proposed the division. The impact for each knowledge dimension, k, is increased or decreased by 10%. The likelihood of an increase (rather than a decrease) was set to be the *authorship* rating of the student who made the division. Student authorship ratings were randomly generated in the range [0,1] and can be thought of as the student's ability to discern higher and lower quality parts of learning objects.

The target level of instruction for each dimension of the learning object's knowledge was moved 10% closer to the current knowledge of the student making the division. We refer to this as *personalizing* the newly created learning object towards its author. This represents the idea that a student who is not as advanced as the target level of instruction of the learning object will tend to select the simpler ideas presented, while a student who is more advanced than the target level of instruction of the learning object will tend to select the advanced concepts.

For each experiment, identical learning objects and students were used for the various approaches. After the simulation, each student and learning object was reverted to its original state for the next run. This allows us to remove the variability from randomly differing groups of students and libraries of learning objects when comparing the effects of the various conditions. The interaction history of the original, undivided learning object was copied to the newly created learning object. After the division, each is tracked and considered independently. Ultimately, this algorithm recommends which learning object, including newly created learning objects, should be assigned to a particular student at a particular point in her course of study using Algorithm 2.

For the experiments 20 students interacted with 100 learning objects over 20 iterations and the averaged results are presented[6]. Students were simulated as having certain levels of knowledge as a result of their post-test assessments. To be realistic, we modelled some possible error with the assessment. This was achieved by considering a Gaussian distribution with a mean of 0 and a standard deviation of 0.1, yielding a number that would adjust the student's assessment value to be either somewhat higher or lower. Variable time length learning objects were used. A total time of instruction of 20,000 time units was used and each learning object ranged from 30 time units to 480 time units. The impact of each learning

[4] This simplification, that division is always halved, may be relaxed in future work and extensions made for arbitrary division of objects.

[5] This was done in order to model the fact that only part of the entire possible benefit from the learning object was experienced.

[6] This was done to reduce random noise in the results.

object ranged from -0.05:0.05 for a 30 time unit lesson, scaled proportionately by length of lesson[7] through to an impact of -0.8:0.8 for a 480 time unit lesson.

In order to plot learning curves, the average knowledge ($\in [0, 1]$) of all students is plotted against their progress in the course of study. Algorithms perform well when the average knowledge attained by students is high. A set of algorithms to select learning objects for students were run, to demonstrate the value of the proposed approach. **Random Association** associates each student with a randomly assigned learning object; **Greedy God** chooses the best possible interaction for each student for each trial[8]. These two curves are the benchmarks (low performance and "the ideal"). Three variations of Algorithm 1 were then run to set the LO in focus for the student. **Raw Ecological** has each student matched with the learning object best predicted to benefit her knowledge; **Pilot Group** has a subset of the students (10%) assigned, as a pilot group, systematically to learning objects - these interactions are used to reason about the best sequence for the remaining 90% of the students; **Simulated Annealing** provides a good approximation of the global maxima for a large search space. During the first 1/2 of the trials there is an inverse chance, based on the progress of the trails, that each student would be randomly associated with a lesson; otherwise, the ecological approach was applied. This approach provides a nice balance between exploration and exploitation.

4 Simulations

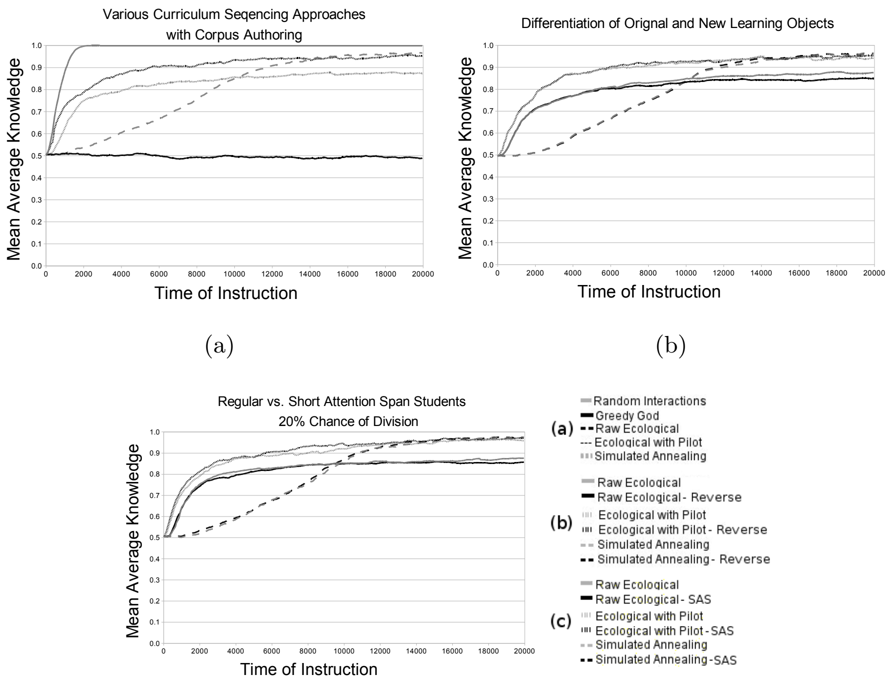
4.1 Divisions of Learning Objects

For this experiment we contrasted (i) divided learning objects, assigned to students using Algorithm 2 with each of the ecological, pilot or simulated annealing approaches with (ii) assigning the newly created learning object randomly or (iii) using the greedy god approach. In this experiment, students are able to propose divisions of learning objects and the repository of learning objects is extended to include divided learning objects as well as the original ones. As mentioned above, learning objects are divided 20% of the time when a student interacts with them, and at most a learning object can have three generations (that is, only itself and its children can be divided, its grandchildren can not). These results are displayed in Figure 1 (a).

Personalization and authorship were used for this simulation to provide a richer modeling of students and learning objects in the simulation. In other words, the impact of the learning object is increased when the division is performed by a student who is well skilled in authorship, leading to an increased chance that the divided learning object would be shown to a new student if

[7] For example, a 90 time unit lesson would have impact ranging from -0:15:0:15, three times a 30 time unit lesson.

[8] This is achieved by giving the algorithm full access to the fine-grained knowledge levels of the students and learning object, testing what the outcome would be for every possible interaction, then choosing the best interaction for each student for each trial.



(a) (b)

(c)

Fig. 1

the authorship levels of the dividing student were higher; in addition, the target level of the learning objects is brought closer to that of the dividing student to increase the chances of showing this object to like-minded students.

Results: These results put the efficacy of reasoning about divided learning objects in context between the baselines of the random (lower end) and greedy god (ideal case) approaches. The steep curve of the greedy god condition is due to the large number of learning objects, 500, and the newly created learning objects, all of which this approach can immediately capitalize on.

We see that with the divided learning objects, the strengths of the various approaches still hold. The simulated annealing under-performs initially, but delivers the best result by the end of the experiment and the pilot does best early on, at the cost of a poor experience for the pilot group who prime the system for their classmates. These results suggest that it is not harmful to allow peer-based authoring, even in situations with highly variable authorship quality.

4.2 Differentiation of Original and New Learning Objects

In previous work [6] our simulations considered a fixed library of learning objects, all of which are present at the beginning. Divided learning objects, therefore, presented a new challenges as they introduce new learning objects part way through the simulated students' course of study. This is an example of the cold-start problem: how to recommend a new item with no interaction history. In this work, we took the perspective that inheriting the history of interactions from the parent learning objects was the best approach to "prime" the newly created learning object. As the parent and child each has further interactions with students, they can be differentiated between and (potentially) each recommended to populations of student who would benefit from these interactions.

One challenge is that after creation and the attachment of these interactions, the parent and child learning objects will have identical interaction histories, and therefore will have equal predicted benefit for any particular student. Which to recommend in the case of this tie is then the question that presents itself. For this experiment, we contrasted yielding to the parent learning object versus yielding to the child learning object (which we refer to as Reverse). These results are displayed in Figure 1 (b).

Results: The similar results from giving priority to the original learning object or the newly created learning object suggests that either approach allows for the divided learning object to be differentiated from the original, and that a systemic bias towards one or the other isn't a large concern.

How the algorithm breaks ties and differentiates between the original and newly created learning objects does not seem to compromise the performance of our algorithm. This confirms that we are effectively learning which objects to present to students in order to achieve the best benefit, regardless of the initial assignments of objects to students.

4.3 Short Attention Span Students

We incorporated the idea of students with different learning styles by modeling students with short attention spans. We scaled the impact of learning objects for these students by increasing the shortest third (30 to 180 time units) of lessons by 25%, and decreasing the impact of the longest third (330 to 480 time units) of lessons by 25%. This was contrasted, for the raw ecological, ecological with pilot and simulated annealing curriculum sequencing approaches, with a group of students who were identical except for not having short attention spans (which we refer to as SAS). These results are shown in Figure 1 (c).

Results: The results for each of the various curriculum sequencing approaches show that our approach handles student populations with different learning needs (in this case length of lesson), making recommendations such that appropriate learning objects are shown. With the raw ecological approach, the short attention span group underperformed slightly, while in the ecological with pilot approach it outperforms and in the simulated annealing they are evenly matched.

The Pilot group out-performance can be explained by the more extensive interaction history provided by the pilot group. This makes it easier for the system to identify the superior performance from shorter lessons, and therefore preferentially recommend these from the start. The raw ecological and simulated

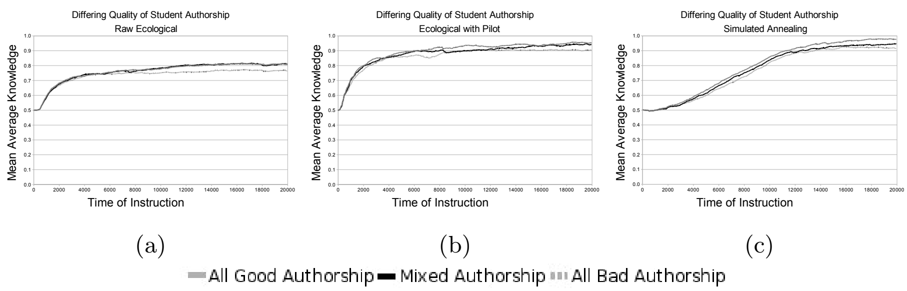


Fig. 2: Varying Authorship Ability in Students

annealing must gather data before there is sufficient history to systematically assign shorter learning objects to students.

It is important to point out that our approach was not tailored to students benefiting from or being penalized for varying lesson lengths. This was incorporated into the simulation of the interactions between students and learning objects, but our approach reasoned about interactions using only the pre- and post-assessments of each interaction. Using this data it was able to tailor the curriculum to the special needs of this group.

4.4 Varying Authorship Ability in Students

We contrasted different populations of quality of student authoring by allowing the student authorship characteristic to be all good (impact will always be raised), all bad (impact will always be lowered) and ok, i.e. evenly distributed between both (50% chance of being raised and 50% chance of being lowered). These results are shown in Figure 2.

Results: These 3 results show that, for each of the curriculum sequencing approaches, our technique was able to make useful recommendations to students. The similar results in the raw ecological and the ecological with pilot for both the good and the ok authorship demonstrates the techniques are very good at finding worthwhile authorship (an even divide of good and bad authoring is comparable to all-good authoring). The minimal under-performance of the all-bad authoring results provides support for the perspective that even in the case of very low authorship quality, a reasonable curriculum (based primarily on the original learning objects) will be delivered.

5 Discussion

McCalla's [2] work advocates the leveraging past interactions with students in order to determine how best to interact with a particular student in a personalized manner. We go beyond this proposal, however, to allow for growth of the initial repository. This is important due to pervasive challenges in assembling appropriate tutorial content, as discussed by researchers, such as [7, 8]. We therefore contrast with other researchers in peer-based intelligent tutoring (e.g. [9–11]) in allowing peers to (cautiously) play an additional role in the tutoring: that of content authors. As a novel direction, we are careful to avoid presenting peer-based objects which may possibly harm a student's learning (distinct from other peer-based tutoring researchers).

Other intelligent tutoring systems researchers have explored the value of simulating students. Van Lehn et al. ([12]) also specifically track and formally represent, for each student, their knowledge before learning, the behaviour during the learning, the instruction and the student's knowledge after learning. In contrast, we are interested in tracking behaviour with respect to learning objects, and focus on modeling the student's knowledge before and after interactions with those learning objects. They measured the accuracy of production rules, in terms of successfully matching a step in solving the problem, compared to number of training problems or frequency of learning opportunities. We follow a similar approach in the evaluation of our work, where we use the resulting learning curves [13] to contrast educational environments.

Judy Kay's work on Lifelong Learning Modeling [14] considers how to approach building a lifelong user model to support universal, personalized lifelong learning. Kay advocates giving learners control of their model (what goes in, what comes out and the ability to examine the model itself). She discusses stereotypes (e.g. [15]) and communities as methods of reasoning about learners. Learner modeling will certainly be an ongoing part of our work, and stereotypes and communities might be a sophisticated way to create simulated students who approach an intelligent tutoring system in different ways.

6 Conclusion

In this paper, we have presented a framework for peer-based intelligent tutoring that not only leverages the past experiences of peers using a repository of learning objects but that also allows peers to propose divisions of existing objects in order to introduce new objects into the corpus. Our algorithms for determining which learning objects to present to each student are personalized, drawing on the benefits enjoyed by like-minded students in the past. Our primary focus with this research has been to investigate the appropriate support for divided learning objects and for peer-based direction of the process of division. Through simulations of the learning achieved by simulated students, we are able to verify that the average knowledge levels achieved by students in the learning environment described above are on a par with those where no divisions are allowed. Moreover, even in cases where the peers may be poorly skilled in proposing effective divisions of objects, our framework ultimately presents to students objects that deliver appropriate value.

The personalization offered by our particular approach is further reinforced by simulations of populations of students who are more attuned to learning objects of shorter duration (and who therefore thrive best when continuously presented with shorter, divided objects). We further illustrate how our approach results in different objects for different students, with examples that map out how inappropriate objects are stigmatized for these particular students and how two students with certain distinguishing characteristics end up experiencing different objects from the repository. In all, we offer a promising direction for Web 3.0: allowing users to propose streamlined viewing of web objects with an algorithm that effectively suggests which content should then be shown to peers.

References

1. Breese, J.S., Heckerman, D., Kadie, C.M.: Empirical analysis of predictive algorithms for collaborative filtering. In Cooper, G.F., Moral, S., eds.: Proceedings of the 14^{th} Conference on Uncertainty in Artificial Intelligence. (1998) 43–52
2. McCalla, G.: The ecological approach to the design of e-learning environments: Purpose-based capture and use of information about learners. Journal of Interactive Media in Education: Special Issue on the Educational Semantic Web **7** (2004) 1–23
3. Champaign, J., Cohen, R.: Peer-based intelligent tutoring systems: A corpus-oriented approach. In: Tenth International Conference on Intelligent Tutoring Systems: Bridges to Learning, Pittsburgh (2010) 212–214
4. Pimsleur, P.: A memory schedule. The Modern Language Journal **51**(2) (1967) pp. 73–75
5. Gladwell, M.: The Tipping Point: How Little Things Can Make a Big Difference. Abacus (2002)
6. Champaign, J., Cohen, R.: Exploring the effects of errors in assessment and time requirements of learning objects in a peer-based intelligent tutoring system. In: 2011 FLAIRS Conference Proceedings, Palm Beach, Florida (2011)
7. Vassileva, J., McCalla, G., Greer, J.: Multi-agent multi-user modeling in i-help. User Modeling and User-Adapted Interaction **13** (2003) 179–210 10.1023/A:1024072706526.
8. Murray, T.: Authoring intelligent computer systems: An analysis of the state of the art. International Journal of Artifical Intelligence in Education **10** (1999) 98–129
9. Brusilovsky, P., Henze, N.: Open corpus adaptive educational hypermedia. In: The Adaptive Web. (2007) 671–696
10. Cheng, R., Vassileva, J.: Design and evaluation of an adaptive incentive mechanism for sustained educational online communities. User Model. User-Adapt. Interact. **16**(3-4) (2006) 321–348
11. Read, T., Barros, B., Bárcena, E., Pancorbo, J.: Coalescing individual and collaborative learning to model user linguistic competences. User Modeling and User-Adapted Interaction **16**(3-4) (2006) 349–376
12. VanLehn, K., Ohlsson, S., Nason, R.: Applications of simulated students: An exploration. Journal of Artificial Intelligence in Education **5** (1996) 135–175
13. Matsuda, N., Cohen, W.W., Sewall, J., Lacerda, G., Koedinger, K.R.: Evaluating a simulated student using real students data for training and testing. In: User Modeling. (2007) 107–116
14. Kay, J.: Lifelong learner modeling for lifelong personalized pervasive learning. TLT **1**(4) (2008) 215–228
15. Chin, D.N.: User modeling in UC, the UNIX consultant. In: Proceedings of the SIGCHI conference on Human factors in computing systems. CHI '86, New York, NY, USA, ACM (1986) 24–28