

Mobile Learning: Is Anytime + Anywhere = Always Online?

Anna Trifonova and Marco Ronchetti

University of Trento, via Sommarive 14, 38050, Povo (TN), Italy

Tel: +39 0461 88 2033; Fax: +39 0461 88 2093

{Anna.Trifonova; Marco.Ronchetti}@dit.unitn.it

Abstract

The new e-learning trend, called mobile learning opens variety of questions to solve. To allow access to learning content anytime and anyplace a technique called hoarding is sometimes indispensable for covering the periods of disconnection. In this paper we show the outcomes of the hoarding sub-system of Mobile ELDIT system, developed at the University of Trento. Our results demonstrate that in certain scenarios anytime, anywhere m-learning might be reached even without online access available.

1. Introduction

Mobile learning is a new and wide research domain [1]. The main advantage it gives to the learners and educators is the freedom to use what is needed where is needed. Small but powerful devices could allow access to needed content and services anytime and anywhere. This is often obtained by accessing the desired content via Internet, using different connection options available. There are really quite a lot of possible solutions – wireless LANs, WAP, GPRS, UMTS etc. But what happens when no connection is available? Does mobile learning mean always online? Our answer to this questions is ‘No. Anytime, anywhere might be achieved also when disconnected.’

In our previous work we discussed possible architecture to support mobile learning [2]. We argued that an indispensable part of such architecture should be the support for offline delivery of content. We called “hoarding” the process of selecting the learning materials for allowing access even during disconnected periods. In another previous paper [3] we discussed the possible strategy to solve the hoarding problem in the general case. In this paper we report the analysis of the results of the hoarding experiments we performed.

2. Architecture to support m-learning

Different research bodies have provided different architectural solutions to support m-learning. Often the choice depends on the concrete functionalities one would like to provide to the mobile users, but some approaches and modules will be common. In [2] we

discussed that most often m-learning would try to take the advantage of reusing possible services already provided by e-learning platform. Such idea is supported by many other researchers, like [4, 5, 6].

According to our proposition there are three distinct main functionalities that have to be provided in an m-learning platform – ‘Context Discovery’, ‘Content Management and Adaptation’ and ‘Support for Offline Access to Content’. Details about each module are described in [2], but here we present the outcomes from a real-world system developed according to this architecture and in particular from its ‘Support for Offline Access to Content’ module.

3. Mobile ELDIT

According to the above mentioned architecture and to the guidelines for successful m-learning available in literature, we developed a real system called Mobile ELDIT. It is based on an innovative e-learning system, designed for helping learners to prepare for the exam of bilingualism (Italian and German languages) needed in South Tyrol [7]. The mobile system sits on top of the e-learning one, using its raw material, i.e. the learning content. On the server side a special redesign engine takes care of converting the content into proper for the mobile device form. From the device the learner accesses the study material (i.e. set of texts with associated words and comprehension questions) through a standard web browser. Whenever the learner wants to see the translation, explanation, examples or additional information about a word in the text he can click on it. The learner actions are captured and recorded in log files by a local proxy that responds to the browser’s requests with locally pre-fetched content. The log files are used to analyze users’ behavior and to predict what material will be needed for the next offline learning session. The predicted chunk of data should be pre-fetched locally on the device.

The first experimental results of the mobile system are described in [8]. Here we present further results of the hoarding subsystem of Mobile ELDIT.

4. Hoarding in Mobile ELDIT

The hoarding idea [3] is that in absence of connection and when due to memory problems it is impossible to keep on the mobile device a full copy of all the available material, an automatic agent should cache in advance *the relevant portion* of the material itself. The main problem is how to identify such *relevant portion*. Of course, one needs to predict what the user will need in the next sessions: what will be the starting point, which material will s/he access starting from there etc. This implies the need to have information about user's style, preferences and knowledge.

In our experiment, to obtain the first hoarding results we observed one user at a time. We were giving to the user a short list of available texts and we considered this to be always the user's 'starting point', as described in the general algorithm [3]. Then, for creating the 'candidate set' we selected all the words that were accessible from the chosen text and we did pruning, based on what we believe the user already knows (based on his/her previous sessions). Our goal was mainly to test the automatic pruning which means essentially to discover automatically the user knowledge set. We used the following assumption: the user knows all the words that were presented to him/her in a previous text, and whose links were not followed. In this way, as same words were used in consecutive texts, on every next iteration more words were pruned and the hoarding became smaller (Figure 1). On the figure the abscissa is the step, which we chose to be one text; the blue dots in the lower part show the real user requests; the continuous line shows how the hoard decreases at every next text that the user was reading; the triangles show the miss rate (calculated as the percentage of accesses for which the cache was ineffective) if it is not zero.

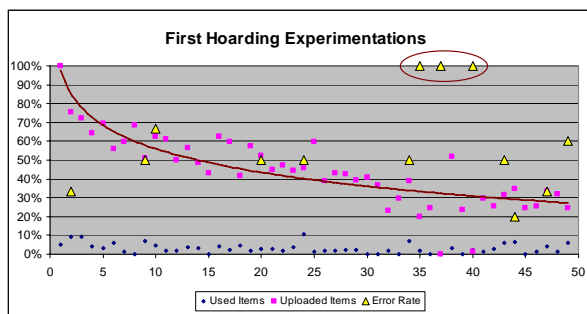


Figure 1: Simple hoarding

First of all, the graphic shows that hoarding process works, even in this rough first iteration where we used the simple pruning rule described above. One can see that within 50 steps the hoard size decreased to about 30%. As a consequence the hit rate (i.e. the number of

hits divided by the total number of uploaded predictions) grows from about 5% on the first step up to 25%. Even though the pruning rule was rather simple we see that in 75% of the cases we had no misses, which is quite good correctness. We decided however that we needed to refine the algorithm by improving it in two directions:

- To make the hoard decrease faster and
- To make the algorithm work more precisely.

To make the hoard decrease faster (i.e. in fewer steps and with bigger values) we should gather knowledge from usage data of other users. This means we have to analyze the similarity between users. For similar users (for example similar in their proficiency on the studied subject) we can guess that a user knows certain word, based on our awareness that the other similar user (or users in a group) knows it. In contrast with the first experiment, where we considered a word familiar for the learner only when he already had the possibility to see it, in a further trial we will try to guess in advance.

On the other hand one can see on the figure that in some cases we have a large (up to 100%!) miss rate. It has to be mentioned that for the m-learning scenario the accuracy is very important. This was proven by a questionnaire filled by our users, where almost everybody mentions hoarding misses as the most disturbing problem. A miss in the hoarding might lead to termination of the study process or even worse to misunderstanding of the material. Thus we have to assure that the algorithm works more precisely.

One of the main reasons for errors of the hoarding algorithm is the simplicity of the pruning rule that we used in the first experiment. In the few cases where we had a 100% miss rate, the reason was that the user had requested a text and without reading it, and then pressed the back button and continued with other material. This misled the algorithm to infer that the user knows all the words that were provided in the text. Those words were therefore removed from the hoarding set in the next interactions. When in the next sessions the user requested the same text again, this time really reading it, every requested word was missing, since our algorithm had already decided that all words are known to the user and excluded them. This problem can be solved by monitoring the time spent on a page, so as to be able to understand whether the page was actually read. We have measured the average time needed for reading a text (more than 3 minutes in Mobile ELDIT). If the time spent on a text is significantly less than this average, we now conclude that the user most probably did not really read it. The addition of time measurements in the hoarding algorithm as a guarantee for real review of the material was a simple step that helped excluding some of the misses – the ones surrounded on Figure 1.

Still however lots of misses continue to exist.

As a next step for improving the hoarding algorithm we decided to concentrate on minimizing the miss rate, as every miss in the hoard might be critical for the users' understanding of the studied material and thus the low miss rate is probably the most important factor for the hoarding process.

A particularity of ELDIT is that the clickable words are linked to their infinitive form, thus that even different forms of the same word are requested by the same URL. For example all derivation of the Italian word 'sentimento', like 'sentimentale' or 'sensazionale' are requested and thus saved in the log files, by the link "it.n.sentimento.l.derivati". In certain texts a word and its unusual form or conjugation might be very important for the understanding of the text. This means that certain words might be 'critical' for these texts, for users' understanding or for answering comprehension questions. We have decided to create a 'Critical Set' that will contain such 'critical' words and use it for improving the hoarding performance. Our supposition was that certain words will appear very often in the requests, i.e. a big part of the users will review them.

We have performed an analysis on the frequency of use of the words in every text and created an ordered list of them according to the number of occurrences in the log files. The 'Critical Set' should be created for every text and will contain a number of the most requested words. Every time a text is prepared to be hoarded the words of its 'Critical Set' will be included into the hoarding set independently of the fact if it is in the user's knowledge base and is prepared for pruning. In other words even if up to this moment the system believes the user knows some of these words they will be made available during the offline period.

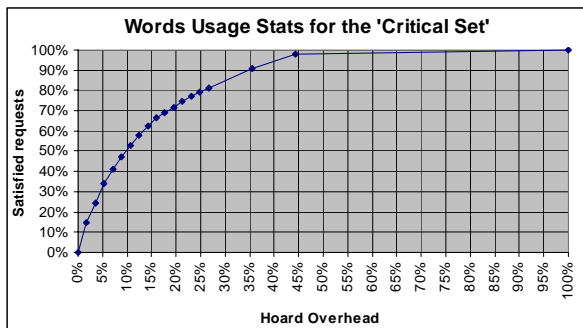


Figure 2: Words usage analysis

It is important to decide how many words to include into the 'Critical Set'. At the same time it is important to keep an eye on the overhead that the 'Critical Set' brings to the hoard. By overhead we mean what fraction of the whole set of words is included into this addition to the hoard, i.e. how bigger the final hoarding

set becomes because of the inclusion of the 'Critical Set'. On Figure 2 we show the number of satisfied requests in function of the hoard overhead. It is obvious that the bigger the 'Critical Set' is the bigger the number of satisfied requests will be, thus the smaller the miss rate value will be. On the other hand this will lead to increased size of the hoard and thus lower hit rate (percentage of pre-fetched items really used). Nevertheless one can see that the graphic is steeper at the first few percents of overhead, because of the unequal distribution of the requests. In Mobile ELDIT with only about 5% of the overhead in the hoard almost 35% of all requests will be satisfied and a 10% limit of the 'Critical Set' will lead to satisfying about 50% of the students' requests.

Though the initial idea for introducing the 'Critical Set' was to increase the accuracy (i.e. decrease the error rate) it might be used also for speeding the decrease of the hoard size. It is very possible (at least it turned to be the case for Mobile ELDIT) that part of the words in every text are never accessed. On Figure 2 such portion is about 50%. Although this might be because of the small number of test users, we would expect such behavior even with a large number of learners.

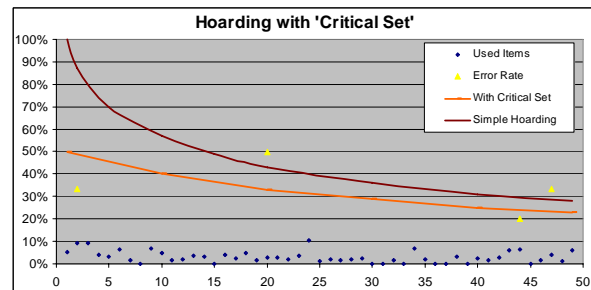


Figure 3: Hoarding with 'Critical Set'

Figure 3 shows that both the hoarding size and the error rate decrease with the help of the 'Critical Set'. One can see that still the hoard size is much bigger than the real learner's usage. At the same time, though much less than in the first trial, some errors appear.

The first effect (the hoard does not become smaller than 20%) is due to the way we were doing pruning. In fact for this experiment we were excluding two things – first, as in the first experiment (previously described), what we consider the 'user knowledge' set contains, e.g. the items that were shown to the user but he/she decided not to use; and second - the items (words of the current text) that were never used by other users. In the texts that we used these were never bigger than 50%. We would expect that it will be even smaller with a larger number of users involved. However the 'Critical Set' threshold can be set to value larger than 0, for example 5-10%. This would make the

hoard decrease more, but at the same time it would possibly increase the error rate. Thus it is important to mention that this threshold for pruning is one of the parameters that should be very carefully chosen in a real-world system.

The second effect, namely the continuing existence of hoarding misses, happens because the user actually requested an item that was never before requested by other users and thus was (wrongly) pruned. In our opinion this effect is mainly due to the small number of users that we had for the experiments. In a situation where tracking data from many more users will be available we would expect this to happen rarely.

In the cases when the device’s available memory is still smaller than the predicted hoard after the pruning step we might use the usage percentage from the statistics above as criteria for ordering the items for hoarding. The prioritizing should be done after pruning the items, which belong to the user knowledge set.

It is important to mention that with higher number of participants we would expect great diversity in users’ behaviors and in the sets of words they know. The big quantity of tracking data will lead to lower the performance of the hoarding algorithm and more specifically of the pruning, as done until now. In other words we expect that by increasing the number of users, the number of words that were never requested will decrease due to those requested rarely (i.e. by few users). Therefore when trying to apply the above described strategy for pruning we will have every time less items to prune. A way to solve this issue is to group users by similarity and do the same statistic only based on very similar users. This will imply that if a certain user has shown that s/he knows certain words we can suppose that another user with similar knowledge will also know them.

We experimented with automatic grouping of our users using different criteria and combinations of parameters. Our objective was to find if there are meaningful ways to do such grouping with the parameters extracted from the tracking data. The goal was not only to see if the algorithm can automatically group the users, but also to see if there is some persistency in the clustering, based on different chunks of data (the consecutive, but not related texts, as they were presented to the users). What was important to know is if grouping done based on part of the information (the part extracted from the first iteration(s) of the user with the system) is preserved in time so that it could be used for our needs.

Automatic grouping of the users was performed based on the data of words usage for 16 users over six texts. The grouping of the users over one text was independent of the grouping for the other texts. In Table 1 we show the results of k-means clustering [9],

where 2 clusters are produced. The clusters are marked with color and number. It is noticeable that the classification is generally stable, i.e. the users are classified quite steadily in the same cluster. Note also that half of the users (the grey-shaded ones) are classified always (i.e. for all the six texts) in the same cluster.

Table 1: Users grouping based on requested words

User	Text 1	Text 2	Text 3	Text 4	Text 5	Text 6
1	2	1	2	1	1	1
2	1	1	2	2	1	2
3	1	1	1	1	1	2
4	1	1	1	1	1	1
5	1	1	2	2	2	2
6	1	1	1	1	1	1
7	1	2	2	2	2	1
8	1	1	1	1	1	1
9	1	1	1	2	1	2
10	1	1	2	1	1	1
11	1	1	1	1	1	1
12	1	1	1	1	2	1
13	2	2	2	2	2	2
14	2	2	2	2	2	2
15	1	2	2	2	1	2
16	2	2	2	2	2	2

We can use the information produced by the clustering algorithm for measuring the similarity between users. A possible approach is to calculate it as a count of occurrences in the same group throughout different texts. Another option is to define similarity directly based on the words used in every text. This means that users that have more cases of equal known/unknown words are similar. Our experiments have shown that in most cases the similarity measured with the two methods overlaps well, however which one will suit better in the concrete application should be experimentally discovered.

We have performed experiments on clustering the users based on usage times, both for single words and aggregated data for every text. Such clustering will certainly be useful as a further step after the separation based on concrete requested words. An important question for a real-world system is “What should the number of clusters be and what does this number depend on?”. In our experiments we used two different methods – 1) let the algorithm automatically discover the best number of clusters and 2) force the creation of a given number of clusters. It proved that in some cases the automatic separation is not possible, because of the small data set we experimented with.

In certain cases the goal behind such user analysis is to predict the user behaviour and needs. Based on the clustering shown in Table 1 we experimented on using prediction algorithms. The data was split into training (60%) and testing set (30%). We used the data collected for texts 1 to 5 and the clustering as shown in Table 1 for predicting the grouping for the last text.

The algorithm (see [9]) analyses what is the best value of k, i.e. the number of neighbours to be compared during the prediction. Results are shown on Table 2.

Table 2: Users grouping based on requested words

Row Id.	Predicted Value	Actual Value	Residual	
2	1.75	2	0.25	correct
7	1.777778	1	-0.777778	wrong
8	1.25	1	-0.25	correct
13	2	2	0	correct
14	2	2	0	correct
16	2	2	0	correct

Experiments with different random separation of the users into training and testing set gave error of 17%-33% (i.e. 1-2 wrong out of 6 predictions). Prediction based on 3 clusters and also manually choosing the text over which to predict gave the same precision.

Another possible technique to be used in hoarding is to use automatically discovered associations as rules for increasing or decreasing the priority for the LO of the 'Candidate' set. For example the following rules (Table 3) are discovered over all users' requests on one of the examined texts.

Table 3: Users grouping based on requested words

Antecedent (a)	Consequent (c)	Supp. (a)	Supp. (c)	Conf.
it.n.ambiente.1.lemma=>	it.n.camicia.1.derivati	18%	37%	100%
it.v.mollare.1.lemma, it.v.stirare.1.lemma=>	it.n.gancio.1.derivati	18%	62%	100%
it.v.rendere.1.lemma=>	it.n.gancio.1.derivati	18%	62%	100%
it.v.stirare.1.lemma=>	it.n.gancio.1.derivati	25%	62%	100%

When association rules are acquired after clustering the users, as described previously, we can find even more meaningful or stronger rules (with bigger support).

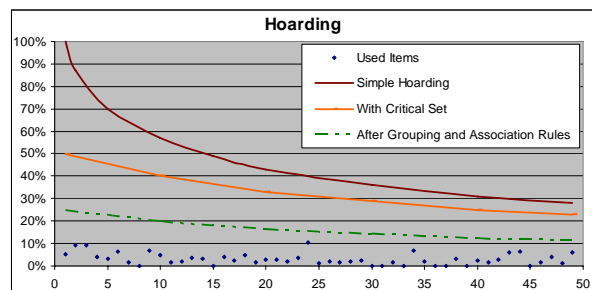


Figure 4: Hoarding (presumable) with 'Critical Set' and after LO prioritizing

The so far described clustering and association rules discovering should serve for further decreasing the hoarding set. This can be done by prioritizing the LO and setting certain limits for what to include/exclude. It will lead to situation, as shown on the Figure 4. The dash-dotted line shows the presumable size of the hoarding set if all mentioned techniques are used. How high/low the line will be, depends on the parameters (limits) set for the pruning.

5. Conclusions and Future Work

We discussed the idea of hoarding as a possible solution to make relevant data on mobile learning systems available anytime-anywhere even with memory limitations and without a permanent on-line connection. We presented the results of the hoarding sub-system of Mobile ELDIT, a mobile language learning system. We collected tracking data from a (small) number of users in Northern Italy. Such data were used to experiment with different ideas to reduce the hoarding set while minimizing the miss rate. We used the notion of "critical set" (a set of containing very important tokens that should always be hoarded), and clustered users by the similarity of their knowledge, by testing different data mining algorithms.. Our experiments with automatic knowledge extraction show positive results and thus good chances for successful implementation in various applications.

Results shown in this paper are based on small-scale experiments. It is clear that hoarding requires analyzing user behavior and knowledge. Certainty the confidence of our deductions depends on the quantity of data that was analyzed. An obvious further step would be to check the correctness of the deductions in a larger scale.

6. References

- [1] Trifonova A., Ronchetti M. "Where is Mobile Learning Going?", *E-Learn 2003*, Phoenix, AZ, USA.
- [2] Trifonova A., Ronchetti M. "A General Architecture to Support Mobility in Learning", *ICALT 2004*, Joensuu, FI.
- [3] Trifonova A., Ronchetti M. "Hoarding Content for Mobile Learning", *Int. J. of Mobile Communications* v.4(4).
- [4] Kurbel, K., Hilker, J. "Requirements for a mobile e-Learning Platform", *Proc. of IASTED 02*, US Virgin Islands.
- [5] Des Casey, "u-Learning = e-Learning + m-Learning", *Proc. of E-Learn 2005*, Vancouver, Canada, Oct. 24-28, 05
- [6] Goh T. & Kinshuk, "Getting Ready For Mobile Learning". *ED-MEDIA 2004*, Lugano, Switzerland.
- [7] Gamper J., Knapp J. "A Data Model and its Implementation for a Web-Based Language Learning System", *Proc. of WWW'03*, Budapest, HU.
- [8] Trifonova A., Knapp J., Ronchetti M., "E-learning versus M-learning: Experiences, a Prototype and First Experimental Results", *ED-Media 05*, Montreal, Canada.
- [9] Hand D.J., Mannila H. and Smyth P. "Principles of data mining", *Massachusetts Institute of Technologies, 2001*.