Proposing "collaborative filtering" to foster collaboration in ScratchR Community

Georgios Fessakis & Angelique Dimitracopoulou
LTEE laboratory, University of the Aegean, Demokratias 1, 85100 Rhodes, Greece
gfesakis@rhodes.aegean.gr; adimitr@aegean.gr

Abstract: The present work focuses the interest of study in a naturally emerged and intense online community, this of ScratchR "programmers by choice" community, that actually practice collaborative learning in an authentic way. Our interest is not to support collaborative learning process, but to foster collaboration opportunities. We propose a personalized recommendation system based on a "collaborative filtering" technique aiming at inciting collaboration and increasing the frequency of Scratch projects remixing, in order to foster collaborative learning. In this paper the proposed collaboration fostering mechanism is outlined with the assistance of a test data set. The significance of the proposal is discussed while the future work is described.

Introduction

Computer programming is generally considered an important competence not only because of its economic significance but also of its value as a learning environment. More recently computer programming is referred to as literacy for modern society which enables people to be active contributors to interactive digital content on the web2.0 (Monroy-Hernandez, 2007). Despite its significance, computer programming learning is recognized as a difficult task for students (Wiedenbeck, 2005). In order to lower the barriers in the development of computer programming skills, several approaches have been proposed, such as: educational programming languages (e.g. Logo, Scratch, Alice) to make the programming more attractive (Guzdial, 2004), and the study of contexts for informal learning of computer programming (Maloney et al, 2008). In this paper we focus on the Scratch educational programming environment and especially on the ScratchR online community where several individuals, hobbyists in their majority, are learning computer programming collaboratively for entertainment and in an informal context. The capability of ScratchR community for Scratch projects remixing, constitutes a central mechanism of collaborative learning. In this paper we propose a recommendation system based on a "collaborative filtering technique" aiming at inciting collaboration according to the members' preferences and increasing the frequency of Scratch projects remixing, in order to foster collaborative learning within the ScratchR community. In the followings the Scratch environment and ScratchR community are described first in order to understand the context of the proposed collaboration fostering application then the proposed recommendation mechanism is outlined with the assistance of a test data set. Finally, in the discussion section all of the above is summarized, while its significance and future work are described.

About Scratch and ScratchR community

Scratch (MIT Media Lab, 2008) is an educational programming environment developed by the Scratch project from MIT Media Lab. Scratch encompasses a graphical programming language which makes programming more accessible to children (ages 8 and up), teens and other novices to computer programming for the developed of media rich projects. The ScratchR (Monroy-Hernandez, 2007) is a web site (scratch.mit.edu) where Scratch users can share their projects. ScratchR facilitates collaborative learning of computer programming through (Monroy-Hernández & Resnick, 2008): **(a) Inspiration:** browse, download and reuse project elements; **(b) Feedback from and to the community:** the other users serve as an audience with varying expertise levels giving opportunities for peer teaching and scaffolding; **(c) Creative appropriation (remixing):** the modification, extension, correction of other's projects. Projects' remixing is considered by the authors as a major observable expression of collaboration among the Scratch community members. The current extent of collaborative learning through creative appropriation in ScratchR is clarified in the following quote: *"The Scratch website serves as a repository of code and ideas that can be creatively appropriated to spawn new ideas and new projects. … Fifteen percent of all of the 23,294 projects shared (as of August 14, 2007) where remixes of other projects. Of those, the types of changes made ranged from simple changes to image and sounds to modification of the actual programming code."* (Monroy-Hernández & Resnick, 2008). It is reasonable to assume that devising a mechanism to increase remixing in ScratchR community will probably increase collaboration and collaborative learning.

Proposing a collaborative filter for ScratchR remixers recommendations

Goldberg et al, 1992 used the term "collaborative filtering" for the first time in a system called "Tapestry". "Collaborative filtering" refers to a set of techniques that exploit social annotations and interaction data from the

members of a community to facilitate searching in a large set of multimedia objects. One popular example is that of tags that users attach to objects, thus formulating flat categorization systems known as "folksonomies" which are depicted by "tag-clouds". ScratchR community does not have any personalised recommendations system yet. By using collaborative filtering in ScratchR, it is possible to build a recommendations mechanism that after taking into account projects that have been remixed by other "similar" users, would propose projects for remixing to each member personally.

We are aiming to recommend projects for remix from the more frequently remixed projects. ScratchR offers the list of top remixed projects along with the list of remixers' psedo-names. We use this list to collect the preferences of the users to projects for remix. Usually the first step of a collaborative filtering algorithm is to estimate the similarity of each user from a large set, to a specific user. In our case we need to estimate which users prefer to remix the same kind of projects and then suggest to a specific member a list of projects that he/she has not yet remixed or even studied and has been remixed by similar users.

As a first step a way to represent remix preferences is needed. Preferences can be represented by an Array: $RP \in \{0,1\}^{nxm}$, where $RPij = \begin{cases} 1 & \text{if project(j) has been remixed by user(i)} \\ 0 & \text{otherwise} \end{cases}$, and n=number of users while m=number of projects. In real applications it is necessary to limit large data sets for example, in our example data set the 10 most remixed projects which involve 1960 members will be used. For example the RP for the top seven remixed projects of ScratchR on 25/10/2008 and for five of the members, is in Table 1. The raw data has been collected manually from ScratchR. Using this small set of data it is possible to see that the project "*3 Trampoline"* could be recommended to users *Fetsch*, *kaotheroogoncreator*, and *mtaylor* etc. Of course these recommendations are not so obvious when using the whole set of data because it is not easy to locate similar members. So we need a way to estimate similarity of members.

Table 1. Remix Preferences array subset for test data.

User	1 PacMan	2 Pong	3 FishChomp	3 Trampoline	4 MarbleRacer	6 Doodle	Doodle
Planetbravo	1	1	1	1	0	1	1
Fetsch	1	1	1	0	1	1	0
kaotheroogoncreator	1	1	1	0	1	1	1
Mtaylor	1	1	1	0	1	1	1
olly144	1	1	1	1	1	1	1

The second stage in the process is to determine the degree of members' similarity in terms of projects they prefer to remix. In other words we need to compare the set of remixed projects for each member to the remixed projects set of every other member. In order to do this we need to choose a similarity metric (Segaran, 2007). In our case Jaccard coefficient is used in which joint absences are excluded from consideration. Jaccard coefficient gives equal weight to matches and nonmatches. By computing this similarity measure for each possible pair of users we obtain a square Similarity nxn Array: SA, where $SA_{ij} = \frac{\sum_{k=1}^{m} RP_{ik} \otimes RP_{jk}}{\sum_{k=1}^{m} RP_{ik} \oplus RP_{jk}}$, ($\sum_{k=1}^{m} RP_{ik} \oplus RP_{jk} \neq 0$, since we consider member with at least one remix). From the computation formula of Jaccard coefficient it is obvious that $SA_{ij} \in [0,1]$, furthermore $SA_{ij}=0$ means that user(i) and user (j) have not remixed any common projects while $SA_{ij}=1$ means that user(i) and user(j) have remixed exactly the same projects. For the test data set mentioned above we can see a subset of SA in Table 2. To produce recommendations for a specific member we order the list of other users in decreasing order of similarity. Then we propose projects from the remixes of the more (but not absolutely, SAij=1) similar users which has not been remixed by the specific member. For example to user *__sakura__* we can propose projects from the remixes of users *100seconds*, and/or *101eisoJ*, and/or *110ronaldo*.

Table 2. Excerpt from the similarity matrix (SA) for the ScratchR test data set

	__sakura__	02sergi02	06howardr	100seconds	101eisoJ	110ronaldo
__sakura__	1	0,000	0,500	0,500	0,500	0,500
02sergi02	0,000	1	0,000	0,000	0,000	0,000
06howardr	0,500	0,000	1	1,000	0,000	0,000
100seconds	0,500	0,000	1,000	1	0,000	0,000
101eisoJ	0,500	0,000	0,000	0,000	1	1,000
110ronaldo	0,500	0,000	0,000	0,000	1,000	1

An example of the potential

In order to have a practical example of the potential of the proposed technique we are going to apply multidimensional scaling on the Similarity Array. This is going to give us a graphical summary of the complex relations between members. It is interesting to see an example of the recommendations produced by the above technique and to estimate their quality. An excerpt of the two dimensional diagram from the Multidimensional Scaling to the Similarity Array appears in figure 1. From the diagram we can see groups of similar members that remix rather the same group of projects. Members in the center remix the most remixed group of projects.

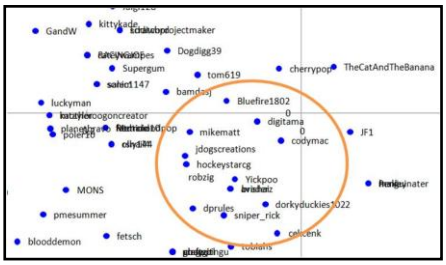

Figure 1. Excerpt from the central part of MDS diagram of SA

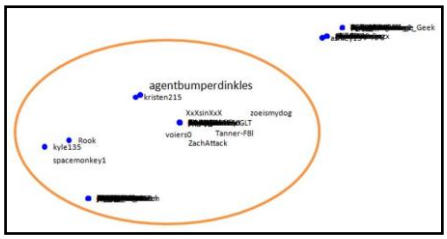

Figure 2. Excerpt from a marginal part of MDS diagram of SA

When we want to make recommendations for one specific member we have to consider the nearest neighbors of him/her. For example let's focus to user *jdogscreations* who (according to the RP) has remixed the following projects: "*1 PacMan"*, "*2 Pong"*, "*4 MarbleRacer"*, "*Doodle"* and *"Marble Racer starter"*. User *hockeystarcg* is a close neighbor of *jfogscrations* and has remixed the projects: *"2 Pong"*, *"4 MarbleRacer"*, *"6 Doodle"*, and *"Marble" "Racer starter"*. So the system could suggest to *jdogscreations* to consider projects *"1 PacMan"* and *"6 Doodle"* for remix. Let's take another random sample from the marginal area of the diagram (Figure 2). In the margins users tend to be more exceptional and not so similar. Following the same procedure the system could suggest to user *agentbrumperdinkles* who has remixed the projects: "*1 PacMan"* and "*3 Trampoline"* the project "*Perfect Sidescrolling Engine v1.0"*. This is quite an interesting recommendation because the suggested project is a model project that shows how to implement side scrolling in Scratch; this could empower the user to build more complex programs. Even these small data set shows that it is possible to build one recommendation mechanism for the ScratchR community that could foster the collaborative learning of programming.

Discussion

Computer programming is an educational interesting competence for economical and general learning reasons. Scratch is an educational programming environment which is quite popular and successful. The system provides an easy introduction to computer programming. In addition ScratchR provides user of Scratch with an online community where they can publish, share, comment, and creatively appropriate (remix) projects. In this paper we proposed and described the development of one collaboration fostering mechanism, based on the collaborative filtering technique for the production of personalized suggestions lists with projects recommended for remix. Remix of a project is considered a significant collaboration event in scratch and in every programming environment. The proposed mechanism, increases the frequency of projects remixing and constitutes a collaboration fostering mechanism. There are many other possibilities to apply interaction analysis (Fesakis, Petrou, & Dimitracopoulou, 2004; Dimitracopoulou, 2008) techniques in ScratchR. For example, it is possible to make recommendations of members for collaboration, projects for examination, galleries to participate etc. This work is going to be continued after the initial exploration of these possibilities in ScratchR and other communities, with the provision of the personalized recommendation system, its evaluation by the community members and the study of its effects

References

Dimitracopoulou A. (2008). Computer based Interaction Analysis supporting self-regulation: Achievements and prospects of an emerging research direction. In TICL Journal, 6(3) .

Fesakis, G, Petrou, A, & Dimitracopoulou, A (2004). Collaboration Activity Function: An interaction analysis tool for CSCL activities. In Proceeding of the ICALT, Sep.1, 2004, Joensuu, Finland pp. 196-200

Goldberg, D., Nichols, D., Oki, B. M., & Douglas, T. (1992). Using collaborative filtering to weave an information tapestry. Communications of the ACM , 35 (12), pp. 61-70.

Guzdial, M. (2004). Programming environments for novices. In S. Fincher, & M. Petre, Computer science education research (pp. 127-154). Lisse, The Netherlands: Taylor & Francis.

Maloney, J. H., Peppler, K., Kafai, Y. B., Resnick, M., & Rusk, N. (2008). Programming by choice: urban youth learning programming with scratch. ACM SIGCSE Bulletin , 40 (1), pp. 367-371.

MIT Media Lab. (2008). Scratch. Retrieved OCT 27, 2008, from http://scratch.mit.edu

Monroy-Hernandez, A. (2007). ScratchR: a platform for sharing user-generated programmable media. In M. B. Skov (Ed.), 6th International Conference on Interaction Design and Children. Aalborg, Denmark.

Monroy-Hernández, A., & Resnick, M. (2008). Empowering kids to create and share programmable media. Interactions, March-April , pp. 50-53.

Segaran, T. (2007). Programming Collective Intelligence. Sebastopol: O'Reilly Media, Inc.

Wiedenbeck, S. (2005). Factors affecting the success of non-majors in learning to program. In Proceedings of the 1st int. Workshop on Computing Education Research , Seattle, USA Oct 1 -2, 2005 (pp. 13 - 24)