The Preference Matrix As A Course Design Tool

John Paxton
Montana State University
Universität Leipzig (Guest Professor)
Computer Science Department
Bozeman, MT 59717 USA
paxton@cs.montana.edu

ABSTRACT
The preference matrix is a theoretical tool based on principles of evolutionary psychology. This paper briefly introduces the theory and then describes how the preference matrix has been applied as a pedagogical design tool for an artificial intelligence course. After making a preliminary assessment of this experience, the paper concludes with several discussion questions.

Keywords
Preference Matrix, Computer Science Pedagogy, Course Evaluation

1. INTRODUCTION
The contribution of this *discussion paper* is to introduce the concept of the preference matrix and its applicability to course design. The preference matrix is a theoretical construct based on principles of evolutionary psychology. The preference matrix provides a lens with which to view course design best practices.

The paper is organized as follows. In section 2, the preference matrix construct is introduced. In section 3, it is explained how the preference matrix was used to design a specific course on the topic of artificial intelligence. Section 3 also describes a first attempt at evaluating the effectiveness of using the preference matrix to design a course. Finally, section 4 raises some discussion questions with respect to using the preference matrix as a pedagogical tool.

2. PREFERENCE MATRIX
The preference matrix is a construct of Stephen and Rachel Kaplan [7]. The Kaplans have synthesized a theory of humans as information processors that is grounded in evolutionary psychology. In order to survive successfully, an individual must be able to recognize objects in the environment (there is a grizzly bear on the edge of the meadow), make predications (the grizzly bear is coming towards me) and evaluate the consequences (this is dangerous). This is done through the use of a mental construct called a cognitive map.

Before introducing the preference matrix, it is important to examine the concept of familiarity. In Table 1 [7], the column labeled "low preference" indicates an environment that an individual does not like very well and the column labeled "high preference" indicates a preferred environment. In a "low preference" environment, a low amount of familiarity results in the individual finding the environment strange while a high amount of familiarity results in the individual finding the environment boring. In a "high preference" environment, a low amount of familiarity results in the individual finding the environment fascinating, while a high amount of familiarity results in the individual finding the environment comfortable.

To increase the likelihood that a person will spend time within an environment, it can be seen from Table 1 that the person's familiarity with the environment is less important than whether the person prefers the environment. In a preferred environment, low familiarity will catalyze the individual to engage with the environment. As a side effect of this involvement, learning will likely take place, leading the individual to function more effectively. In contrast, high familiarity with a preferred environment will foster effective functioning, but will not necessarily catalyze learning to take place.

Table 1. Familiarity matrix

	Low Preference	**High Preference**
Low Familiarity	Strange	Fascinating
High Familiarity	Boring	Comfortable

Table 2 [7] depicts different types of preferred environments. The two key dimensions that lead to an environment being preferred are (1) whether an individual can **make sens**e of the environment and (2) whether an individual can be **involved** with the environment through learning and/or exploration. Making sense and involvement can both be examined from the standpoint of time. When an environment makes sense in the present, it is considered "coherent". When an environment appears that it will make sense in the future, it is considered "legible". When an environment provides involvement in the present, it is considered "complex". And when an environment appears that it will provide involvement in the future, it is considered "mysterious". Note that the CS community is currently exploring the notion of active learning (for example, [4]). A crucial aspect of active learning is involvement.

Table 2 is called a preference matrix because the more of these four traits that are present in a given environment (coherence, legibility, complexity, mystery); the more highly preferred this environment will be.

Table 2. Preference matrix

	Makes Sense	**Involvement**
Present	Coherence	Complexity
Future	Legibility	Mystery

The preference matrix is applicable to any environment, be it natural (e.g. finding one's way in a jungle) or human designed

(e.g. a book). The remainder of this section will focus on implications that the preference matrix provides with respect to designing a course in an educational environment.

"Understanding and respecting the cognitive requirements of the intended recipient constitute probably the single most effective step one can take in improving the process of sharing knowledge". (page 195) [7] The preference matrix provides many immediate useful tips when designing a course:

- To promote making sense, new knowledge should be connected to existing knowledge. Telling a story, using an analogy and/or using a concrete example are all possible techniques for accomplishing this.
- To promote making sense, not more than 5 (plus or minus 2) major concepts should be introduced in any one session. Otherwise the short term memory capacity of the student might be overwhelmed.
- To promote involvement, it is important to develop materials that the learner cares about. Giving the learner some control over the learning process (whether it is through self-paced learning or open-ended assignments) is one way to make this happen. Games [2] also have a high involvement factor.
- To promote involvement, it is important to understand roughly the knowledge the learner brings to the course. Otherwise the learner might judge the environment to be low in "mystery" and consequently be unmotivated to learn.

3. APPLICATION

In this section, one successful pedagogical application of the preference matrix is described. In section 3.1, a brief introduction of an artificial intelligence course is given. In section 3.2, the influence of the preference matrix on this course is provided. In section 3.3, the course is assessed to determine whether the preference matrix has yielded positive benefits.

3.1 Course Overview

CS 436, Introduction to Artificial Intelligence, is a 3 credit, senior-level elective course [10]. The course is offered each fall semester and I have taught this course on 16 consecutive offerings, beginning with the fall of 1990.

The first three weeks of the course are spent introducing the required programming language, Common Lisp. The remaining 12 weeks are spent covering fundamentals of search, knowledge representation and learning in the context of practical applications.

3.2 Preference Matrix Influence

I first learned about the preference matrix when I was in graduate school during the late 1980s. When I began my career at Montana State University, the preference matrix appeared to be a good guideline to use for designing the courses that I would teach.

Although in the case of CS 436 (Introduction to Artificial Intelligence), I have taught the class 16 times and the course has gone through numerous revisions and updates, the core underlying philosophy of using the preference matrix as a course design guide has never changed.

At a very high level, the preference matrix states that a good learning environment is one where (1) a student will be able to "make sense" of the material both now and in the future and (2) a student will be "involved" with the material both now and in the future.

In each offering of the course, some of the designed features of the course that help it "make sense" to the students are

- The course objectives are clearly stated at the beginning of the course.
- A web based syllabus is designed that is simple, complete and easy to use. The syllabus is maintained on a daily basis.
- All exam questions are designed to test a student's comprehension of the course objectives. It is important to foster critical thinking on the part of the students [8].
- All programming assignments are designed to help facilitate a student's comprehension of the course objectives.
- Lecture material is presented that builds upon previous course material and what the typical student should already know. Relationships to previous material are made explicit. (For example, it is pointed out that a best first search can be implemented using a previously studied data structure: the priority queue.)
- Practical applications of lecture material (such as showing a video clip of the 2005 Mohave Desert robot race) are provided regularly.

In the Fall 2005 offering of the course, some of the designed features of the course that enhanced student "involvement" were

- One programming assignment required students to implement the k-means learning algorithm and then apply it to a problem of interest.
- One programming assignment required students to implement a Sudoku problem solver using appropriate search techniques and constraint satisfaction. The programs were evaluated based on how quickly they could solve undisclosed problems of varying difficulty.
- One programming assignment required students to implement a cribbage playing program. A class tournament then allowed the programs to play against one another. Part of the program's grade was based on its performance in the tournament. Part of the program's grade was based on the sophistication of its strategy. The cribbage assignment is a good example of the concept of "mystery". On the first day of the semester, students were told about this assignment. As the semester progressed, students knew that they must actively assemble bits and pieces of the conceptual understanding necessary to succeed on the cribbage assignment.
- During lecture, all students were called on in a systematic fashion to answer questions. Students knew that their answers would not affect their grade. Some lectures were devoted towards philosophical discussions. Dynamic interaction of all forms is an

important mechanism for fostering "involvement" [11].

Although the programming assignments change on every offering, I find that offering open-ended assignments that tap into students' interests is a very effective way to "involve" students with the course. For example, during the Fall of 2005, a Sudoku craze was sweeping campus and many of the students would work a Sudoku puzzle in the campus newspaper on a daily basis. Students were excited to have the opportunity to write a computer program to solve these problems and were surprised at how quickly a well-written program found a solution. Their intrinsic interest in the problem caused them to develop far more sophisticated solutions than what the assignment minimally required.

3.3 Results

In order to conduct an initial assessment regarding the effectiveness of using the preference matrix as a course design tool, I have examined four semesters worth of evaluation data from Fall 2004, Spring 2005, Fall 2005 and Spring 2006. During these four semesters, 19 senior level courses were offered. Two of these 19 senior level offerings were CS 436. Other instructors who do not use the preference matrix as a design guide taught the other 17 offerings. Table 3 shows the evaluation questions that students were given during the last week of the semester before finals week.

Table 3. Evaluation questions

Question	Text
Q1	How does this course compare with similar technical courses?
Q2	What is your level of interest in taking an advanced course?
Q3	Did you find this course challenging?
Q4	Were the objectives of the course clearly stated?
Q5	Were the objectives of the course met?
Q6	How important were the lectures?
Q7	How important were the assignments/programs?
Q8	How important were the tests/quizzes?

Students could respond with one of five answers: 1, 2, 3, 4 or 5. A 1 indicated the most positive response, a 3 indicated a neutral (or

average) response and a 5 indicated the most negative response.

Table 4 shows the results of the evaluation. The first column shows the question being evaluated. The second column shows the mean response for all senior level courses, excluding CS 436. There were 225 responses in this category. The third column shows the mean response for the two offerings of CS 436 that occurred during the four semester evaluation period. There were 33 responses in this category. The fourth category shows the standard deviation for all of the data collected to give the reader a sense of the dispersion. Finally, the fifth column shows the percent of improvement that the third column showed over the second column. To make a mean of 1.0 correspond to a 100% improvement, the percent improvement was computed as follows. Let x be the number from column 2, let y be the corresponding number from column 3 and let z be the percent improvement. Then $z = 1 - ((y - 1.0) / (x - 1.0))$.

As can be seen from Table 4, using the preference matrix as a course design instrument appears to improve the course significantly. For all eight of the questions, there was at least a 10% improvement and for five of the eight questions, there was greater than a 50% improvement. It is also interesting to note that students found CS 436 to be 43% more challenging than other senior level computer science courses. Thus the high evaluations are not due to the course being an easy one.

Table 4. Evaluation results

Question	Non CS-436 Mean	CS-436 Mean	σ	Percent Improvement
Q1	2.19	1.58	0.96	51%
Q2	2.53	1.67	1.29	56%
Q3	1.92	1.52	0.84	43%
Q4	1.83	1.24	0.95	71%
Q5	1.93	1.33	0.91	65%
Q6	1.79	1.52	0.99	34%
Q7	1.82	1.33	0.91	60%
Q8	2.04	1.94	0.98	10%

The three largest improvements are all related to preference matrix factors. Question 4 (were the objectives clearly stated? - a 71% improvement) is a result of helping the students to "make sense" of the course by telling them exactly and repeatedly what concepts they are supposed to incorporate into their cognitive maps. Question 5 (were the objectives met? – a 65% improvement) is a result of helping the students to "make sense" of the course concepts by explaining the concepts in such a way that students can develop a deeper and richer understanding of the concepts. Question 7 (how important were the assignments/programs – a 60% improvement) is a result of choosing programming assignments that get the students "involved". Some of the assignments are open-ended, allowing a student to explore his or her interests. Other assignments are games or puzzles (such as Sudoku), that tap into students' current interests.

It should be noted that the study reported here is an initial one. Although the results are encouraging, there are many factors in addition to the use of the preference matrix as a course design tool that could be influencing the result. Some of these factors include differing courses, differing instructors, and differing sets of students. Isolating these variables is very challenging in practice. Further thought regarding how to conduct more convincing studies is needed.

4. DISCUSSION

In this section, four discussion questions are raised. Based on comments from the reviewers, the second half of the allotted presentation time at the conference was spent facilitating a discussion on the question raised in section 4.2.

4.1 Limitations

The preference matrix is a pedagogical tool grounded in the field of evolutionary psychology. However, no tool is without

its drawbacks. As computer science teachers interested in pedagogy, we strive to be researchers as opposed to students or amateurs [9]. This requires that we examine both the strengths and the weaknesses of any given tool.

Discussion Question: What are the limitations of using the preference matrix as a pedagogical tool?

4.2 Appropriate Research Methodology

The results in section 3.3 show promise with respect to using the preference matrix as a pedagogical tool. However, it is very challenging to isolate and measure individual factors in educational studies.

Discussion Question: What are appropriate research methodologies for measuring the impact of the preference matrix in a convincing manner?

4.3 Visiting Professor Course Assessment

During Winter Semester 2006-2007, I have received a senior lecturing Fulbright Award [5] to develop and offer two courses at The University of Leipzig in Leipzig, Germany. At the department's request, one course will cover web programming topics (with an emphasis on PHP and MySQL) and the other course will cover intermediate level data structures and algorithms, motivated through ACM programming competition problems [1].

Although I will design and teach both of these courses according to fundamental tenets of the preference matrix, I am unsure how to proceed with a meaningful assessment of these courses. Challenges that must be overcome include (1) teaching the courses for the first time, (2) staying at The University of Leipzig for only one semester and (3) cultural differences.

Discussion Question: How can a course be meaningfully assessed when it is offered by a visiting professor?

4.4 Alternative Approaches

The preference matrix has served me well as a pedagogical tool.

However, other broad pedagogical approaches also exist such as Bloom's taxonomy of educational objectives [3] or even Piaget's stages of intellectual development [6].

Discussion Question: How do other pedagogical frameworks compare to the preference matrix? Are these frameworks alternative or complimentary approaches?

5. ACKNOWLEDGMENTS

My heartfelt thanks goes to Stephen Kaplan for introducing me to the preference matrix.

6. REFERENCES

[1] *ACM International Collegiate Programming Contest*. http://icpc.baylor.edu/icpc/ - Accessed July 28, 2006.

[2] Bayless, J. and Strout, S. Games as a "Flavor" of CS1. In *Proceedings of the Thirty-Seventh SIGCSE Technical Symposium on Computer Science Education*. (Houston, Texas, March 1-5, 2006). ACM Press, New York, NY, 2006, 500-504.

[3] Bloom, B. (Editor) *Taxonomy of Educational Objectives: Handbook I: Cognitive Domain*. Longmans, Green and Company. 1956.

[4] Budd, T. An Active Learning Approach to Teaching the Data Structures Course. In *Proceedings of the Thirty-Seventh SIGCSE Technical Symposium on Computer Science Education*. (Houston, Texas, March 1-5, 2006). ACM Press, New York, NY, 2006, 143-147.

[5] *Fulbright Scholar Program*. http://www.cies.org/ Accessed July 28, 2006.

[6] Ginsburg, H. and Opper, S. *Piaget's Theory of Intellectual Development, 3rd Edition*. Prentice Hall, N.J, 1988.

[7] Kaplan, S. and Kaplan, R. *Cognition and Environment: Functioning in an Uncertain World*. Ulrich's, Ann Arbor, MI, 1983.

[8] Krishna Rao, M. Infusing Critical Thinking Skills into Content of AI Course. In *Proceedings of 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*. (Monte de Caparica, Portugal, 2005). ACM Press, New York, NY, 2005, 173-177.

[9] Lister, R. Computer Science Teachers as Amateurs, Students and Researchers. In Proceedings of the 5th Baltic Sea Conference on Computing Education Research. (Koli, Finland, 2005). 2005, 3-12.

[10] Paxton, J. *CS 436, Fall 2005*. http://www.cs.montana.edu/paxton/classes/fall-2005/436/ Accessed July 27, 2006.

[11] Roberts, E. An Interactive Tutorial System for Java. In *Proceedings of the Thirty-Seventh SIGCSE Technical Symposium on Computer Science Education*. (Houston, Texas, March 1-5, 2006). ACM Press, New York, NY, 2006, 334-338.