A Feasibility Study on using Clustering Algorithms in Programming Education Research

Marzieh Ahmadzadeh
School of Computer Engineering & IT
Shiraz University of Technology
Shiraz, Iran
ahmadzadeh@sutech.ac.ir

Elham Mahmoudabadi
School of Computer Engineering & IT
Amirkabir University of Technology
Tehran, Iran
mahmoudabadi@aut.ac.ir

ABSTRACT
Designing an experiment for programming education research, in which collecting and interpreting a large number of qualitative data about programmers is required, needs careful consideration in order to validate the experiment. When it comes to finding a pattern in the programming behaviour of a specific group of programmers (e.g. novice, intermediate or expert programmers), one of the critical issues is the selection of similar participants who can be placed in one group. In this study we were interested in finding a method that could shorten the path to finding participants. Therefore, the use of clustering algorithms to group similar participants was put to test in order to investigate the effectiveness and feasibility of this approach. The clustering algorithms that were used for this study were K-means and DBSCAN. The results showed that the use of these algorithms, for the mentioned purpose, is feasible and that both algorithms can identify similar participants and place them in the same group while participants who are not similar to others, and therefore are not the correct subject of the study, are recognised.

Categories and Subject Descriptors
D.1.5 [Object-oriented Programming], D.2.5 [Testing and Debugging]: Code inspections and walk-throughs.

General Terms
Algorithms, Measurement, Experimentation.

Keywords
Program Comprehension, Experiment, Clustering Algorithm, K-means, DBSCAN, Computer Science Education.

1. INTRODUCTION
Most programming research, including program comprehension research, is by nature experimental [1, 4, 7, 11] where several issues must be taken into consideration to maintain the validity of the experiment. The first issue is the choice of participants, subjects who should have the qualities that the experiment is based on. For example, when novice programmers are the subjects of the study, a researcher must pay attention to choose participants who fit the definition of novice in a novice programmers’ group category. Likewise, when expert programming ability is studied, one should think about what makes a programmer an expert.

The second issue is the method by which data is collected. Again, in this kind of research in which programmers’ behaviour is studied, data gathering is done by methods such as observation, interviews, data logs, and so on. These methods of data gathering are qualitative which require one to analyse the data qualitatively. Among the huge amount of data collected by these approaches are always large amounts of data which are not significantly important to the analysis. These can be identified by the analyser in the process of interpreting the collected data. This, however, is a time consuming job.

The third issue is that the process of interpreting the collected data is subjective, therefore, if the amount of collected data is huge, and consequently the number of analysers is more than one, it will likely lead to inconsistent results.

The issues mentioned above motivated us to develop a method which helps us in many ways. This included: selecting the right participants by filtering potential subjects by various metrics, which were defined by us; and, reducing the amount of qualitative data collected in the experiment to lead us to focus on more important data to eventually assist us in having more consistent data analysis.

The solution to these problems, offered by us, is the use of clustering algorithms. In this research we chose two clustering algorithms, K-means[12] and DBSCAN[6]. We studied the feasibility of using these algorithms in order to categorize the right participants in the same group to be further analysed qualitatively. Therefore, two issues have been tested in this research: a) does this method (the use of clustering algorithms) of data summarization work for this kind of research (i.e. programming research); and, b) which algorithm is more appropriate and shows better results.

The K-means and DBSCAN algorithms were chosen as an instance of centre-based and density-based clustering algorithms respectively. Since this was a feasibility study, other algorithms were not tested in this experiment.

The organization of this paper is as follows: a review of literature is given in the next section; the third section presents the two clustering algorithms that are used for this research; in the fourth section the experiment and its results are addressed; and finally, the last section discusses the conclusion of this study.
Two types of research are related to the current study. The first group is concerned with learning to program and the second group focuses on the application of data mining algorithms including clustering algorithms. Our research is concerned with the application of clustering algorithms in program learning research.

Research concerning learning to program has spanned nearly four decades with an endless variety of approaches. The broad range of research in this field consists of proposing a cognitive framework to explain the process of learning to program and comprehension to developing tools to assist teaching [5] or measuring a tool’s feedback on programmers’ motivation to program. As an instance, in 1979 Schneiderman and Mayer [12] proposed a cognitive framework to explain programmers’ behaviours when they were composing, debugging and modifying a source code. The participants were selected from the range of novices to expert programmers. One of the most recent studies concentrates on how different feedback from a tool impacts the motivation of programming and introduces a tool that was designed for this purpose[8].

The application of data mining algorithms, including clustering, has been broad. For example, to analyse a log file in order to get information for debugging the program, the K-means algorithm has been used [9]. In other research, text-mining was used to extract functionality of a system from a textual requirement [3]. To produce initial data for evaluation of plagiarism detection a variant of K-means (i.e. bisecting K-means) was used [10].

Investigating the literature reveals no research in which the technique of clustering, in order to study the process in which students comprehend or develop a program, was used. Therefore, this research was developed to act as a feasibility test on using these techniques to make this type of study easier.

3. CLUSTERING ALGORITHM

One way to summarize a large amount of data is to use clustering techniques to group data in a meaningful way so that the objects inside the groups, or clusters, have the most similarities while objects in different groups have the most differences.

Two types of clustering algorithms are available: nested and partitioned. A nested clustering algorithm creates overlapped clusters while a partitioned clustering algorithm creates non-overlapping clusters. For programming research, in which differences between groups of programmers is required, the second type of clustering algorithm is more appropriate. For this research we considered two partitioned algorithms called K-means and DBSCAN, which will be explained in the next two subsections. These algorithms were chosen as representatives of two types of portioned clustering algorithms i.e. center-based and density-based.

3.1 K-means Algorithm

This algorithm is a center-based clustering algorithm in which a most representative point (object) for one cluster is chosen and the distance between the representative point and all other points (or objects) are computed. Since one can choose K clusters, this algorithm is called K-means. This means that K representative objects (i.e. centroid or medoid) are selected. Each object is then assigned to the closest centroid and therefore, the related cluster. In the next step, the center point is updated according to the objects that have been assigned to the cluster. For the newly created centroid, the new members of the cluster are computed again. This process is repeated until the centroids do not change. At this point the members of each K cluster are known [12].

3.2 DBSCAN Algorithm

This algorithm is among the group of algorithms that are based on density. This means that a cluster is a dense part of the objects which are surrounded by a low density collection of data. In this algorithm, any two objects that are close enough within a pre-specified distance (epsilon) are assigned to one cluster. A cluster is formed if it exceeds the specified minimum number of objects (minpoint). The number of clusters, therefore, is estimated by the density distribution of the objects. Thus, the distance and the number of objects required to form a cluster are the two parameters needed for this algorithm. The algorithm starts from a random point and computes the epsilon-neighbourhood of that point. If the number of these points exceeds minpoint, a cluster is formed otherwise, the point is introduced as a noise. This process is repeated until all points of data are labeled [6].

4. EXPERIMENT

To investigate whether or not clustering algorithms can help in grouping similar programmers correctly, to be further studied qualitatively, an experiment was designed with 7 participants who are called A, B, C, D, E, F and G for ease of use. Since this was a qualitative research, the low number of participants was not an issue. All seven programmers were novice students who had recently learned object-oriented topics, finished their first semester and achieved almost similar marks. This means that we selected students who had the same efficiency in our classes. Some of these students had some prior experience in programming and high interest in this job while others had no experience and less interest. The summary of the information that we had from these students can be seen in Table I. Since exam marks and the degree of interest were very close it seems reasonable that students with programming experience cluster together against others. We kept in mind that the ability in programming is not the same as the ability in program comprehension [1]. The marks show their programming ability only. Therefore, programming marks might not be an indicator of other abilities such as program comprehension. In this experiment, we were interested in grouping the students based on their program comprehension ability therefore, an experiment that put their program comprehension ability to test was designed.
see how one student’s approach in a group was different from his/her peers that could affect performance, which was not possible otherwise. In other words, we wanted to find the differences between the processes in which problems are solved by different groups of novices in order to get a better understanding of their program comprehension differences. We can then take advantage of these findings to improve our teaching.

In this study two similar experiments were designed. The experiments ran in two phases in which participants read the given program specifications and completed the given code in order to get a full working code. We chose to have two experiments since we wanted to evaluate the results from the first experiment by the results from the second experiment. If the results achieved from the second experiment matched the results from the first, we could assert that this method was working.

While students were carrying out the given task, their actions were logged by a screen recorder software called Camtasia and later translated into an activity code. For example, defining an instance variable, running the program, switching from one class to another, moving the mouse, defining a class, typing static in a method signature, waiting, and all other activities that were observed were given a unique activity code. For the kind of problem that was given, 86 distinct activity codes were defined. Some of these activities were ones that we expected based on the task that was given (for example defining an instance variable) and others were recognised by actions that were observed from the captured video. For each participant, therefore, a series of the activity codes, sorted by the time of occurrence, were inserted into a table in a database which included a total of 4278 records. Then K-means and DBSCAN clustering algorithms developed in Weka [2] software were applied to the data. The same process was repeated for the second experiment and the results were compared.

4.1 Clustering by K-Means Algorithm

This experiment was repeated three times, each time with a different number of clusters i.e. two, three and four clusters. In these experiments, during which we were looking for participants who showed similar patterns in comprehending the given program and completing the task of programming, a proper number of clusters were numbers for which more than one member could be assigned to. This meant that clusters with one member were not desired. Therefore, it did not make sense to have more than four clusters because most of the clusters would definitely include only one participant. We were not sure about four clusters therefore, we had to put them in the test.

As can be seen from Table II, when two clusters were chosen, four out of seven participants (i.e. A, B and E, F) stayed in the same group in each of the experiments and three other participants resided in different groups. If we have a second look at Table I, this grouping makes sense since both A and B had no experience and E and F had some experience.

Although the participants’ exam marks were close, A and B had closer marks. The same was true for E and F. Participant C had a similarity with A and B (i.e. marks) and a similarity with E and F (i.e. experience) which meant there was no perfect match between those groups. Without looking at the process of program comprehension we did not know in which group participant C should reside. K-means algorithm put this participant in the same group as E and F and the same group as A and B for the first and second experiments respectively. This was also true for participant G. However, the story was different for participant D which seemed more similar to A and B in terms of marks and experience but had been grouped with different clusters each time.

<table>
<thead>
<tr>
<th>Table II. K-Means Clustering Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Clusters</td>
</tr>
<tr>
<td>C1</td>
</tr>
<tr>
<td>C2</td>
</tr>
<tr>
<td>3 Clusters</td>
</tr>
<tr>
<td>C2</td>
</tr>
<tr>
<td>C3</td>
</tr>
<tr>
<td>4 Clusters</td>
</tr>
<tr>
<td>C2</td>
</tr>
<tr>
<td>C3</td>
</tr>
<tr>
<td>C4</td>
</tr>
</tbody>
</table>

A clustering was carried out with the same algorithm and three clusters. Again, A and B stayed in the same group and E and F were in another group while other participants were grouped differently in both experiments.

Executing a clustering algorithm with four clusters did not make sense since in the second experiment most of the clusters included only one member. This was not desired for the kind of research we were conducting.

As a result of these clusterings, the right participants for program comprehension were A and B versus E and F. To validate these results we needed to do clustering with another clustering algorithm, which is seen in the next subsection.

4.2 Clustering by DBSCAN Algorithm

As explained previously, the DBSCAN algorithm does not input the number of clusters instead, it calculates it by itself. The number of clusters is computed by the minpoint and epsilon that is provided in this algorithm. We set the minpoint to 6 for all the clustering that we ran, which means the numbers in each cluster could not be less than 6. This was not a problem because for one participant we had more than 6 records of data. For example, for participant F we had 342 records of data for one experiment only. We assigned different values to epsilon to receive the number of clusters that we desired to enable us to compare the results with K-means clustering. For example, to have two clusters epsilon was set to 1.1 and 0.9 for the first and second experiments respectively. This number was 0.9 and 0.8 for three clusters and 0.7 and 0.66 for four clusters in the first and second experiment respectively.

<table>
<thead>
<tr>
<th>Table III. DBSCAN Clustering Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Clusters</td>
</tr>
<tr>
<td>C1</td>
</tr>
<tr>
<td>C2</td>
</tr>
<tr>
<td>3 Clusters</td>
</tr>
<tr>
<td>C2</td>
</tr>
<tr>
<td>C3</td>
</tr>
<tr>
<td>4 Clusters</td>
</tr>
<tr>
<td>C2</td>
</tr>
<tr>
<td>C3</td>
</tr>
<tr>
<td>C4</td>
</tr>
</tbody>
</table>

As seen from Table III, when the algorithm clustered data into two groups, it put participants A and B in the same group and
participants E and F in the same group in both experiments. The results were not as consistent for other participants. The same was true when the algorithm clustered the data into three groups. Again, grouping data into four clusters was not valid for our research since most of the clusters in the second experiment included only one element of data.

The significant point of the experiment was that the results achieved from this algorithm perfectly matched the results achieved from K-means when data were grouped into two clusters. This was true even when four clusters were formed. The results of clustering for three groups were the same for both algorithms in the second experiment only. But we concluded for sure that both algorithms put A and B together and E and F together.

5. CONCLUSION
In this research we aimed to find the right participants to be studied against each other. We proposed to use one of the clustering algorithms. We could have used one algorithm like K-means or DBSCAN, but our results needed to be validated. Two approaches were available for us to validate a clustering, an internal and an external approach. In an external approach there should be some external information available that could validate the clustering. The only information available was a student’s programming marks and prior experience in programming. The results suggest that participant E and F could be grouped and studied against participants A and B. If we look at their programming marks and their prior experience as an external metric, we see that these results are valid (see Table I). To validate this clustering internally, we used two clustering algorithms instead of one, to see if the results matched. It can be seen from Tables II and III that similar groupings can be extracted from the result.

In this research, the first question that was answered was that it is possible to use clustering algorithms to group programmers in order to study their behaviour. This was seen by the consistent results that were achieved from running two clustering algorithms. What was not observed was a perfect grouping. However, if two or more elements of data were similar they were put in the same group. This helped the researcher to summarize the data. For example, in this research we found out that group A and B should be studied against group E and F while others did not match with the rest of the participants and therefore could be excluded from the study.

The second question that was answered was that both K-means and DBSCAN were reliable enough to be used for this kind of research.

Two issues need further consideration. The first issue is related to the similar clusters themselves. We need to find out what makes A and B vs. E and F so similar that each of the algorithms puts them in the same group. If this is discovered, the validity of these groupings will be accepted more. The second issue relates to the participants who are placed in different groups each time. It should be explored as to what makes these participants similar to some participants in one execution of the clustering and different in another execution of the clustering. However, this is not related to programming research but is related to performance difference of clustering algorithms.

At this point we can summarize our data and choose the right participants for further study, which involves the understanding of the process of program comprehension of different programmers. From this point qualitative data analysis starts. One opportunity for further work is to investigate the feasibility of analysing the codes that are clustered by those algorithms automatically. If this happens, the process of analysis of these kinds of research becomes very simple.

As mentioned earlier, the two clustering algorithms that were chosen for this experiment are just an example of centre-based and density-based algorithms. Other algorithms, including hierarchical clustering algorithms, can also be tested for different applications in program comprehension research.

6. ACKNOWLEDGMENTS
This research was supported by Shiraz University of Technology, grant number 90-EE-1.

7. REFERENCES


