OPEN BOOK EXAMINATIONS IN AI TEACHING: A CASE STUDY

Dimitar Kazakov
Dept. of Computer Science
University of York
York YO105DD
kazakov@cs.york.ac.uk
http://www.cs.york.ac.uk/~kazakov

ABSTRACT

This paper discusses the benefits of using open book examinations in artificial intelligence teaching on the case of a 4th year MEng module, Adaptive and Learning Agents, and the experience gathered over three years in the design of the exam paper.

Keywords

teaching, artificial intelligence, agents, open book assessment.

1. INTRODUCTION

This paper discusses the use of open book examinations in the teaching of the fourth-year, MEng module *Adaptive and Learning Agents* (ALA) at the Department of Computer Science at the University of York. We briefly describe the module contents, and how they are related to the rest of the MEng programme, as well as the AI research in the department. An open source Prolog Multi-Agent Simulator used in the examination is also described. The paper then discusses the examination design and its potential to provide the right balance between the degree of guidance given in the paper and the possibility for students to demonstrate their creativity and knowledge of the subject matter.

2. BACKGROUND

YorkCS is a 5** research-driven department, whose teaching has also been praised (judged "excellent" by HEFCE). Individual and research group-wide interests have their impact on the curriculum. For instance, teaching Prolog and concepts of Logic Programming in the second half of Year 2 facilitates the teaching of several 3rd and 4th year Artificial Intelligence modules (Natural Language Processing, Constraint Programming, Adaptive and Learning Agents). Similarly, teaching the introductory first-year Principles of Programming module using Scheme [3] also provides for the needs of the Functional Programming group, and the 3rd year module on Real Time Systems makes heavy use of Ada, first introduced in ADS (Year 1). Most members of staff will supervise around 4 undergraduate and MSc projects a year. Many of these projects are related to the staff research interests, and some of the above mentioned material.

The department attracts students with strong background despite the nation-wide decline in applications for CS courses after the peak in 2001. One would expect the majority of students to have AAB or comparable A-levels. They will progress to the third and fourth year of their course to take options, which often are directly related to their lecturers' research. A small number among them will produce a final-year project of publishable quality, and may co-author a peer-reviewed research paper with their supervisor [1, 2].

The students' background on arrival is very varied: some have years of programming experience in a range of languages (from microprocessor assembly to VB, C++ and Java), others -- a substantial minority -- will be limited to browsing the WWW and using MS Office. Even the ones who can program may not be able to put their skill in the wider scientific and engineering context of the field.[1]

3. ADAPTIVE AND LEARNING AGENTS MODULE

ALA was taught for the first time in Spring 2004. The module is proving popular, with a steady increase in students' numbers: 11 in 2004, 13 in 2005, and, most recently, 15 in 2006, which ranks it fifth out of 12 options with more than 40% of the

[1]The tutorial on *Computer Science vs Software Engineering* (see [1], Appendix A) gives a certain insight in the students' state of mind at the beginning of their first year.

students taking the ALA option. The module is advertised to students as follows:

> The topic of this module is situated at the intersection between Machine Learning and Agents. The module covers background in both areas followed by a discussion on the methodology of the emerging AI domain of Adaptive and Learning Agents, and a demonstration of its ideas on a few focus topics. Machine learning studies the acquisition of knowledge from examples. The module provides insight into the principles on which machine learning is based, and focusses on two approaches: evolution as used by genetic algorithms, and inductive logic programming as one of the most powerful types of symbolic learning. Agents are a useful abstraction providing a general modelling framework. In software development, they allow for applications structured around autonomous, communicative elements, and provide techniques and algorithms for dealing with interactions in dynamic and open environments. In such environments, adaptation and learning are necessary to achieve optimal performance and make design a tractable task. Natural selection in a population of agents and individual learning will be linked through the evolution of learning bias. The relative merits of learning vs. innate abilities will be discussed. In an isolated agent, machine learning has to deal with issues such as timeliness, and use of shared computational resource. Communication, co-ordination and co-operation/competition become central where multi-agent systems are concerned. Of these, the module will provide a more detailed insight of and focus on two forms of social behaviour: co-operation through altruism and the evolution of language.

The main learning outcomes envisaged are to introduce the notion of agents, familiarise with some of the most commonly used machine learning techniques, become acquainted with the principles of adaptation through evolution and natural selection and inductive logic programming, and, finally, gain hands-on experience with the application of these methods to the development of adaptive and learning agents. There is also a hypothesis testing element in the material in order to justify and understand the experimental designs discussed.[2]

ALA is taught over 4 weeks, and includes 18 one-hour lectures and 8 two-hour computer laboratory hands-on practicals. The practical outline usually provides a very detailed guidance. This guarantees that students will explore the prescribed range of ideas and techniques. They are also requested to provide feedback to the lecturer, and e-mail a mixture of factual results and short analyses or pros and cons arguments for several of the subtasks in a given practical. Tasks of exploratory nature, albeit of limited scope, are present in all practicals.

4. Assessment Role and Design

At present, the way 4th year MEng modules are taught is a result of a trade-off between the "horizontal" teaching in the first three years, when material is taught over one or two terms, and the earlier, purely "vertical" 4th year teaching where students would study one module at a time, usually for a week, and be assessed in the 7 to 10-day gaps between two such blocks of teaching. At the moment, the 4th year timetable is a mixture of the two, with modules taught over the course of 4-5 weeks. Of the 14 MEng assessments in 2005/06 (including an individual and a group project), only one is a closed book assessment, while the rest are open book, spanning several days or even weeks (if Easter holidays are counted).

The 1:3:5:7 weighting scheme used for the annual averages over Years 1-4 means that the contribution of a single 10-credit 4th year module to the student's final degree is 3.5%, which amounts to more than 50% of all Year 1. Also, there are no resits in the final, fourth, year. In that context, a traditional, closed book examination makes a substantial proportion of the overall degree dependent of the student's performance on a single day.[3] Another factor in the choice of a closed or open book assessment is that the latter is better suited to assess practical, hands-on skills, on which ALA and a number of other modules put a special emphasis. Unsurprisingly, the open-book assessment is the format of choice for most modules. A potential issue with all open-book assessments is the possibility of collusion and plagiarism. However, we believe that this danger can be reduced substantially through a suitable design of the assessment, an issue, which will be revisited later in the paper.

The design of the ALA assessment is based on the intention to make it enjoyable, and give it a substantial formative role. The students are expected to spend 22 hours on it, or more than a fifth of all ALA workload, so they should be able to benefit from it beyond the mere assessment of their knowledge of the material already covered in the lectures and practicals.

4.1 ALA 2004 Assessment

In 2003/04, the first year in which the module was offered, the assessment centred on the implementation of a genetic algorithm (GA) and its use for the selection of the three most important arguments in a data set in order to divide the data

[2] With the introduction of other, specialised modules covering these concepts some of this material will be phased out in the coming years.

[3] While there are mechanisms in place to mitigate the impact of extenuating circumstances, these are not always well measurable, and/or of verifiable nature.

into eight clusters with maximal entropy. The main challenges in the assessment were to select an appropriate encoding of the candidate solutions (i.e., the GA chromosomes), implement the entropy-based fitness function, the GA itself (selection and genetic operators), then use a statistical test to compare the algorithm performance for 2 different settings specified in the paper, and, finally, discuss the pros and cons of using a GA for this type of task, and suggest a possible practical use of the overall approach. The marking scheme assigned 60% of the marks for implementing the GA, 30% for the statistical test studying its properties, and the remaining 10% for the subsequent discussion.

The paper provided a lot of guidance in the form of design requirements. So, for instance, aspects of the representation used (binary chromosomes) and the type of GA (generational, using both crossover and mutation, deterministic tournament selection, number of individuals in a tournament) were part of the specification. Similarly, the paper describes the type of graph to be used in the choice of test points for which two prescribed alternatives of the GA parameters are to be compared. The discussion section also bids the students to consider the link between the apparent application area (document clustering) and its potential relevance to online search.

The result of this close guidance is that while it represents a clear challenge, it focusses on the students' ability to achieve the individual goals set in the paper. However, the overall design does not leave them much freedom, and follows a linear progression of steps set by the lecturer. This is an important choice. On one hand, it reduces the overall complexity of the exam. While students should know how to address the required tasks, they do not necessarily have the experience needed to select a viable design *prior* to implementation. Instead, the assignment prescribes a design that can be implemented within the time given (e.g., tournament selection is simpler than one using a roulette wheel, and does not require fitness scaling). Much of the assignment itself leads the student to practice a systematic, experimental approach to the choice of parameters, using the GA tournament size as an example. This type of assigment is very much a guiding hand on the way to full independence, aiming to assess, but also allow the student to practice a number of important concepts (possibly more than would have been covered, had the student a greater freedom to choose his/her course of action). The lesser chance of a major failure also means the assessment is more likely to represent a positive experience for the student, and, consequently, to reach for the techniques and approaches taught in the module in his/her future work.

4.2ALA 2005 & 2006 Assessments

A look at the 2004 histogram of marks (Fig. 1) also showed that, while their distribution is unimodal and symmetrical, the average mark was unusually high for such an assessment, and the spread too low (see Fig. 4). This, and the fact that the upper quartile of marks was in the 90-96% interval was interpreted to mean that the students probably received too much guidance, and the assessment should be more flexible and allow for alternative approaches to give a better chance to students to show their knowledge and creativity.

Various forms of feedback sugested that some students found the genetic algorithms focus of the 2004 paper not giving them the chance to consider what they saw as the central ground of the module scope: learning agents situated in an environment.

In answer to these considerations, the 2005 and 2006 assessments confronted the students with exactly this kind of problem.

> **Setup:** You are provided with a simple multi-agent environment with two types of agent, hunters and prey. All agents are placed on a two-dimensional orthogonal grid. At the beginning of each run, four hunters and two prey are placed in the environment in a (pseudo-)random way. Prey uses a simple, fixed evading behaviour to escape hunters. Similarly, the environment comes with a simple pre-defined behaviour for hunters. The environment is updated in rounds. All agents are prompted in turn to plan their next move, after which all moves are carried out simultaneously. Agents can `see' the contents of all 8 adjacent squares. Both hunters and prey can choose from five possible moves: move in one of the four cardinal directions (North, South, West, East) or stay in the same square. The simulation is run until a hunter and a prey end up in the same square or a maximum number of rounds is reached.
>
> **Goal:** Design, implement, describe and evaluate a procedure that employs evolution and/or learning to produce behaviour that outperforms the default hunters' behaviour. The size of the environment can be changed at will for learning/evolution purposes, but the resulting behaviour should be tested for environments of size 10.

The students were provided with a Prolog agent environment (complete with a manual) developed for this purpose. The paper specified again a list of tasks to be completed, but, in comparison to the 2004 paper, these were formulated on a much more abstract level: design, implement and describe (1) a representation for the agents' behaviour (10% of the marks), (2) how performance will be evaluated (20%), (3) the learning/adaptation procedure to be

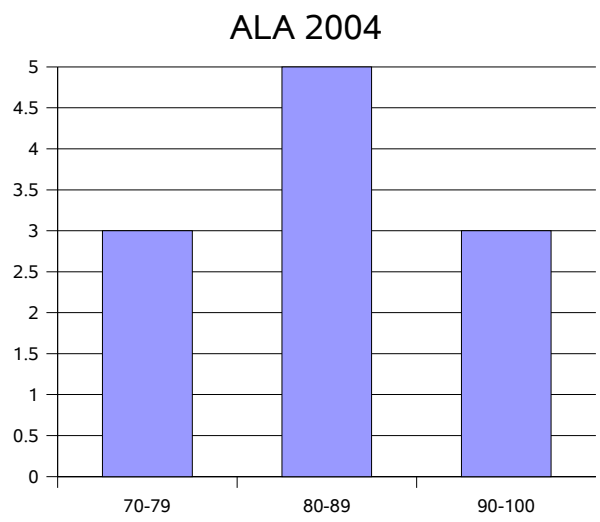

Fig. 1: Spread of marks in 2004

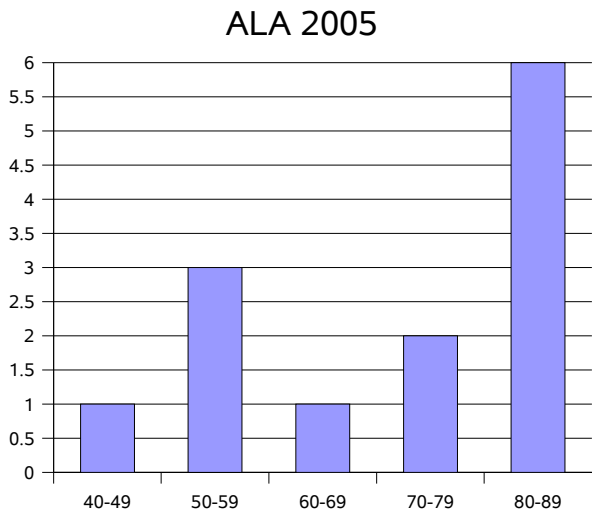

Fig. 2: Spread of marks in 2005

used (40%), and, (4) the evaluation procedure comparing the new and the default behaviour and drawing statistically significant conclusions. In the latter case, 20% of the marks were given for the design, and another 10% for collecting and displaying the results. As in 2004, the students were asked to submit the complete code and instructions needed to reproduce their results, together with a report describing their work and findings.

The 2005 paper offered a much greater degree of freedom in the choice of approach: students could potentially use any of the advaptive/learning approaches covered in the material. The risks associated with this decision were somewhat reduced by providing suggestions about some of the parameters that could be used to evaluate performance, toghether with a suggestion for one possible way of plotting the results. This was meant to provide a step of a more routine nature hence a chance of scoring to the average student while leaving the tasks associated with the bulk of marks open to the competition.

A look at the marks in Fig. 2, and the statistics in Fig. 4 shows a much greater spread, and the desired drop of the average score. Also the distribution now appears to be bimodal (if these relatively small numbers are anything to go by), with a peak in the 2:2 (50-59%) region, and another at distinction level (80+%). This “two hump” distribution is familiar to other CS lecturers [4] and correlates well with what is observed in the introductory Year 1 programming module in York. However, the relatively large gap between the two main modes, and the fact that only one student had achieved a 2:1 (60-69%), the most common grade among MEng students, meant that, despite the additional split of marks between design (30%), implementation (40%), and the quality of the report (30%), perhaps a relatively large proportion of the marks were assigned for tasks where success in one led to success in the next one. The latest design stage, in which the choices made could explain well the gap of around 30 marks between the two modes is the one of selecting the learning/adaptation procedure. If the student has not approached adequately the challenge set in the paper by this stage, there is no real chance of recouping the handicap accumulated even if the subsequent evaluation and analysis stage is done to prefection.

To make is easier to construct a search (adaptation or learning) procedure that could find competitive hunting behaviours, the ALA 2006 exam paper extended the 2005 setup with the ability of hunting agents to communicate through simple signalling (howling). On one hand, this was meant to allow the students to use the link between communication and cooperation discussed in the module. On the other, howling made prey run away, a fact described in the paper, which allowed for some relatively simple hunting strategies to be successful, as the prey tended to concentrate towards the periphery of the environment. The result can be seen in Fig.3: while the distribution is still bimodal, the two modes are now of the same size, and closer to each other. One of the modes now coincides with the 2:1 grade span of marks. In addition, Fig.4 shows a further drop in the mark average (which is now in the middle of the 2:1 interval), and a standard deviation increase.

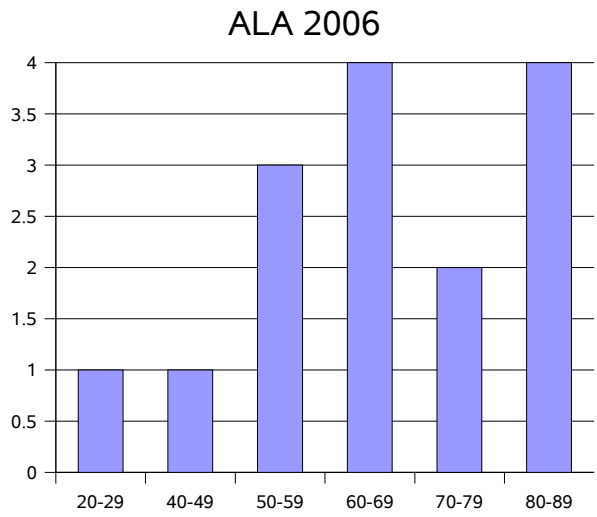

Fig. 3: Spread of marks in 2006

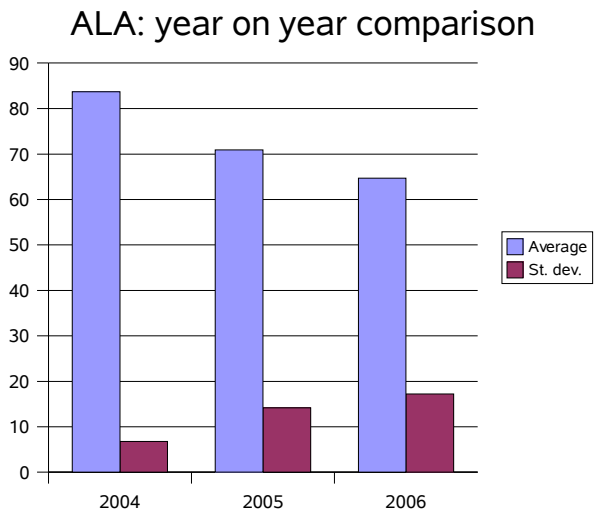

Fig. 4: Annual statistics

5. Collusion and Plagiarism

Collusion and plagiarism are natural worries whenever an open book examination is concerned, particularly when the work is carried out without supervision and over a number of days. To address any possible issues with decisions made in good faith, the following guidelines were issued.

> **ALA 2004:** You are allowed to use any programming language, library or tool to implement the genetic algorithm. Any publicly available code can be used, provided its provenance and location within your source code is clearly indicated. Failure to do so may represent plagiarism with all its implications. Also, no code written by one of the students taking this examination can be used by another.
>
> **ALA 2005 & 2006:** You can use any third party Prolog code/libraries you come across provided they have not been written specifically for this assessment, and you acknowledge them.

In practice, there have been no reasons to suspect any misconduct. Additionally, the disponibility of the actual code made possible to verify much of the claims the students made in the report. Software testing as part of the marking was done on a very extensive scale in the first year; it had to be reduced with the increasing student numbers and task complexity.

6. Sample Solutions and Discussion

It is interesting to have a feel of some of the students' solutions to appreciate the amount of work done and the extent of subsequent achievement. Two examples representing some of the best 2006 submissions will be used, as described by the students. In both cases, a genetic algorithm was used to search for the best hunting behaviour.

Similarly, both students allowed each of the four hunters to evolve a separate behaviour, making possible in this way to discover suitable specialised roles.

> **Example 1:** Two predators move to top right corner while howling (the student's analysis rightly suggests that one would have been enough); the other two move around the bottom, focussing on the bottom left. The situation is illustrated in Fig.5.

One can see why the strategy would be productive. The howling predators drive the prey away, and to the other two members of the pack, who are waiting at the opposite end of the environment.

> **Example 2:** Two predators wander at random in the middle of the environment, howling. The other two move around the edges – one clockwise, the other counterclockwise.

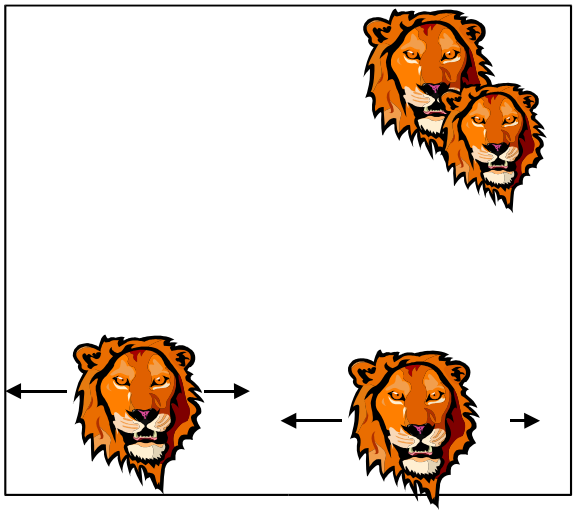

Fig. 5: Evolved predator behaviour I.

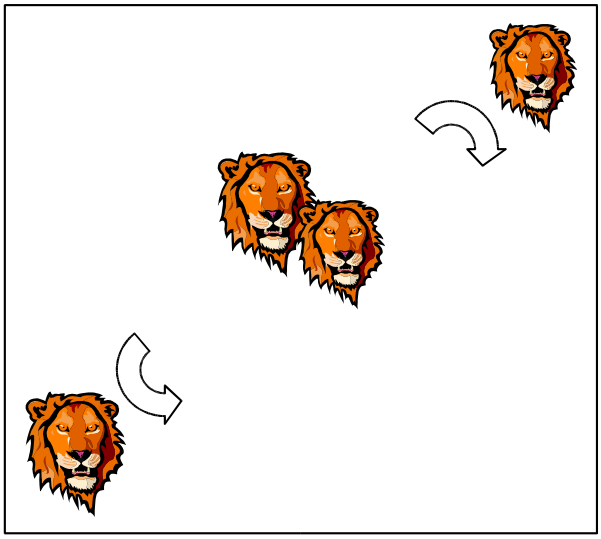

Fig. 6: Evolved predator behaviour II.

Again, there is a clear rationale behind this behaviour: the howling predators will drive the prey towards the edges, where they can be caught in the pincers of the other two predators.

Most marks were for method, not results, but is pleasing to see such interesting results, which would motivate both lecturer and students.

The predator-prey Prolog platform developed by the author has proved useful, and will continue to provide support for future assessments and practicals.[4] Together with the lecture slides, the module Web site, and the two eponymous volumes the author has co-edited on the subject [5, 6], they represent a growing body of material linking the teaching of the ALA MEng module with the latest research in this area. The plans for the future include further developing the lecture notes to form a more consistent body. In Spring 2007, ALA will also be offered for the first time as part of the new MSc in Non-standard Computation.

As ALA addresses an emerging technology, which is likely to become one of the predominant software paradigms in the 21st century, it is hoped and expected that the module will be further developed and offered in future years. We argue that AI teaching at Master's level can successfully teach advanced concepts and approaches, and can – and should – combine the theory with a hands-on element. The benfits of this can be best seen in the results of an open book assessment which combines the right level of supporting structure with sufficient freedom of choice to allow the students to show their potential to the full while providing the best conditions to discriminate between individual students' abilities in a gradual, quantitative fashion.

7. References

[1] Turner, H. and Kazakov, D., Stochastic Simulation of Inherited Kinship-Driven Altruism. *Journal of Artificial Intelligence and Simulation of Behaviour*, **1**(2), p. 183-196 (2002).

[2] Frisch A., Peugniez, T. Doggett, A. Nightingale, P., Solving Non-Boolean Satisfiability Problems with Stochastic Local Search: A Comparison of Encodings. *Journal of Automated Reasoning* (Jan 2006).

[3] Kazakov, D., *Introduction to Programming* Commons Portfolio, http://www.cs.kent.ac.uk/people/staff/saf/dc/portfolios/ (2006).

[4] Dehnadi, S. and R. Bornat. *The camel has two humps.* in Little PPIG 2006. Coventry, UK.

[5] E. Alonso, D. Kudenko and D. Kazakov (eds). *Adaptive Agents and Multi-Agent Systems*. Lecture Notes in Artificial Intelligence (LNAI), vol. 2636, Springer, 2003. 322 pages.

[6] D. Kudenko, D. Kazakov and E. Alonso (eds.). *Adaptive Agents and Multi-Agent Systems II.* Lecture Notes in Artificial Intelligence (LNAI) vol. 3394, Springer, 2005. 312 pages.

[4] The complete Prolog code and documentation are available on request from the author.