Hoarding Content in M-Learning Context

Anna Trifonova and Marco Ronchetti
Department of Information and Communication Technologies
University of Trento, 38050, Povo (Trento), Italy
{Anna.Trifonova, Marco.Ronchetti}@dit.unitn.it

Abstract

With the advances in mobile technologies it is already possible to support learners and teachers activities on the move. We analyzed the functionalities that should be provided by a general mobile learning platform and identified a problem that is weakly studied by previous research, namely offline usage of learning material (hoarding of content). Hoarding can use some of the techniques that are used by different caching and pre-fetching schemas, but in most cases the goal of the last two is to reduce latency time, bandwidth consumption and/or servers workload, while in hoarding the aim is to improve the accuracy of set selection. Caching and pre-fetching content are problems that are considered widely since the introduction of Internet for mass usage, still hoarding is not so well explored. We want to study the parameters that could help a hoarding algorithm improvement in order to cover the peculiarity in m-learning scenario. Our goal is to provide an efficient strategy, taking into account additional parameters, extracted automatically by the system.

1. Introduction

E-learning is growing very fast and many Universities and companies are already supporting in some way an e-learning solution. There is now little doubt that the World Wide Web is a very successful educational medium. On the other hand the rush in the field of wireless and mobile technologies creates opportunity for new field of research - so called 'mobile learning'. The domain of mobile learning can include a wide variety of applications and new teaching and learning techniques [2]. In their tries of finding the best way to apply mobile devices in education people are experimenting with different fields. Courses modules were created throughout different projects for people with numeracy and literacy problems, for kids and university students, for teachers, for studying computer science subjects, psychology or language learning.

We analyzed different ways to apply mobile devices for educational purposes. This led us to classifying services that are specific and should be provided by a general m-learning platform and later we concentrate on one of these services as a concrete problem to solve [3]. Namely this is the hoarding of content for offline usage.

Hoarding is a technique for selecting a set of documents to be pre-fetched on the user device and used when disconnected. Related terms are caching and pre-fetching, though they are more often used when considering online conditions and Web performance. Caching is a technique for keeping content that has been requested by one user available on the nearest server for a certain amount of time so other requestors can access it faster. Pre-fetching on the other hand is a technique which tries to guess what will be needed to the client in the near future, cache it and this way improve the clients' experience. Different schemes of caching and prefetching are proposed and the goal is to reducing network traffic, minimizing access latency, bottlenecks, servers' workload and etc. in the WWW world. Although the goal of hoarding content for offline usage is little shifted from the one of Web caching, some of the techniques can be reused. However while in the online case one can balance between the accuracy of the cached set and the added traffic, in the situation we consider very high accuracy is required and the added limitation is the memory availability. The learning scenario has characteristics that expose some additional information to be considered and thereby possibility to improve the existing solutions.

2. The hoarding process

The hoarding process should consist of few steps that we can formalize as follows:

1. Predict the ‘starting point’ – the algorithm should start by finding the entry point of the current user for the next learning session
2. Set its hoarding priority to MAX
3. Predict the most probable session path - the algorithm should discover the most probable sequence of LO the user will be following.
4. Create a candidate set - all related documents (objects) should be found and a ‘candidate for caching’ set of objects should be created
5. Prune the set - the candidate set should be pruned, i.e. the objects that will not be needed by the user should be excluded from the candidate set, thus making it smaller. This should be done based on user behavior observations and domain knowledge.
6. Find the priority to all objects still in the hoarding set - when the candidate set is pruned, using all the knowledge available about the user and the current learning domain, to every object left in the hoarding set should be assigned a priority value. The priority depends on how important the object is for the next user session and should be higher if we suppose that there is a higher probability that an object will be used sooner.
7. Sort the objects, based on their priority - the hoarding algorithm produces an ordered list of objects
8. Cache, starting from the beginning of the list (thus putting in the device cache those objects with higher priority) and continue with the ones with smaller weights until available memory is filled in

We can see that the algorithm will heavily depend on system’s knowledge about the user. This knowledge includes user’s learning style, natural learning habits and abilities, the level of expertise in the studying field and topic. This knowledge about the user can be acquired in different ways – by direct assessment of the user, by questionnaires and quizzes, but also by observing and analyzing the user behavior during the studying with the system, thus automatically discovering user’s learning style, preferences, acquired knowledge and etc.

For making the things clear we can consider two separate engines. One will deal with observing the user and creating user models and the other for the hoarding. We call the first one ‘User Behavior Analyzing Engine’ and it should be discussed later on. The hoarding algorithm should take as input the output from the ‘User Behavior Analyzing Engine’ (i.e. the user models with the similarities and the differences of the particular user with the common users’ behavior and the current user preferences and learning history) and additional information about the learning content itself (domain knowledge). This will be also discussed further.

Some questions appear on this stage:

- What is the best starting point for the user’s next session?
- What is a ‘session’ in the mobile learning scenario?
- How can we predict the most probable learning path (sequence of LO)?
- How do we create (formalize) a useful user model?
- What are the important parameters of the user behavior which have influence on the prediction?
- How do we use different parameters of the user model for predicting and/or pruning and do these different parameters have different significance for the prediction and/or pruning process?
- How do we formalize the domain knowledge?
- Do different domain knowledge parameters have different significance for the prediction (and/or) pruning process?
- How do we measure the successfulness of the automatic hoarding and how do we improve the work of the algorithms, considering these measures?

The rest of the paper attempts to answer these questions, starting from the last one (Section 3). In section 4 we look at the ways to find the student’s learning sequence and discuss the problem of the lack of initial data. Section 5 shortly argues about the difference between the general definition of ‘session’ in the WWW world and the one applicable in the mobile scenario. In 6 we discuss some aspects of the relations between learning objects (LO) and how their correlations can be used in the hoarding for pruning. Afterwards in Sec. 7 we discuss different possible ways to model the student and his/her behavior so we can ‘predict’ what materials will be needed during the offline period. In Section 8 we talk about the additional data about the learning material and the studied topic that might be useful for the hoarding. Conclusions in 9 are followed by references in 10.

3. Measure the quality

An important question is to measure the quality of the hoarding and to try to improve it every next time. An often used metric in the evaluation of caching proxies is the *hit ratio*. Hit ratio is calculated by dividing the number of hits by the total number of uploaded predictions (cache size). It is a good measure for hoarding systems, though a more often used measure is the *miss ratio* - a percentage of accesses for which the cache is ineffective. The authors in [1] defined a *miss cost* as a main difference in the evaluation of a caching and a hoarding system. In caching/prefetching systems the misses in the prediction reflect as a time penalty as the missing content should be retrieved from the web. This differs from the mobile case where with unavailable internet connection a miss in the hoard might be fatal. In order to quantify this measure it is possible to demand a user rating on every miss, using few different impact values. [1] also defines *time to first miss* measure - a simple count between the start of the disconnected operation and the first hoard miss. Note that this evaluation criterion can be used only on real-use of a system (and its hoard part). It is also strongly related to the hoarding size. Another possible measurement is the *miss-free hoard size,* defined as the minimum amount of disc space that a particular hoarding system would require to allow a complete disconnection period to take place without any misses.

The goal of the hoarding algorithm is to maximize the 'hit rate' and at the same time to minimize the 'miss rate'. In other words the ideal situation is to achieve hit_rate=100% and miss_rate=0%, which would mean than the hoarding set contains *all* and *only* the items that the user needs during her/his studying session as shown on the figure below.

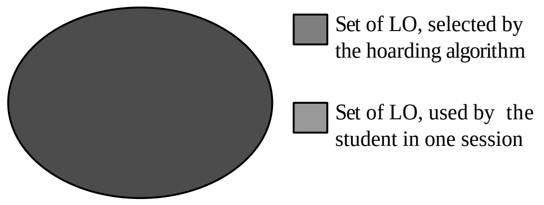
Set of LO, selected by the hoarding algorithm
Set of LO, used by the student in one session

Figure 1: *The ideal hoarding set*

Though the ideal picture is to select *all* and *only* those items that will be used by the user it is obvious that in a real system such an ideal situation is almost impossible to reach. Most probably we will have some (desirably big) overlapping between the cached by the hoarding algorithm LO and those LO really requested by the learner.

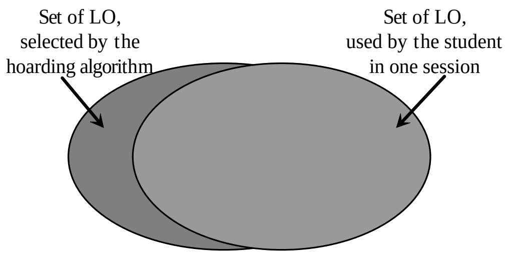
Set of LO, selected by the hoarding algorithm
Set of LO, used by the student in one session

Figure 2: *The expected picture*

If the *miss cost* measure mentioned before is used it might be better to also try to minimize the overall cost of all misses.

As mentioned before the hoarding module should be also able to analyze how successfully the previous hoarding was done for improving further prediction. For this we should be able to check which parameters or combinations of parameters of the user model and/or domain knowledge have bigger impact on the goodness of the algorithm.

By analyzing the goodness of the prediction of the hoarding algorithm we can try to tune its work. For example if a user indicates a LO miss as fatal the algorithm should check why this LO was not cached, e.g. if this entry was pruned or was given a small priority, and later the 'rules' for pruning and/or prioritizing should be reconsidered accordingly.

4. Students' learning sequences and the cold start problem

Though it should be possible to extract specific knowledge about the user behavior and to try to predict students' future steps, on the first user access to the system it (the system) is totally unaware of the properties and preferences of this specific user.

The problem, known as 'cold start', can be faced by assessing the learner knowledge through a quiz and/or questionnaire and making some assumptions.

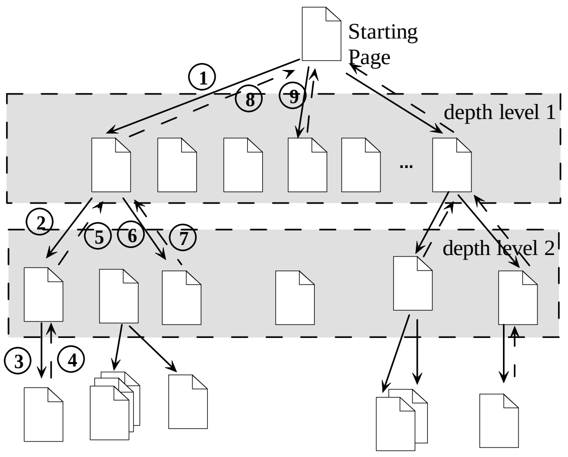

Figure 3: *Browsing path over a web-based material*

Basically the user browsing path over a web-based material (and in particular on any web-based learning source) can be viewed as a hierarchy structure (tree or directed graph). The user follows the links (the edges on the picture) from one page to another, or can go back to a previously viewed page (see fig. 3). Thus based on the knowledge about the learning material structure, the system can be aware of the most possible starting point of the student and suppose that the user will be following the links in the pages in consecutive order.

Also based on the observations on all previous users the system can estimate the average depth, in which the students browse during their first session.

In the context of caching the content on the first user access the system should upload as much as possible data trying to satisfy all user's requests.

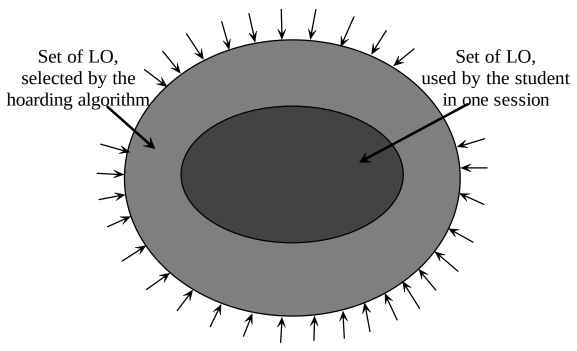
Set of LO, selected by the hoarding algorithm
Set of LO, used by the student in one session

Figure 4: *The hoarding starting step*

Later the system can try to detect the user expertise level on the study topic (by questionnaire for example) and to narrow the hoarding set using some domain knowledge (e.g. if LO_i should be proposed to advanced users, while current user is a beginner, the algorithm should exclude LO_i from the hoarding set). When no other rules can be applied to decrease the size of the hoarding set the LO left might be randomly uploaded until the memory limit is reached.

5. Defining Session Length

In the Internet world a session is defined as "a continuous period of time during which a user's browser is viewing Web pages or a Web application within the same server or domain" (source - MSDN Library). It is series of transactions or clicks on the web pages links made by a single user. There are different criteria to decide if a session is over or not. Two of the most commonly used are the session length and the inactivity period of the user. For the first method the time limit for the session length is set to a certain value and the activities later than this limit (counting from the start of the session) are considered a new session. In the second method if the user activity stops for a certain period of time on the resumption of the activity by the same user a new session is considered started.

On the other hand for hoarding in a mobile system the importance falls on the time between two possibilities of the user to synchronize with the main server. In this sense we find more useful in this context to define a session as *the time between two synchronizations* of the mobile device with the main online system. The default session length might be one day, as commonly synchronization is done once per day, but during the system usage other session length might be observed and explicitly set for every user.

6. Links and correlation between LO

As mentioned earlier one of the steps of the hoarding algorithm is to construct the 'candidate' set of objects, related (linked) to the starting point or to other objects that were predicted to be viewed. When using a web-based material the user clicks on the links of one page to go to another one and can either continue to browse further or can go back to a previously viewed page. The links between the pages give us the structure of the web site (a learning material in particular), thus we can extract the relation between the LO, for example by parsing the pages.

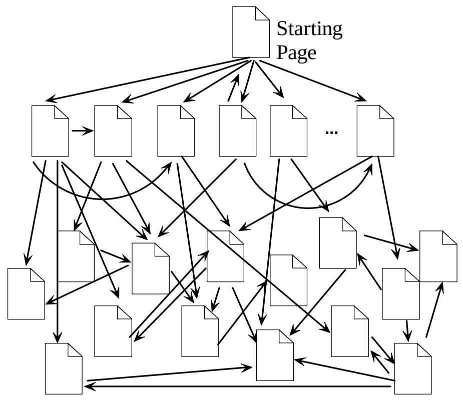

Figure 5: *Web-based material structure*

The links might be either bi-directional or not. We can build a LO correlation table in the following way:

```
for (every LO) {
 create a row;
 for (i=1, number_of_LO, i++) {
  if current_LO contains link to LOi
    set celli = 1;
  else set celli=0;
 }
}
```

In the table we can see that LO_1 contains link to LO_2 and to LO_n , but not to LO_3. The link is bi-directional for LO_2 and LO_3. In this way we can easily observe the set of LO that the user will be possibly requesting if he/she decides to browse deeper in the site, i.e. to go one level of depth further. Those are the objects directly linked to a particular object.

	LO_1	**LO_2**	**LO_3**	**…**	**LO_n**
LO_1	x	1	0		1
LO_2	0	x	1		1
LO_3	1	1	x		0
…				x	
LO_n	1	0	1		x

From the table above we can easily construct the 'candidate' set of LO for every next level of hoarding. Later this candidate set will be pruned (its size can be decreased by dropping some of the objects that are not likely to be requested).

On the other hand we can analyze the correlation between the objects, based on their contemporary usage in other user sessions. For example association rules can be discovered over all users' sessions containing an upper level LO. We can take into account only 'very strong' connections, i.e. associations discovered with confidence near to 1 and big enough support value. Note that it is expected that not a lot of such associations will be found, as the common scenario is to have big variety of LO and also big diversity of students' knowledge, interests and learning preferences. The rules extracted in this way will be of the type $LO_i \Rightarrow LO_j$: conf=0.99 sup>0.5 which we can read as "Almost every time when the LO_i was viewed also LO_j was viewed in the same session (where LO_i can be an example problem and LO_j the solution given by the lecturer)".

Example:

	LO_1	**LO_2**	**LO_3**	**LO_4**	**LO_5**	**LO_6**
$Session_1$	0	0	0	1	1	1
$Session_2$	1	1	1	0	0	0
$Session_3$	0	0	0	1	1	1
$Session_4$	0	0	1	0	1	1
$Session_5$	1	1	1	0	0	0
$Session_6$	1	0	1	0	0	0
$Session_7$	1	0	0	0	1	1

* Where 1 means that LO_i was viewed during $Session_j$ not taking care of the sequencing.

Association rules algorithm discovers with confidence=1 the following relations:

$LO_1 \Rightarrow LO_6$; $LO_2 \Rightarrow LO_1$; $LO_2 \Rightarrow LO_3$;
$LO_4 \Rightarrow LO_5$; $LO_5 \Rightarrow LO_6$; $LO_6 \Rightarrow LO_5$;

Association rules can be discovered also in more limited number of sessions (not all at a time), for example search for correlated objects only in the sessions of users that ware classified in the same group as the user for which the current hoarding set is being prepared. Considering the example above if we apply a clustering algorithm (for example k-means), the algorithm produces 2 clusters from the above shown data. Applying association rules only to the sessions in the same cluster we get some additional associations. The clusters and discovered associations are as follows:

Cluster	**Instances**	**Associations**
$cluster_0$	$Session_1$ $Session_3$ $Session_4$ $Session_7$	$LO_1 \Rightarrow LO_5$ $LO_3 \Rightarrow LO_5$ $LO_3 \Rightarrow LO_6$ $LO_4 \Rightarrow LO_6$
$cluster_1$	$Session_2$ $Session_5$ $Session_6$	$LO_1 \Rightarrow LO_3$ $LO_3 \Rightarrow LO_1$

The above associations (like $LO_1 \Rightarrow LO_5$) show that if the object LO_1 is to be selected for the hoarding set there is big probability that the user will also be accessing the object LO_5 during the same session. Moreover associations of the type LO_5=1 & LO_6=1 $\Rightarrow$ LO_2=0 can also be discovered, showing that if the user will be viewing objects LO_5 and LO_6 it is most probable that the object LO_2 will not be viewed, thus can be excluded from the hoarding set or at least its hoarding priority can be set to much lower level.

For the example above we considered only associations with confidence=1 and any support greater than 0, but in a real situations the best values for these parameters should be discovered experimentally.

The confidence value of the discovered associations LO can help us also in placing the items of the 'candidate set' in an ordered list.

Also other data mining and/or machine learning algorithms should be considered and tested to see their appropriateness for the hoarding process and how they can be combined best.

7. User modeling

In the literature one can find lots of different ways to model a user and/or her behavior depending on the application and its needs. In the context of hoarding we recognize two groups of characteristics that will be used differently by the algorithm. We schematically call the first 'user behavior' and the second 'user knowledge'. Depending on the mobile learning system it is possible that not all the parameters can be discovered or they might be discovered through different techniques. The data about the user might be obtained by (any combination of) questionnaires, tests and quizzes or automatically by tracking the user and analyzing the log files. The process for retrieving automatically the user patterns consists of few steps, shown in the figure below. The first step is the preparation of the data for analyzing. For this step the log files should be preprocessed and integrated into a database. Afterwards different data mining algorithms can be applied to extract interesting relations.

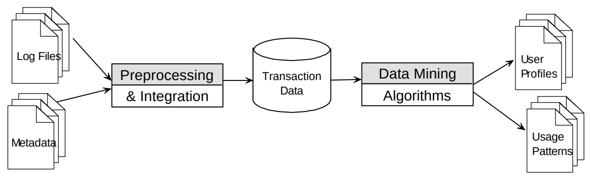

Figure 6: *Architecture for deriving user profiles*

The *user behavior* can be described in terms of browsing styles (e.g. consecutive, random, interest driven, etc.); preferred type of educational media (e.g. prefers video to combination of text and pictures); … Based on the user behavior we can group the learners and analyze the similarities and differences between the groups and between the members of the same group. This should help us to predict what will be needed, i.e. this data will be used to fill-in the hoarding set.

On the other hand the *user knowledge* profile should consist of everything that the system knows about what the user already knows. Example is the system awareness of the user's competence in a certain subject (i.e. beginner, intermediate, advanced) or a list of all the topics already covered by the user previously. In contrast of the *user behavior* the profile of the *user knowledge* will be user for pruning the entries from the hoarding set, i.e. for excluding objects in order to decrease the size of the hoard.

We can distinguish static data about the user and dynamically changing data. The static data is for example the user age, gender, mother tongue and etc. On the other hand the dynamic data is our current knowledge about the changeable over time user parameters and should be reviewed in certain periods of time. For example the user browsing pattern might change drastically few days before an exam date, thus the hoarding system should be able to quickly recognize such changes and react accordingly.

8. Metadata | Domain Knowledge

The metadata represents specific domain knowledge or knowledge about the specific learning material. Metadata is in general provided by the educator or the learning material creator. It will generally vary from one application to another, but can be used by the hoarding algorithm to improve it work.

One direction is to help in solving the 'cold start' problem by providing specific knowledge about the learning material structure, like 'initial point', provisioned common learning path, or connections between individual LO. The relationships between LO can also influence different weights of the parameters that are forming the priorities of the LO while hoarding.

9. Conclusions

In this paper we have described the hoarding problem for a mobile user without connection. The problem is how to support work on a mobile device when it is impossible to load on the mobile device all the data that comprise the full knowledge.

We have outlined a general algorithm, and we have posed a number of questions that need to be answered in order to solve the problem. We have also attemped to give some first answers to these questions. Our work is still in progress.

10. References

[1] Kuenning, G.H., Popek, G.J., "Automated Hoarding for Mobile Computers", *Proc. of 16th ACM Symposium on Operating Systems Principles*, St. Malo, France, Oct. 1997.

[2] Trifonova A., Ronchetti M., "Where is Mobile Learning Going?", Proc. The World Conference on E-learning in Corporate, Government, Healthcare, & Higher Education (E-Learn 2003), Phoenix, Arizona, USA, November 7-11, 2003.

[3] Trifonova A., Ronchetti M., "A General Architecture to Support Mobility in Learning", *Proc. of the 4th IEEE International Conference on Advanced Learning Technologies (ICALT 2004 - "Crafting Learning within Context")*, August 30 - September 1, 2004, Joensuu, Finland.