

Evaluating Assessment with Competency Mapping

Robyn A. McNamara

School of Computer Science and Software Engineering,
Monash University,
Victoria 3800,
Australia.

Email: ram@csse.monash.edu.au

Abstract

How do we know what students get out of an introductory computer programming course? Computer programming skills are notoriously difficult to assess fairly and economically. This paper presents a method — competency mapping — which extracts a better and more detailed picture of what students actually learn, without requiring them to sit extra assessment activities. Competency mapping was applied to the set of marks from an introductory course in computer programming.

Insights were gained into the way students learn computer programming. Guidelines for designing assessment to work better with competency mapping are also provided, as well as a design for a dedicated competency mapping tool.

1 Introduction

As educators, we want to assure ourselves that the students have constructed a conceptual picture of the subject matter that is not too dissimilar to the one we have been trying to present to them. Indeed, in the act of assessment we define what constitutes a “sufficiently similar” construction. However, current assessment tools are a blunt instrument: after the assessment is over, what information is available to the course leader? Usually, only a few numbers: the pass rate, the mean, the standard deviation. Although some qualitative information about the students’ progress may be available if the course leader asks the classroom teachers, this information is subjective and usually based on only a small subset of the student body. It is not really possible to extract a conceptual picture from the information that is made available, and yet the marking process usually generates reams of data — data that, although possibly fruitful, is abstracted into just a few numbers.

1.1 Concept mapping

Concept mapping, developed by Novak (Novak & Gowin 1984) in the 1960s, is one way to communicate these sorts of mental constructions. A concept map is a diagram that shows ideas, represented by boxes, and their relationships, represented by arrows. These maps could be elicited from students at the end of the course and compared to an ‘ideal’ map drawn

Copyright ©2004, Australian Computer Society, Inc. This paper appeared at Sixth Australasian Computing Education Conference (ACE2004), Dunedin, New Zealand. Conferences in Research and Practice in Information Technology, Vol. 30. Raymond Lister and Alison Young, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

by the course designer. However, there are several problems with this approach.

First of all, it is likely that students would resist the exercise. To do a concept map properly can be time-consuming, and the mapping would have to be done close to the end of semester, when students tend to have exams and assignments to worry about. The tendency of students to deprioritize tasks that are not summative when under time pressure is well-known (Thomson & Falchikov 1998), and it is difficult to see how the concept mapping exercise could be made summative without jeopardizing its evaluative merits — the point of the exercise should be to obtain an accurate picture of the student’s construction of the subject.

Even if the students can be persuaded that such an exercise is worth their time, the analysis of the resultant data would be difficult. More than two hundred students per year take the course studied in this paper, and just reading all those concept maps would require a significant investment in staff hours. Furthermore, collating the data from hand-drawn, free-form concept maps would be difficult. What is needed is a method for automating the process of concept map generation.

Trochim (1989) describes such a method for analysing a set of ideas. This method has been used in institutional planning and analysis to help define the structure and goals of institutional programs. Stakeholders in the program, such as clients, management and staff, meet and brainstorm a set of ideas that are relevant to the institution under analysis. These ideas are written on cards, and each individual groups these cards into piles of related ideas. A correlation matrix is derived from this grouping information, and cluster analysis is applied to the correlation matrix to determine which ideas are related. Multidimensional scaling is then used to generate a two-dimensional picture of the conceptual structure of the entity under analysis.

This method would obviate the need for labour-intensive analysis of the final result, but would still not be feasible in practice due to the size of the stakeholder group. Moreover, it would be desirable to generate this map without adding to the burden of assessment already weighing upon the student body, and without having to work any extra activities into these students’ schedules.

1.2 Competency mapping

Over the duration of any programming course, a lot of data is collected about student performance: prac marks, assignment marks, test results, and exam results. If it is assumed that tasks that test similar abilities will display correlation in their marks, then it should be possible to develop a correlation matrix from these marks. From there, cluster analysis and

multidimensional scaling can be applied to produce a two-dimensional map.

Competency mapping originated as an attempt to empirically determine the cognitive structure underlying an introductory programming course. This was based on the assumption that assessment tasks that are testing related concepts should show a positive correlation in the marks students achieve for them, and that stronger correlations should result from more strongly related concepts. This assumption did not seem controversial at the time, but when our first correlation matrix showed weaker than expected correlations between laboratory prac marks and programming questions from the exam, it was realised that the unstated assumption — that assessment marks directly reflect a student's grasp of a concept — was false. The mode of assessment may be having a more significant impact on student marks than we had thought. The name "competency mapping" reflects the fact that concepts are not assessed directly, but rather through the intermediary of some practical skill.

2 Method

We applied competency mapping to the Semester 1, 2001 results from CSE1301 Introduction to Computer Programming, which is the first subject students undertake in the Bachelor of Computer Science and Bachelor of Software Engineering courses. Students undertaking this course in 2001 were required to undertake 14 distinct summative assessment tasks:

- 12 weekly practical exercises, which were supervised and marked by the lab demonstrators. This component was worth 30% of the final subject mark. Six of these pracs also had a bonus component, completion of which could push the student's mark for that prac over 100%.
- A one-hour mid-semester test, which was worth 10% of the final mark.
- A three-hour, 28-page final exam, which was worth 60% of the final mark.

As well as undertaking this summative assessment, students were also encouraged to attend weekly one-hour tutorials. Because these tutorials were not marked other than by recording attendance, they were not included in this study; they are highly recommended, but they are not compulsory.

2.1 Data selection

In order to maximize our chances of extracting useful information from the cluster analysis, we needed to maximize the number of marks we collected. Therefore, data was selected on the basis of availability. All pracs were included in the analysis, although the bonus marks — which were recorded separately — were omitted because very few students attempted them. This gave us 12 data points per student.

The prac results were available in electronic form, but only the final mark was available electronically for the exam. This would not have given sufficient granularity to demonstrate the effect of competencies on the marks, so the papers were sent for and the marks for each page were keyed in. The mid-semester test was not used, because keying the marks was very time-consuming. The prac data and the exam data, taken together, gave us 40 data points per student.

The exam included four pages of short-answer questions at six per page, and ten pages of multiple-choice questions at four per page. The questions on

each page were not related, so it was not likely that cluster analyses would be able to extract any useful data from them — after all, if each short-answer question were only testing two competencies, each page of short-answer questions would be testing twelve. Total marks for these sections were calculated, and these were used instead. That still left us with 28 data points per student.

Only those students who had both completed the pracs and sat the final exam were counted. Of the 454 students who are recorded in the prac database, only 362 sat the exam. Many of the students dropped out during the semester, and it was felt that including their marks would skew the data. Of course, not all students who did not sit the exam had dropped out: many sat a deferred exam, but because the questions on that exam were not the same, these students' marks were not commensurate anyway. After filtering the students in this way, we were left with 350 students.

The remaining prac data was also subjected to cleaning. Students who were marked 'absent' for a prac had 0 entered. Prac marks of 'sick' or 'exempt' were also translated to zero, even though students are actually awarded their average prac marks for these pracs — in this case, the mark given does not reflect mastery of a competency and it is therefore not suitable for inclusion in the study. This replacement affected 205 of the 454 students in the prac database, most of whom had been marked 'absent'.

The data were tabulated, with each row representing a student and each column representing an assessment activity. The raw numeric marks were then converted into percentages, to ensure that the data was commensurate. The resulting file was imported into SPSS for statistical analysis.

2.2 Factor analysis

Before cluster analysis could be performed, the number of clusters needed to be determined. Factor analysis was performed on the data, and two heuristics deriving from the results of this analysis were used to determine the clustering: the scree test and the Kaiser criterion.

In factor analysis, factors are calculated in such a way as to maximize the amount of variance accounted for at each step until all variance in the initial data has been accounted for. The amount of variance extracted in a single step is called the *eigenvalue*, and plotting the eigenvalues at each stage in the factor analysis gives a scree plot, named for its resemblance to a cliff with a scree slope at the bottom. The scree plot usually has a steep initial descent as major factors are extracted, followed by a section of lesser gradient representing minor factors, and finally flattening out as "noise" or random factors are extracted. Factors in this flat section contribute little to the data.

For the scree test, the graph is examined visually to see where the point of diminishing returns lies: the point at which the graph starts to flatten. The scree plot for the main data set is shown at Figure 1, and the three sections can be seen clearly here. The scree test estimate for the number of clusters based on this plot is three. This method tends to underestimate the true number of factors, so the Kaiser criterion was also used.

The Kaiser criterion is simply the number of factors with eigenvalues greater than 1. This tends to overestimate the true number of factors. In cases where the Kaiser criterion and the scree test gave divergent estimates, plots were generated for each level of clustering between that given by the Kaiser es-

All students, MCQ and SA combined, excluding bonuses

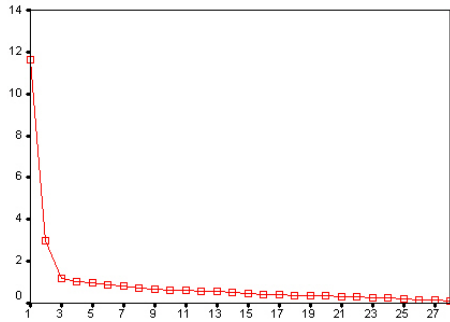


Figure 1: Scree plot

timate and the scree test estimate and the version which made the most sense was chosen.

2.3 Cluster analysis and multidimensional scaling

Once the degree of clustering had been decided, hierarchical factor analysis was performed on each dataset, and multidimensional scaling was used to map the data into two dimensions. Multidimensional scaling is a method by which variables are assigned to points in two or more dimensions in such a way that variables that have a higher measured similarity — in this case, marks with a higher degree of measured correlation — are closer together. In essence, it is an optimization problem: the stress, or “badness-of-fit”, is minimized over several iterations.

It is important to note that the cluster analysis used the cleaned raw data. If we had applied cluster analysis to the output of the multidimensional scaling, points that were spatially close would have been clustered rather than points that were strongly correlated. Multidimensional scaling seldom gives a perfect correspondance between coefficient of correlation and spatial distance, so it is possible for variables to be plotted closer together than their actual degree of correlation would indicate. However, as can be seen in Figures 2 – 4, the results of the cluster analysis and the multidimensional scaling corresponded well.

2.4 Visualisation

SPSS can be made to produce graphs of the output of multidimensional scaling, but they are difficult to read when they contain as many points as our datasets had. It was necessary to find an alternative method for displaying the data.

For presentation, the points were split up by cluster and fed into gnuplot. This produced results that were good enough to display on an overhead projector, but they do not reproduce well on paper.

2.5 How easy is competency mapping?

Any person who is comfortable with the use of statistical software packages can apply competency mapping very easily. The bulk of the work involved is mechanical: acquiring marks data and converting files between formats. Almost all of this work would be easy to automate, and it is envisaged that, once data has been gathered, lecturers should be able to generate and view competency maps on the fly using software written for the purpose.

Not all course units will lend themselves easily to competency mapping. Because the technique is based on a correlation matrix, the sample size needs to be large enough for statistically significant correlations.

It is not yet known exactly how large that sample size will need to be. Furthermore, the assessment needs to be fine-grained to be of maximum use: it is difficult to discern correlative patterns in the data when there are few variables. However, competency mapping can be applied easily to any course whose population and assessment characteristics are sufficient, including courses in other disciplines.

3 Results and analysis

Our assumption that correlations between questions would reflect similarity of subject matter had led us to expect to see tasks clustered by programming concept — for example, pointers, abstraction, or algorithms. Instead, we saw clustering based on mode of assessment.

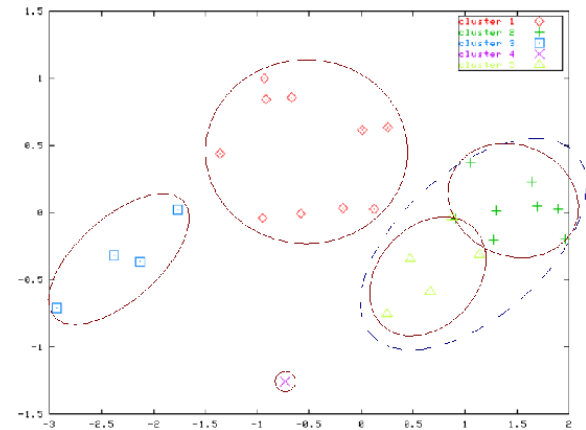


Figure 2: Competency map, all students

Figure 2 shows the competency map obtained from the whole class. Cluster 1, marked by diamonds, includes all the exam questions except for the ones on digital logic, linked lists, test data, the two sorting questions, and the largest programming question. Cluster 2, marked by plus signs, contains prac 1 – 4, 10 and 11, and the exam questions on digital logic and test data. Cluster 3, marked by squares, contains the exam questions on linked lists and bubble-sort, the largest programming question on the exam, and prac 12. Cluster 4, marked with an X, is a singleton, containing only the exam question on selection sort; while the remaining pracs 5–9 are all in cluster 5 and marked by triangles.

Clusters 2 and 5 form a “supercluster”. When the data is clustered at level 4, these clusters merge to form a single, large cluster containing 11 of the 12 pracs and 2 of the 28 exam questions.

It is interesting to see the questions on digital logic and test data clustering with the practical tasks. This allows the possibility that students who are good at testing and deductive logic will also be good at programming. On the other hand, the result could be interpreted to mean that only students who are good enough at programming to complete their programming tasks get any practice at developing sets of test data.

Figure 3 shows the competency map derived from the results of the top third of the student body by raw mark. Cluster 1 is the group of three, which contains the multiple-choice and short-answer exam pages, and the exam question on debugging. Cluster 2, marked by plus signs, contains prac 1–12, the exam questions on digital logic, test data and data structures, and the two smaller programming questions from the exam. All other questions were singletons.

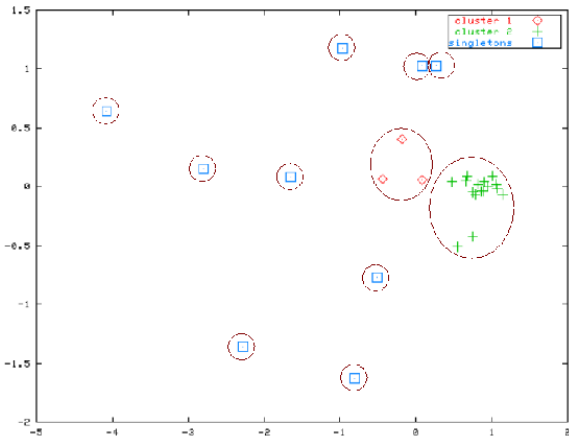


Figure 3: Competency map, top 33% of students

It is only in the results from the top third of the class that we can see programming questions from the exam clustered with the pracs. It seems that the higher-achieving students are doing what we expected, but that everybody else seems to be doing something else. Evidently, the top third of students apply similar sets of analytical skills to programming in an exam context and programming in a practical context. This is unsurprising; what is surprising is that it seems that the rest of the student body does not.

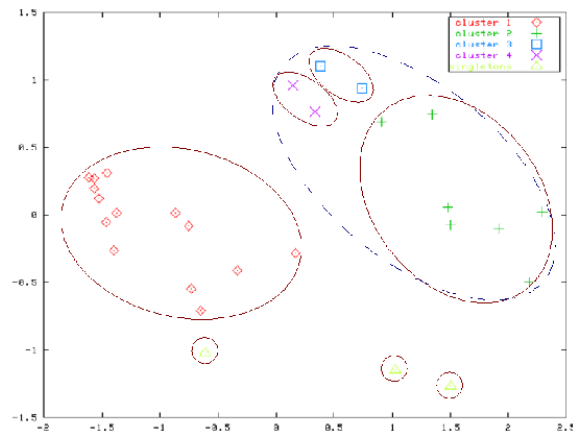


Figure 4: Competency map, bottom 33% of students

Figure 4, which shows the competency map given by the results of the bottom third of the students by raw mark, is starkly different. Cluster 1, the large cluster on the left, contains all the exam questions apart from those on digital logic, test data and selection sort, which are singletons; it also contains prac 12. Cluster 2 is the largest cluster on the right, and contains pracs 1–5, 10 and 11; cluster 3 (squares) contains pracs 6 and 7, and cluster 4 (crosses) contains pracs 8 and 9. At the five-cluster level, clusters 2, 3 and 4 combine to form a supercluster which contains all but one of the pracs, and no other tasks at all.

The differentiation between prac tasks and exam tasks is sharp in this map. It seems that, to these students, practical questions and theory questions have little to do with one another. Perhaps they are not extracting abstract lessons from their practical experience.

Perhaps it should be a little disturbing that the maps derived from the bottom third of students and from the whole group look very similar, but the map derived from the top students looks different. This

could indicate that the “average” student constructs a similar view of the subject to that of the less-capable student: a view that is disjointed and lacking in coherence.

3.1 Random data

How can these graphs be interpreted? Exactly what can be inferred from them? It is a circular problem: the accuracy of the representation of the map cannot be checked against the underlying conceptual structure of the subject, because the underlying conceptual structure (as modified through the students’ learning experiences) is not known with certainty. It is useful to generate dummy data with known properties and see how well competency mapping performs. This exercise will also shed light on the best ways to design assessment for use as input to competency mapping.

In order to generate this dummy data, it was first necessary to design a mathematical model for assessment marks.

3.1.1 A model for competencies

There are two entities that we must model: students and questions. Each question will test one or more competencies, and each competency will contribute a proportion of the total mark for the question between 0 (completely irrelevant) and 1 (marks for that question depend exclusively on this competency). Furthermore, each student will have a certain degree of mastery of each competency, between 0 (no idea at all) and 1 (complete mastery).

Assume that the domain comprises N competencies (roughly equivalent to the clusters on the map). Assume, optimistically, that the clusters are independent and disjoint: they do not overlap.

The set of capabilities of each student can then be represented as a vector of positive numbers S between 0 and 1, where S_i represents the proportion of competency i that the student has mastered. Assessment tasks can be modelled as a stochastic vector of positive numbers Q between 0 and 1, where Q_i represents the contribution of competency i to the question. Now, the probability of a given student getting a given question right is the product of the student’s capability vector and the question’s capability vector. To put it another way, the expected mark is $(Q \cdot S)M$, where M is the total marks available for the question.

Note that, under this model, assessment is an attempt to infer each student’s vector by getting the student to sit a number of tasks with (it is hoped!) known Q vectors. The aim of teaching, of course, is to get all the S_i as close to 1 as possible; more realistically, to get the S_i over some predetermined minimum value. Competency mapping is an attempt to infer the Q vectors.

Of course, this model is an oversimplification. It is unlikely that real domains have independent disjoint competencies. However, it is good enough to be used as a basis for analysis: it is simple, and not hard to calculate.

This model may have uses outside competency mapping, and can provide a basis for any competency-based analysis of student results.

3.1.2 The random datasets

Data was generated according to several different models:

- orthogonal — precisely one element in each Q vector is non-zero. In this model, each question tests a single competency.

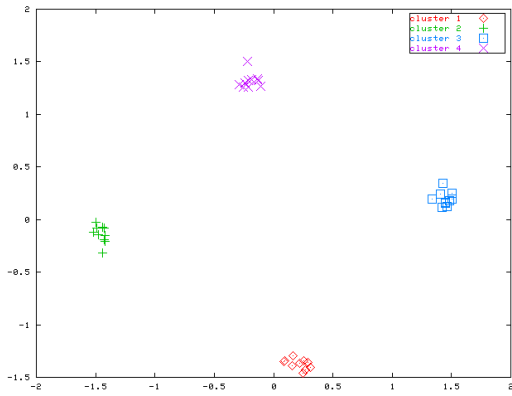


Figure 5: Orthogonal, four competencies

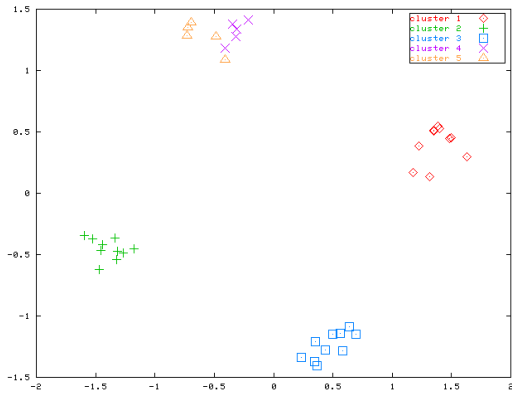


Figure 6: Orthomodal, 0.25/0.25 modal competencies

- mixed — multiple competencies may contribute to each question. This was modelled by beginning with a zero vector and adding $1/N$ to N randomly-chosen elements, for various values of N . The example mixed-model dataset uses $N=3$.
- orthomodal — this is an attempt to model the case in which questions are delivered in two modes which have different cognitive overheads. The Q vectors consist of a mode vector appended to a base competency vector. The mode vector has two elements, of which exactly one will be non-zero; this models the mode of delivery. For this dataset, the base competency vector was orthogonal.

Student competency vectors were generated according to a normal distribution, with identical mean and standard deviation for each student and competency. Note that this does not imply that the student competency vectors are identical: only that they were randomly generated by the same function.

The following datasets were chosen for inclusion:

- O4-40 — orthogonal, four competencies, 40 questions
- OM4-40 — orthomodal, four base competencies, two modal competencies at 0.25, 40 questions
- OMA4-40 — orthomodal, four base competencies, one modal competency at 0.25 and one at 0.75, 40 questions
- M4-3 — mixed, four competencies, three competencies per question

3.1.3 Performance

Competency mapping performed predictably well on the orthogonal datasets. In these datasets, each question assessed only one competency. As can be seen from the example, at figure 5, they produced tight, well-separated clusters that are easy to distinguish. They do not resemble the results of competency mapping on actual student data, but this is unsurprising: student assessment is far from orthogonal. However, it does imply that questions that are as close as practical to orthogonal might produce competency maps that are easier to read and interpret.

The mixed datasets produced the plots that looked most like those obtained from real data. (See figure 8.) Competency mapping did a good job of clustering the questions according to the strongest base competency, but the clusters tend to overlap more than the clusters drawn from real student data do. This is at least partly due to the higher stresses in the multidimensional scaling: when an 80-question mixed dataset with eight base competencies was plotted, the stress in the scaling was over 0.4. In comparison, the stresses for multidimensional scaling in the real student data were in the neighbourhood of 0.1. The reason for this poor fit is not known at this stage. The example shown at figure 8 had a stress value of roughly 0.18.

The competency map for the orthomodal dataset with both modal competencies set to 0.25, shown at figure 6, looks very similar to the map for the orthogonal dataset. The main difference is that the clusters are less tight. However, when one modal competency is set at 0.75 — modelling a mode in which delivery and interface issues dominate student performance — and the other remains at 0.25, the results, shown at figure 7, are quite different. While the tasks for the lighter mode still cluster by base competency, the tasks for the heavier mode cluster together most strongly. These “mode A” tasks, which were assigned to cluster 5 in the example figure, form a single, relatively diffuse cluster. This resembles the clustering of prac tasks in the real data, which opens up the possibility that issues associated with the delivery of pracs, rather than familiarity with essential concepts or mastery of basic skills, may be the dominating factor in determining prac marks.

Other interpretations of the data are possible, of course. For example, it is possible that the pracs are measuring programming ability, but that the exam is not: in other words, the ability to program may be acting as a modal competency in that it is a determiner of marks in one mode but not in the other. But the exam contained programming questions as well as theory questions! If this is the reason for the divergence in correlations, then it must take a substantially different skill set to write a good program under exam conditions rather than in a lab class.

4 Problems

This analysis has not been without problems. First of all, the format and layout of the final exam was not well-suited to competency mapping. The large number of short-answer and multiple-choice questions on those pages of the exam gave us insufficient granularity of measurement to extract a true map of the fundamental competencies in introductory computer programming. A question-by-question breakdown of the marks would have been ideal, but to collect that data from the written exams would have been difficult, time-consuming and error-prone.

Furthermore, the last three pracs were run at the same time as a three-week assignment that was to

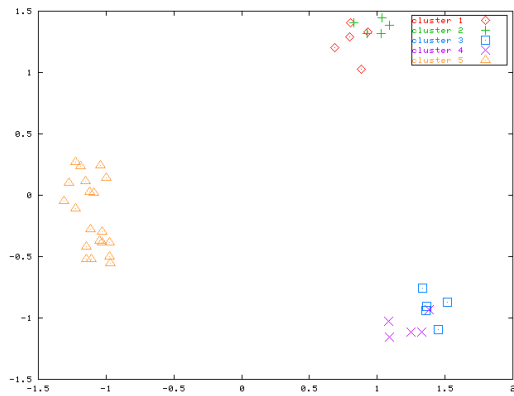


Figure 7: Orthomodal, 0.75/0.25 modal competencies

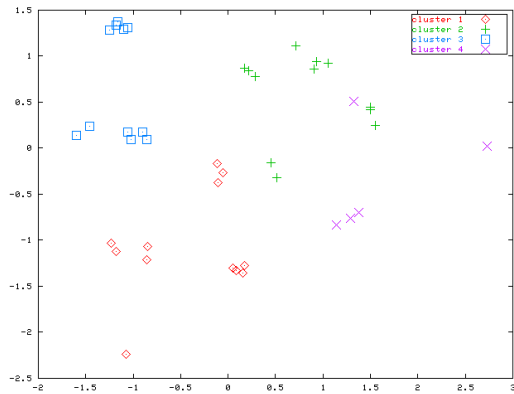


Figure 8: Mixed, 4 competencies, 3 per question

be completed outside prac time, but marked by the demonstrator. Some demonstrators evidently combined all assignment marks with the prac marks for week 12; others marked part of the assignment in earlier weeks and combined the marks with those pracs. This renders the measured data for pracs 10, 11 and especially 12 suspect.

Neither of these problems are insurmountable. If the course is run with a view to competency mapping, the final exam can be written and possibly delivered in such a way as to maximize its potential benefit, and the assignment marks can be separated from the prac marks.

The use of SPSS for statistical analysis narrowed the range of algorithms that could be used, although it was adequate for a preliminary study. More work needs to be done on methods for cluster analysis. In particular, the level of clustering was usually decided on an ad-hoc basis.

5 Future work

Competency mapping has the potential to provide useful feedback to programming teachers, but to fully realize that potential a tool needs to be written to simplify its application. Such a tool would need to be modular in design, at least while the competency mapping technique is still under development, so that different algorithms for cluster analysis and multidimensional scaling can be tested. A possible architecture for such a tool is shown at Figure 9.

To aid in the electronic collection of data, and to ameliorate some of the problems caused by the layout of the exam, some of the written assessment could be replaced by an on-line test. This would allow the collection of separate marks even for the multiple-choice questions, and would also allow the automation of marking for these questions. Of course, such testing

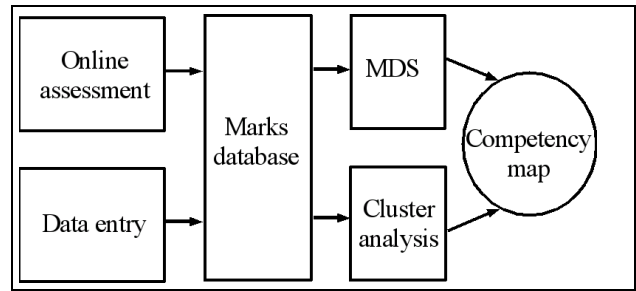


Figure 9: Architecture for a competency mapping tool

will introduce modal competencies of its own; therefore care would have to be taken to ensure low cognitive overhead in the method of delivery. Questions on any exam designed with competency mapping should be small and orthogonal.

It is the potential application of competency mapping that hold the most interesting prospects for future exploration. For example, it has been noted that students who attend tutorials perform better than students who do not (Hagan & Sheard 1998). If competency mapping is applied to students grouped by tutorial attendance, the differences in the competency maps derived from the two groups may point at the topics that provide the most benefit for tutorial coverage.

Similarly, competency mapping may determine whether male and female students actually conceptualize the subject matter differently, or whether the differences observed in their marks derive from other factors such as preferential marking. Social factors such as discomfort with the atmosphere in prac classes will show up as modal factors.

Comparisons of concept maps derived from native English speakers against those derived from students of non-English speaking background may also prove fruitful. As the higher education sector increases in its importance as an Australian export market, tools that can confirm that students of non-English speaking background are being serviced adequately and treated fairly will increase in importance.

6 Conclusion

If competency mapping cannot discern patterns in marks data, what does that imply? It could be that students vary so widely in their understanding that no pattern exists when taken over a large sample. If this is the case, then the technique may still show interesting results on smaller samples taken from subsets of the student population. For example, if the population is divided by gender, it may reveal any systematic differences between male and female constructions of programming — or at least point the way to fruitful areas for in-depth research. Similarly, it could reveal any biasing effects of assessment methods on students who do not speak the language of instruction natively.

On the other hand, if it is not possible to discern patterns in marks data, it could mean that the assessment is not providing enough information for such patterns to manifest themselves. In this case, can we be sure that the tasks set are sufficient for an adequate assessment of an individual? If noise factors in the data are drowning out the signal, then surely the assessment loses confidence.

Competency mapping is a novel technique, and much work needs to be done if it is to achieve its full potential as an assessment evaluation tool. However, that potential is great.

7 References

References

- Hagan, Dianne & Sheard, Judy (1998), The value of discussion classes for teaching introductory programming, *ITiCSE 1996* pp. 108–111.
- Novak, Joseph D. & Gowin, D. (1984), *Learning how to learn*, Cambridge University Press, Cambridge, England.
- Thomson, Karen & Falchikov, Nancy. (1998), Full on until the sun comes out: the effects of assessment on student approaches to studying, *Journal of Assessment and Evaluation in Higher Education* **23**(4), pp. 379–390.
- Trochim, William. (1989), An introduction to concept mapping for planning and evaluation, *Evaluation and Program Planning* **12**(1), pp. 1–16.