DIAGRAM MATCHING FOR HUMAN-COMPUTER COLLABORATIVE ASSESSMENT

Christos Tselonis, John Sargeant,
& Mary McGee Wood

Diagram Matching for Human-Computer Collaborative Assessment

Christos Tselonis, John Sargeant & Mary McGee Wood
School of Computer Science, University of Manchester
{tselonic, johns, mary}@cs.man.ac.uk

Abstract

Diagrams are an important part of many assessments. When diagrams consisting of boxes joined by connectors are drawn on a computer, the resulting structures can be matched against each other to determine similarity. This paper discusses ways of doing such matching, and its application in the context of human-computer collaborative assessment. Results show that a simple heuristic process is effective in finding similarities in such diagrams. The practical usefulness of this varies in different contexts, as students often produce remarkably dissimilar diagrams.

Introduction

The Assess By Computer (ABC) project (formerly Assess By Wire) [1] follows a human-computer collaborative (HCC) approach to assessment. We focus on constructed answers such as text and diagrams rather than answers requiring mere selection between alternatives. The HCC assessment process is an active collaboration between humans and a software system, where the software does the routine work and the humans make the important judgements.

In this paper we focus on technology to support HCC marking of diagrams. The basic idea is that many types of diagrams (e.g. UML class diagrams, entity-relationship diagrams, chemical molecules, electronic circuits) can be represented as boxes joined by connectors in a structure known in Computer Science as a *graph*. Graphs can be matched against each other for similarity, and the results used in various ways. This idea was invented independently by groups at the Open University [2] and the University of Nottingham [3], as well as at Manchester.

In our context we do not expect to use the results for fully automatic summative marking. The aim is to present the information efficiently to the human marker, filter out diagrams which really are identical, and improve the speed and quality of the marking process significantly. We have already shown that a similar approach works well for free text, even without applying Natural Language Engineering techniques, although use of such techniques is being investigated [8]. There are also interesting formative applications.

In the following section we explain how students draw diagrams and how diagram questions are set up. Then we discuss the application of classical graph theory, particularly graph isomorphism, to the problem. The limitations of this approach lead to the development of a simple heuristic method which is shown to be very effective. In the HCC context, user interface issues are important, and we discuss these. Finally we outline the current directions of the work and draw some conclusions.

Diagram drawing tools

The ABC system includes a tool to support the drawing of diagrams. The tool consists of a palette and a drawing area. Students can select boxes from the palette, place them anywhere on the drawing area and use connectors to join them. Both boxes and connectors can be labelled, using both standard keyboard characters and a range of additional ones.

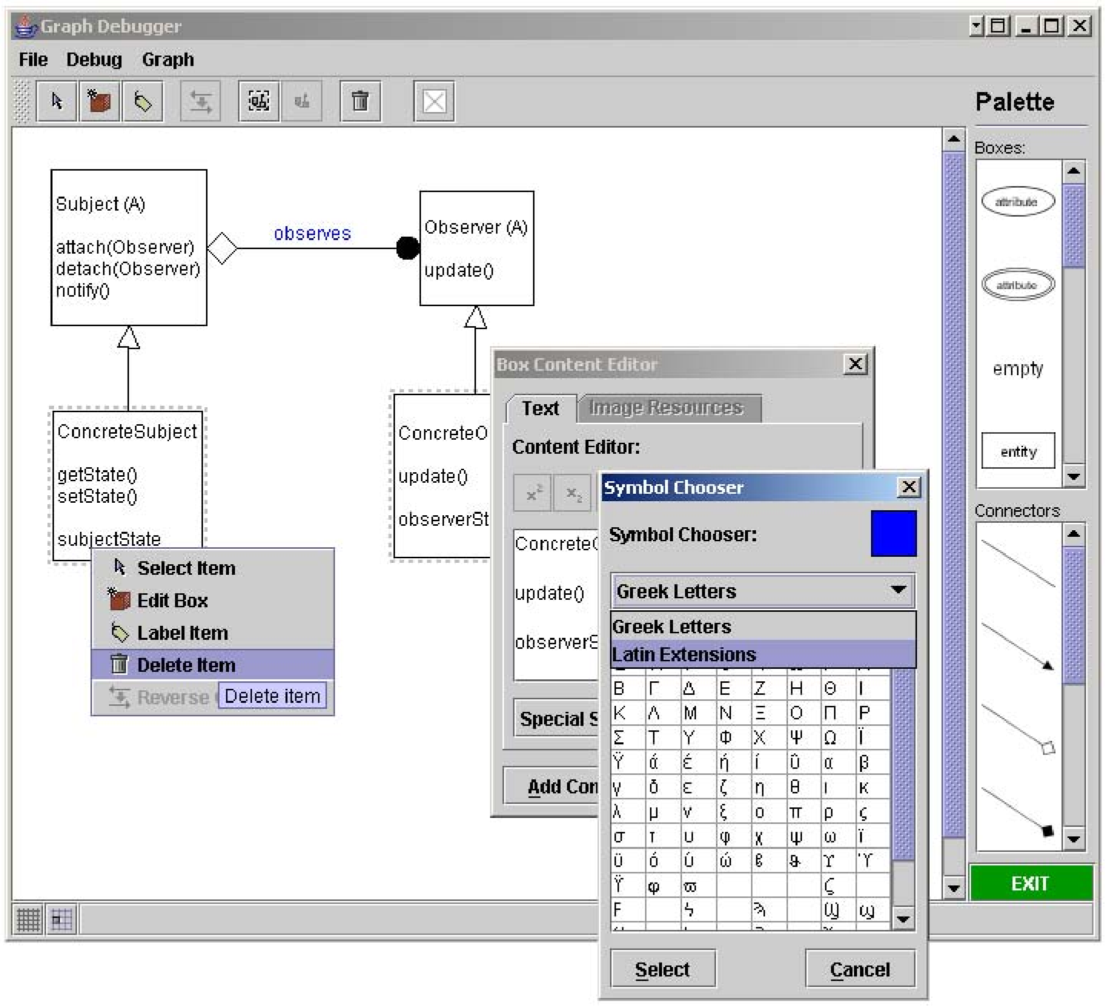
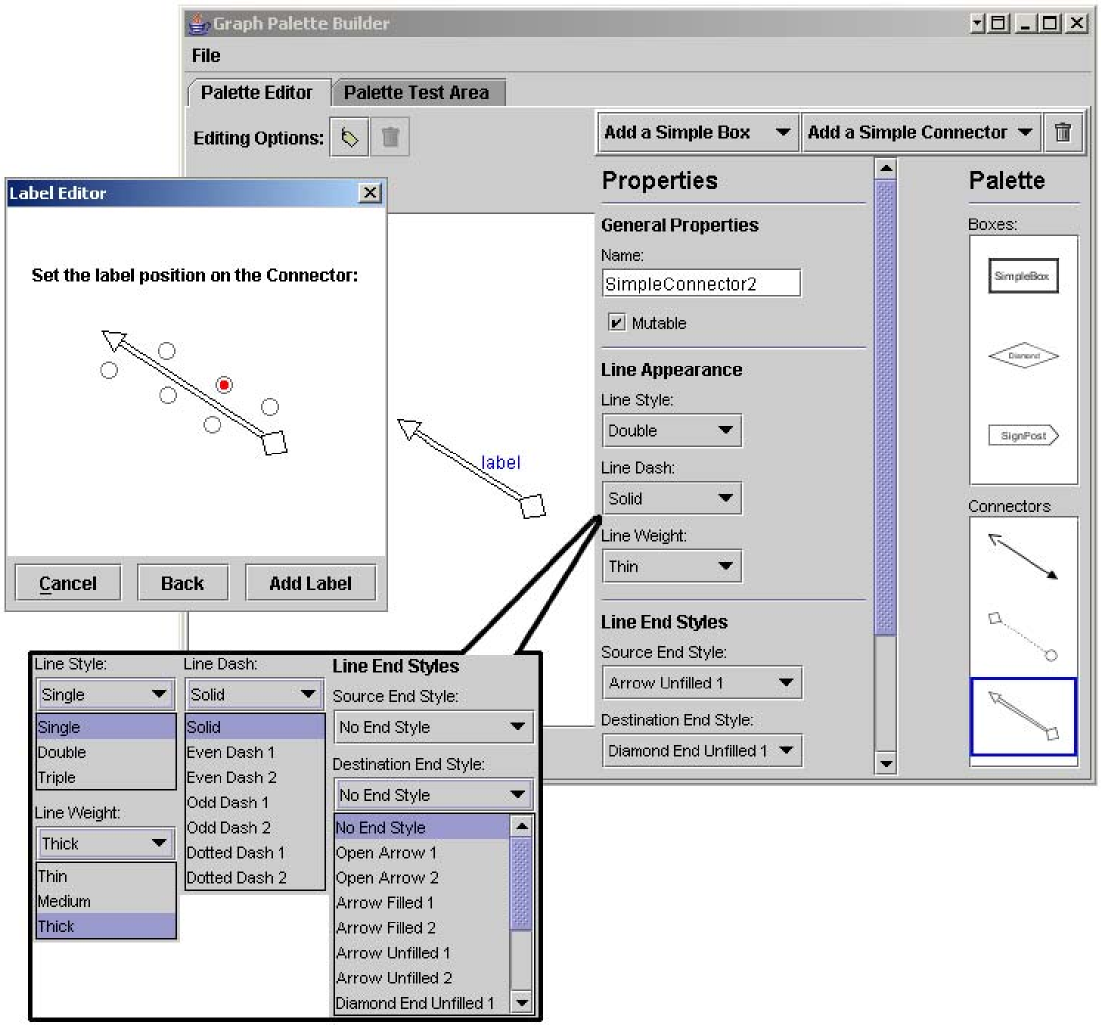


Figure 1. The diagram editor

The diagram type is independent of the system's infrastructure and treated in a uniform manner by it. The palette is fully customisable during examination setting, depending on the kind of diagram the answer is expected to be. Students may therefore be presented with only the components necessary for the domain of the answer, or otherwise, at the examiner's discretion. Figure 2 shows that a much wider range of components is available to the exam setter than would normally be presented to the student.



General Properties
Name: SimpleConnector2
Mutable ✓
Line Appearance
Line Style: Double
Line Dash: Solid
Line Weight: Thin
Line End Styles
Source End Style: Arrow Unfilled 1
Destination End Style: Diamond End Unfilled 1

Line Style: Single
Line Dash: Solid
Line Weight: Thick
Line End Styles
Source End Style: No End Style
Destination End Style: No End Style

Figure 2. The palette builder

The ABC system uses XML to store structured answers. The XML description corresponds to Java data structures. When the matching tool is run, it initially creates Graph objects for each of the diagrams under consideration, consisting of boxes and connectors. A Graph object can be created out of any kind of box and connector combination, without any knowledge of the domain the diagram belongs to.

Matching by graph isomorphism

In classical graph theory, a [simple] graph is defined as “a set of vertices and a set of edges that connect pairs of distinct vertices (with at most one edge connecting any pair of vertices)” [5, p.7]. Hence our boxes are vertices and our connectors are edges. The creation of internal Graph objects enables the execution of standard graph theory operations and algorithms.

The use of such algorithms was investigated and the use of isomorphism in particular seemed a plausible approach: “Two graphs are isomorphic if we can change the vertex labels on one to make its set of edges identical to the other.” [5, p.10]. In this context, a correct answer would most likely be isomorphic to, or at least contain an isomorphic subgraph of, a model answer.

Code was implemented to test whether two graph objects are isomorphic. If they are, a list of matches of corresponding vertices is returned. If they are not isomorphic, the aim is to identify one of the graphs' maximal isomorphic subgraphs, the vertices of which, when renamed appropriately, would be connected by the largest set of edges in the other graph. In this case, along with the matching vertices, the percentage of the number of edges in the maximal edge set, over the total number of edges is returned. The program can handle both undirected and directed graphs (digraphs); in the latter case the edges have a specified direction. The algorithm requires that all of the graph's vertices lie within the same connected component, that is, there is a path from every vertex to every other vertex in the graph. Figure 3 shows two small isomorphic digraphs.

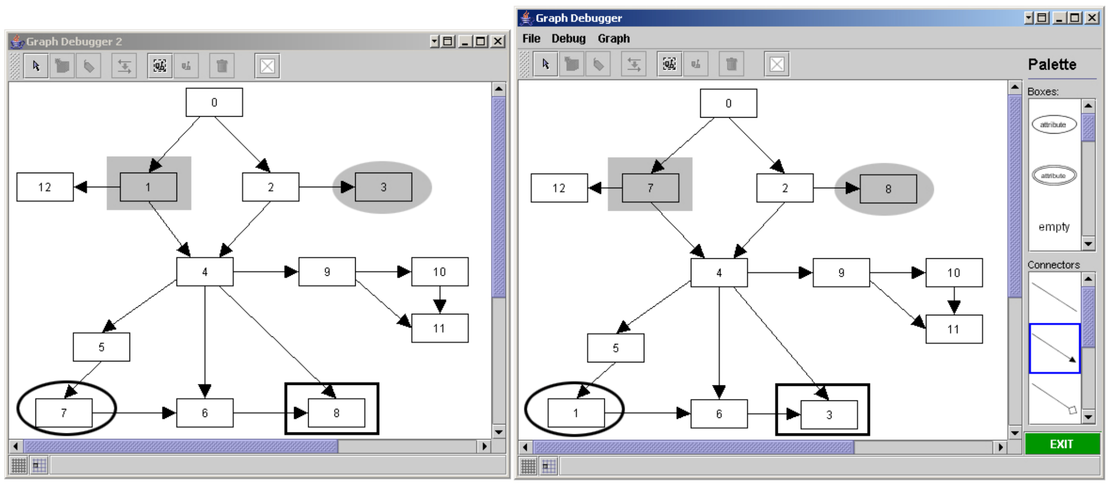


Figure 3. Two isomorphic digraphs

The program recognises that the graphs are isomorphic, and produces a list of the relabellings required. For instance in the example the nodes labelled 1,3, 7 and 8 in the first graph need to be relabelled to 7, 8, 1 and 3 respectively.

The algorithm is recursive, and keeps running until either an isomorphism is found or all possible recursive calls (within the available time and space) are made. In the latter case, the maximal isomorphic subgraphs (the largest pair of subgraphs within the two graphs which are isomorphic) are returned.

This implementation proved to work well even for large, artificial, example graphs (1000 vertices, 100,000 edges for isomorphic graphs and 500 vertices, 5,000 edges for maximal isomorphic subgraph matching). Despite the fact that the isomorphism problem is in principle computationally expensive [5, p.10], in practice the cost is negligible for the size of graphs expected in assessments.

However, isomorphism alone is neither necessary nor sufficient for the purpose of diagram matching in examinations. Diagrams are not *only* graphs: the information to be considered is much richer and components are associated with various individual properties. Such properties include the structure of the diagram's components, their types and labels. Even if two graphs are isomorphic they may be completely different in terms of these characteristics. However, the option of using isomorphism in combination with the heuristics described in the next section is being considered.

A heuristic approach

The method currently being investigated involves the creation of Local Metric (LM) objects for every box of every graph. An LM object contains information such as the type of the box, its degree (number of incident connectors) and its label, the number and types of adjacent boxes and the number and types of incident connectors. The type of a component represents notionally what it is used for within the diagram's domain (e.g. entity, attribute, relation, class, molecule, transition, bond) and visually how it is drawn (frames/line for boxes, end styles/line for connectors). Programmatically however, it is nothing more than a unique string, depending in no way on the diagram's domain. Additionally, an LM object implements methods to accommodate the comparison of its properties to those of another box. Based on such comparisons a score can be generated, expressing the percentage of similarity between the two boxes.

The matching mechanism makes use of these scoring methods and attempts to match every box of a diagram to a box of another diagram. Only the currently unassigned boxes are taken into consideration, so no box is matched to more than one other. Based on the individual box scores, a total score expressing the similarity of the diagrams is calculated. The example in figure 5, based on a 2nd year Computer Science paper examination on databases, illustrates the matching process.

The diagrams show entities (boxes) with attributes (ovals) and relationships (diamonds) and are known as entity-relationship (E-R) diagrams. The two diagrams shown in Figure 5 are similar, but they also have some basic differences. The idea is to compare every box in the first diagram against every box in the second that has not been already assigned to a box in the former, and generate a score. Of all the scores calculated, the highest one would suggest a match.

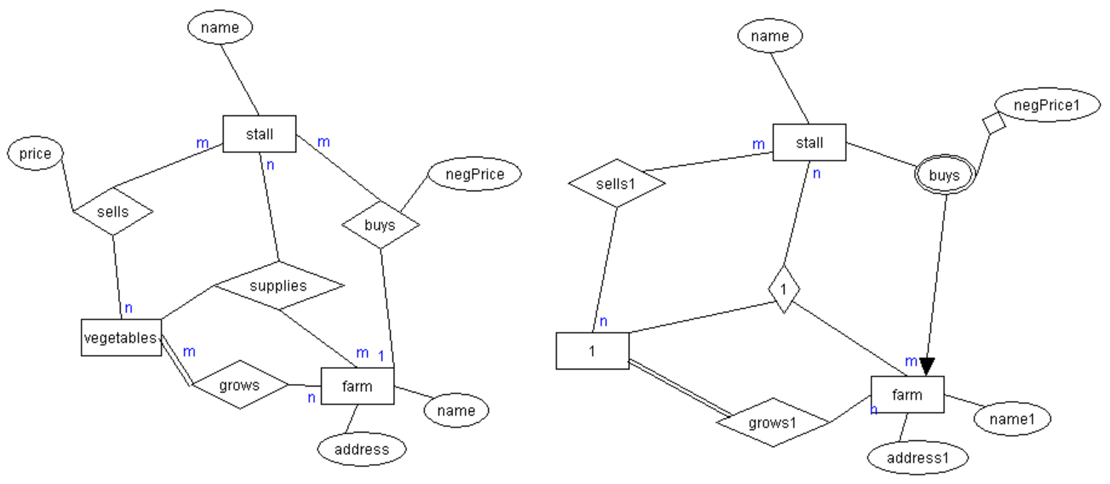


Figure 5. Diagrams to be matched

When comparing two boxes, partial scores are calculated to express their similarity, considering one of their properties at a time. The total score is the weighted average of all the partial scores. Table 1 summarises the currently available weighted metrics along with the methods used to generate the partial scores. More such metrics are likely to be added in the future, as the project progresses. The calculations for the metrics are given informally, as the details are rather complicated in some cases, and subject to change anyway.

The desire to be able to determine, automatically and dynamically, what weight values will produce the maximum (or minimum, depending on the human marker's preference) possible score led to the recent introduction of a Weight Manager (WM). For every combination of the weights of each one of the metrics in the LM objects, the WM runs a series of tests comparing every (answer) graph against a model (if one is provided) or against every other graph. The average score produced is used to characterise that specific combination of weighted metrics. This way, the weights applied are determined on the fly, and may be different for each set of graphs, allowing for marking flexibility. One of the most obvious reasons to require varying weights is the fact that labels on boxes have a highly variable relevance in matching. Often they have a very strong influence, but for example there are cases where in an E-R diagram, multiple attributes of the same name exist. The diagrams in Figure 5, where two “name” attribute boxes exist in each of them, are a simple example. In such cases, it is desired that the other metrics have more weight, in order to distinguish among instances.

Metric	Description	Partial Score Calculation Method
Degree	Number of incident connectors (≥1)	`Return max score if the number is the same, otherwise 0.`
Type	What this box represents	`Return max score if the type is the same, otherwise 0`
Adjacent Boxes	The number of adjacent boxes and their types	`Return an averaged score based on the number of adjacent boxes which do and don't have matching types.`
Incident Connectors	The number of incident connectors and their types	`Return an averaged score based on the number of incident connectors which do and don't have matching types.`
Label	The label's string	`Return max score if the labels are identical, otherwise return a score based on the edit distance between the labels. Above a cutoff edit distance, return 0.`

Table 1. The weighted metrics

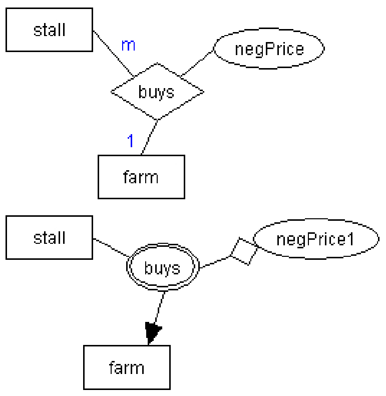


Figure 6. Boxes to be matched

As an example, the boxes labelled "buys" in the previously considered diagrams and their immediate surroundings, shown in Figure 6, are used. Their degrees and labels are the same, as are the number and types of the connected boxes, so these metrics give 100%. The types are different so this metric gives 0. One of the connectors matches, the others don't, so the scoring heuristic gives these a 33% score.

For this example, each metric is given a weight of either 1 or 4, giving a total of 31 combinations ("all 1s" and "all 4s" being equivalent), although metrics can easily be given any number of weights, thus producing a potentially much larger number of combinations. The results from 5 of these cases are shown in table 2. The cases are A: all weights the same; B and C the weights which give the maximum and minimums scores for whole diagram; D and E, the weights which give the maximum and minimum scores just for the "buys" box.

Case	Partial Scores (%)					Total Scores (%)	
	Degree	Type	Adj. Boxes	Inc. Connectors	Label	Box	Diagram
A	100.0	0.0	100.0	33.33	100.0	66.66	78.3
B	400.0	0.0	100.0	33.33	100.0	57.57	**85.18**
C	400.0	0.0	400.0	133.33	400.0	78.43	**76.08**
D	400.0	0.0	400.0	33.33	400.0	**88.09**	77.12
E	100.0	0.0	100.0	133.33	100.0	**39.39**	84.43

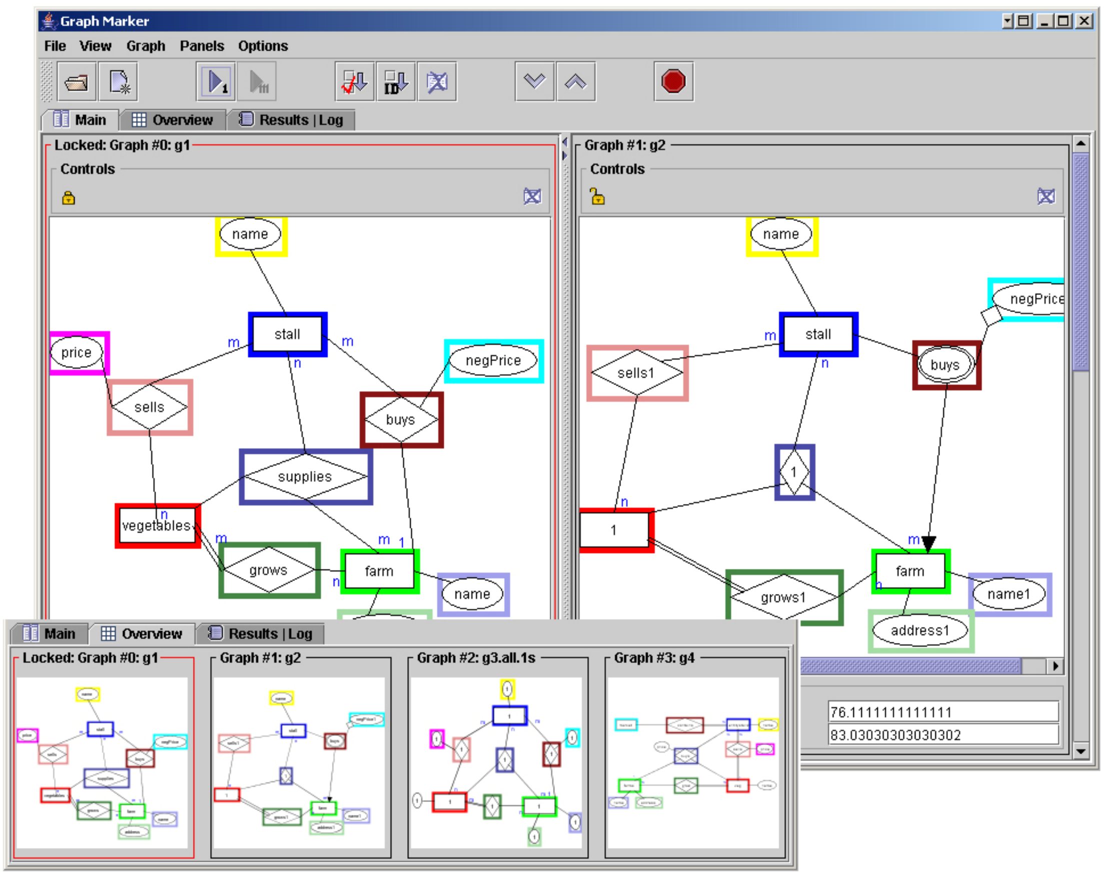
Table 2. Partial and total score test results

Not surprisingly different weights give very different scores for the box. However, the variation over the whole graph is much less – between 76% and 85%, This is typical for cases like this, and suggests that the process is robust: varying the weights does not drastically alter the results. Subjectively, a score of about 80% “seems about right” for these diagrams. However this does *not* mean that the second diagram should be marked as e.g. 8 out of 10: the human must make that judgement (at least in summative situations).

User interfaces issues

One of the system's requirements is that any differences or matches between two graphs should be presented to the user in a clear and useful manner. One aim is to minimise the time taken by, and the cognitive stress involved in, marking. Another is to give useful formative feedback to students automatically. Currently, boxes that have been matched between a model graph and any number of answer graphs (many-to-one matching) are highlighted with the same colour, as shown in Figure 7. This is very useful for research purposes; trials with real users will be necessary to determine whether this is a good presentation format in practice. Many-to-many matching is also supported, producing a sortable table of scores. Both types of matching can make use of the WM to determine on the fly the optimum weights of metrics.

This screenshot demonstrates that although several differences do exist between the two graphs, their boxes are successfully matched. Another pair of diagrams depicting caffeine's chemical molecule (correctly in the second, almost correctly but with an extra triple-bonded carbon atom on the far right in the first) is shown in Figure 8. An interesting feature of the latter example is that there are identical sub-structures in different parts of the graph (an N bonded to three Cs, top right and bottom left). The localised nature of the matching process means that it is not guaranteed to match these “the right way round”, although in the example it does. This suggests that use of a global structure check such as isomorphism in combination with the local heuristics would improve the quality of information presented.



76.1111111111111
83.03030303030302

Figure 7. Diagram matching in action

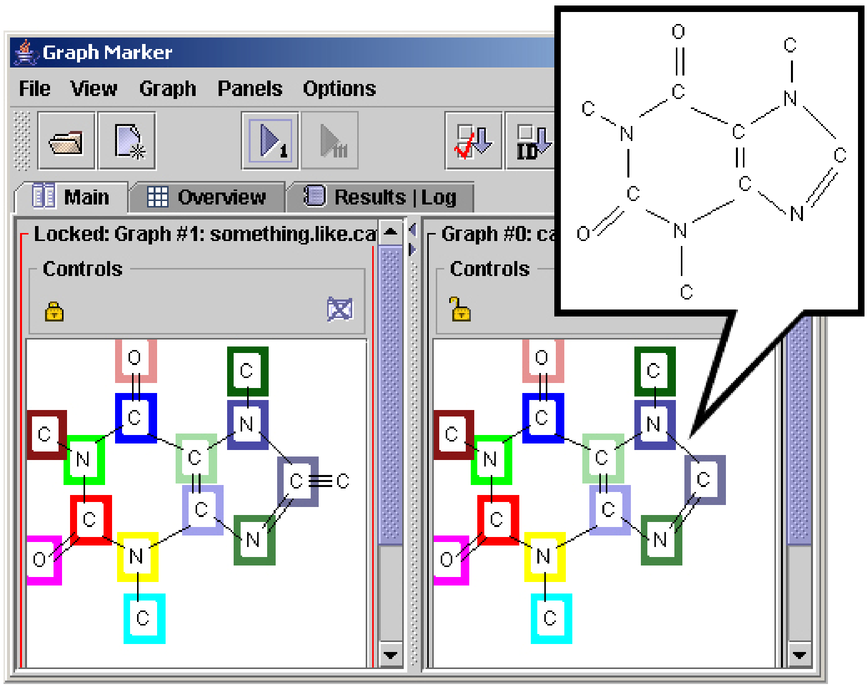

Graph Marker
File View Graph Panels Options
Main Overview Results | Log
Locked: Graph #1: something.like.ca
Controls
Graph #0: ca
Controls
O
C
N
C
C
N
C
C
N
O
C
C≡C

Figure 8. Chemical diagrams being matched

What students really draw

Tests with real exam data produced some unexpected results, however. The real data is remarkably messy, in particular student's answers are often not single connected graphs. Table 3 shows results from an online test where the students had to draw UML diagrams representing a particular pattern. The question constrained the answers much less than for the E-R diagrams.

Out of the 20 students that attempted to answer the question, four of them produced diagrams which could not meaningfully be matched. Intermediate cases also exist where labels characterising a box are attached just above or below the box, and not in it. The system currently considers labelled boxes where the label is in the box, hence diagrams including boxes mislabelled as such received a low score.

	Scores (%)				
Answers	**Using WM***	**Using WM****	**All 1s**	**Human**	**Std. Deviation*****
Ans01	69.55	21.99	48.38	**0**	**49.18**
Ans02	81.00	67.81	73.58	80	**0.707**
Ans03	**94.55**	**78.18**	**88.00**	**100**	3.854
Ans04	76.36	34.56	43.04	80	2.574
Ans05	34.35	14.94	15.29	60	18.137
Ans06	65.13	45.98	55.29	80	10.515
Ans07	**14.55**	**03.64**	**08.00**	**0**	10.288
Ans08	87.88	32.88	73.33	80	5.572
Ans09	84.64	26.09	66.20	40	31.565
Ans10	51.67	43.33	51.33	**0**	36.536
Ans11	69.14	63.31	60.98	60	6.463
Ans12	59.44	50.69	52.86	**100**	28.68
Ans13	81.82	71.08	75.38	80	1.287
Ans14	89.45	57.82	76.80	80	6.682
Ans15	89.09	56.36	76.00	**100**	7.715
Average	**69.90**	**40.51**	**57.63**	**62.67**	**14.65**
*either 1s or 4s (32 combinations, 15x32=480 iterations), best case: degree = labels = connectors = 1, type = boxes = 4					
**either 1s or 4s (32 combinations, 15x32=480 iterations), worst case: labels = connectors = 4, degree = type = boxes = 1					
***Standard Deviation between human marker and WM* scores.					

Table 3. Real examination data results

Nevertheless, there is a reasonable similarity between the system scores and the marks (which were out of 5, scaled to percentages here), although clearly not enough to contemplate automatic marking.

The labels in some of the boxes were practically essays, and dealing with these sensibly is non-trivial. Simple heuristics were used here, but separate work on text similarity detection will need to be incorporated.

Quite a few of the diagrams were awarded a low score by the system because of the components the student placed and their consideration by the system; for example, a “junction box” is practically transparent to the human marker and only serves as a hook for connectors to be drawn forming angles. However, the system currently considers it like any other box, and the addition of an extra box in a graph of four or five makes a significant difference.

Generally, the diagrams considered for this trial were structurally very simple allowing for unrestricted matching (in most diagrams including the model, three out of four or four out of five boxes were of degree 1) and accurate matching is largely dependant on the box labels, the matching of which is not supported optimally yet. However, both the arithmetic and the visual results of figure 8 show that the system extracts useful information, although there is clearly space for improvement.

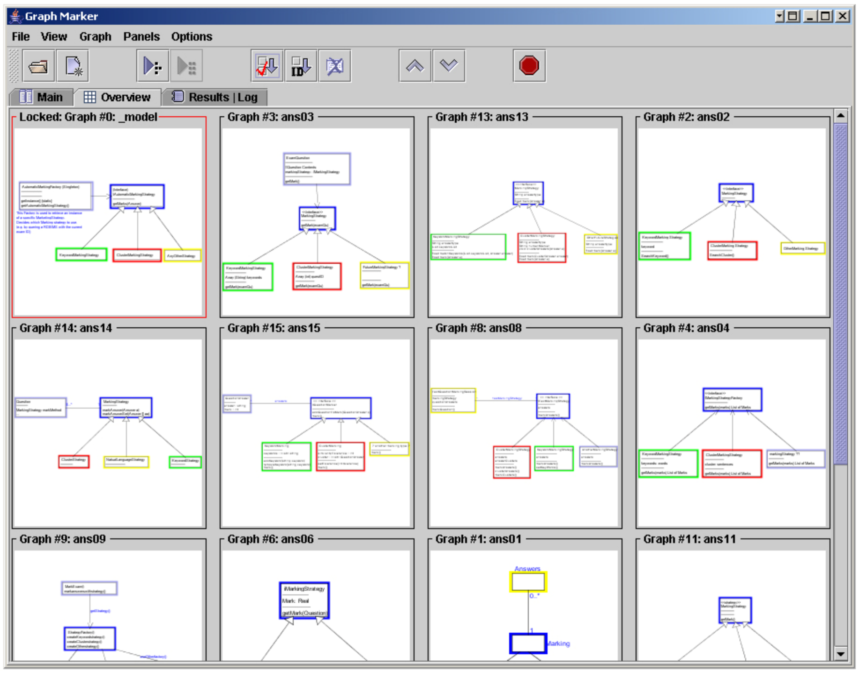


Figure 8. UML diagram matching

Related work

The DEAP system [2] comprises a drawing tool able to handle boxes, connectors and associated text, and encode them into text for storage. The text can be used to reconstruct the diagram, or identify its minimal meaningful units (MMUs), which are then combined into higher level features and compared against those of a model diagram; a grade is eventually awarded based on their similarity. However, domain knowledge is required for this kind of interpretation: “The identification stage uses domain knowledge to identify what we have called minimal meaningful units (MMUs).” [6]. Additionally, there is currently no way of expressing any kind of feedback visually, that is non textually.

The DATsys system [3] provides a diagrammatic framework to CourseMarker [4]. Research is currently being carried out to extend its capabilities into automatic marking. Early work is described in [9], but, probably because of the system being a commercial product, little current information is publicly available.

Conclusions and further work

We have shown that a fairly simple heuristic approach to graph matching can detect the similarities between diagrams effectively. One concern is that it *may* "over-match" – show matches which are not actually valid. So long as the purpose is only to provide input to the human marker, this is not a major problem, but it can in any case be solved by combining the local heuristic method with a global structural check (e.g. isomorphism) and by providing additional information for special cases (e.g. non-commutative operators in parse trees).

The work will be extended in a number of ways. For instance we will investigate combinations of local heuristics and global structure checking, as mentioned above, and incorporate better text matching for labels. For diagrams where spatial relationships are important, and to handle disconnected diagrams, we will probably introduce "pseudo connectors" which represent "near to", "on top of" etc. In the longer term, we plan to investigate ways to deal with hand-drawn input. For instance university-level chemistry diagrams are typically much more complicated than the example above, and are very fiddly to draw with a mouse, so use of a drawing tablet is indicated.

A common theme in all our work is that real student data often contains unexpected complications, and diagrams are no exception. Overall however, we regard the results to date as encouraging.

Acknowledgements

The ABC project is funded by the University of Manchester Distributed Learning Fund. Stuart Anderson constructed the graph drawing tools shown in the first two figures, as well as many other parts of the ABC software. Katherine Getao reviewed several drafts of this paper and her penetrating questions were extremely helpful.

References

1) Sargeant J., Wood M.M., Anderson S.M.: A human-computer collaborative approach to the marking of free text answers. Eighth International Computer Assisted Assessment Conference, Loughborough University, Loughborough, UK, 361-370.

2) DEAP Project, Open University
Available from: http://mcs.open.ac.uk/eap
[Accessed: 25 February 2005]

3) LTR Research Group, University of Nottingham
Available from: http://www.cs.nott.ac.uk/~ceilidh/people.html
[Accessed: 25 February 2005]

4) CourseMarker, University of Nottingham
Available from: http://www.cs.nott.ac.uk/CourseMarker
[Accessed: 25 February 2005]

5) Sedgewick R.Algorithms in Java, Part 5: Graph Algorithms, 3rd Ed., Boston: Addison-Wesley, 2004

6) Thomas P., Drawing diagrams in an online examination. Eighth International Computer Assisted Assessment Conference, Loughborough University, Loughborough, UK, 403-413.

7) Tselonis, C., Enhancements to the diagramming capabilities of a CAA system. MSc dissertation, Department of Computer Science, University of Manchester 2003

8) Wood, MM, Sargeant J, Jones CA., What students really say, paper submitted to the Ninth International Computer Assisted Assessment Conference, Loughborough University, Loughborough, UK

9) Tsintsifas A., A framework for the computer-based assessment of diagram-based coursework, PhD thesis University of Nottingham UK, 2002.